EVALUACIÓN DE LA VIABILIDAD DE LA IMPLEMENTACIÓN DE UN ESQUEMA DE CONTROL JERÁRQUICO DE DOS FASES EN UN PLC INDUSTRIAL

OSCAR ALEXANDER SARABINO ALEGRÍA
RICARDO JOSÉ TRULLO GUERRERO

Trabajo de grado en Ingeniería en Automática Industrial

Director:
CARLOS ALBERTO GAVIRIA LÓPEZ
Ph.D. Automatización Avanzada y Robótica

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Electrónica Instrumentación y Control
Popayán, Septiembre de 2019

OSCAR ALEXANDER SARABINO ALEGRÍA
RICARDO JOSÉ TRULLO GUERRERO

EVALUACIÓN DE LA VIABILIDAD DE LA IMPLEMENTACIÓN DE UN ESQUEMA DE CONTROL JERÁRQUICO DE DOS FASES EN UN PLC INDUSTRIAL

Trabajo de grado presentado a la Facultad de Ingeniería
Electrónica y Telecomunicaciones de la
Universidad del Cauca para la obtención del
Título de

Ingenieros en:
Automática Industrial

Director:
CARLOS ALBERTO GAVIRIA LÓPEZ
Ph.D. Automatización Avanzada y Robótica

Popayán - Cauca
2019

A mis padres y hermanos.

Oscar Alexander Sarabino Alegría

A papá, mamá y hermanita.

Ricardo José Trullo Guerrero

Resumen

El presente trabajo de grado plantea evaluar la viabilidad de implementar un esquema de control jerárquico de dos capas en un entorno Hardware-in-the-loop-simulation (HILS), tomando como ejemplo de aplicación una columna de destilación y un separador de agua libre FWKO. Se ha desarrollado la creación del entorno de simulación necesario con la inclusión del software Aspen Hysys para la simulación dinámica de las plantas de proceso mencionadas junto con la capa de control básico PID, un PLC industrial del fabricante Siemens, para correr los algoritmos de control del MPC, y el software de cálculo numérico MatLab, como pasarela de información entre el controlador industrial y las plantas simuladas, además de la ejecución del algoritmo de optimización LSSO requerido en la capa superior de la estrategia de control.

Las estrategias multicapa son cada vez más investigadas para su aplicación en procesos complejos (generalmente en la industria química y petroquímica) donde se pretenden optimizar recursos sin dejar de lado características tan relevantes y fundamentales como lo han sido la seguridad de los procesos, el seguimiento correcto de los valores deseados y las regulaciones ambientales. Sin embargo, una de las principales dificultades en la implementación de las estrategias de control avanzadas, es la potencia de cálculo requerida, por lo tanto, se plantea que es posible, mediante el uso de algoritmos simplificados y la interconexión de herramientas de mayor potencia de cálculo, implementar la estructura, o parte de ella para que pueda ser ejecutada por un PLC industrial.

Las pruebas realizadas en el entorno de simulación han mostrado un desempeño considerable por parte de los controladores industriales y las plantas utilizadas debido, especialmente, a factores de tiempo real. Queda como un trabajo interesante para realizar en el futuro desarrollar alternativas a los algoritmos de reducción y simplificación de la programación cuadrática y evaluar su desempeño en la estructura de control, especialmente para la capa de optimización.

Palabras Clave: *Estructura de control jerárquica, Implementación en controladores industriales, Evaluación de viabilidad, Optimización en tiempo real, HILS*

Abstract

The present degree work proposes to evaluate the feasibility of implementing a two-layer hierarchical control scheme in a Hardware-in-the-loop-simulation (HILS) environment, as an example of application a distillation column and an FWKO free water separator. The creation of the necessary simulation environment has been developed with the inclusion of Aspen Hysys software for dynamic plant simulation such as the basic PID control function, an industrial PLC from the manufacturer Siemens, to run the MPC control algorithms, and the MatLab numerical calculation software, as an information gateway between the industrial controller and the simulated plants, in addition to the execution of the LSSO optimization algorithm required in the upper layer of the control strategy.

Multilayer strategies are increasingly investigated for application in complex processes (usually in the chemical and petrochemical industry) where resources are sought to be optimized without neglecting such relevant and fundamental characteristics as process safety, correct monitoring. of the desired values and environmental regulations. However, one of the main difficulties in the implementation of advanced control strategies is the required computing power, therefore, it is suggested that it is possible, through the use of simplified algorithms and the interconnection of tools with greater power calculation, implement the structure, or part of it so that it can be executed by an industrial PLC.

The tests performed in the simulation environment have shown considerable performance by the industrial controllers and the plants used due, in particular, to real-time factors. It remains as an interesting work to carry out in the future developing alternatives to the algorithms of reduction and simplification of quadratic programming and evaluate their performance in the control structure, especially for the optimization layer.

Key words: *Hierarchical control structure, Implementation in industrial controllers, Feasibility evaluation, Real-time optimization, HILS*

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

La necesidad de optimizar el rendimiento de los procesos de producción, de operar las plantas de forma cada vez más flexible y dar respuesta a los requerimientos de las nuevas regulaciones ambientales, se debe al aumento de la competencia en las diferentes industrias, especialmente en las que involucran procesos químicos como las refinerías, las plantas de gas, las destiladoras y las industrias petroquímicas en general. Las técnicas de control de supervisión son generalmente utilizadas para operar las plantas bajo diferentes condiciones como las mencionadas anteriormente, las cuales se caracterizan por estar sujetas a constantes cambios durante su funcionamiento. En los últimos años, es evidente la atención que las industrias han puesto sobre la técnica de optimización en tiempo real (RTO), la cual pretende optimizar el rendimiento de los procesos, los cuales son medidos y evaluados bajo criterios económicos como costos operativos y ganancias [13], convirtiendo la técnica en una herramienta de cuantificación fundamental para la toma de decisiones.

La optimización de los procesos en tiempo real es la clave para mejorar la productividad al mismo tiempo que se disminuyen los costos de producción y se solucionan problemas como la escasez de materias primas y el aumento del costo de consumo de energía al obtener puntos de operación en el proceso que resulten ser más rentables, evitando de esta forma emprender una expansión a gran escala, mejorando la eficiencia de los recursos existentes y disponibles [14].

En general, en las plantas de proceso, una estructura jerárquica es tomada como modelo para abordar la operación óptima. Una de las interpretaciones más sencillas que es posible atribuirle al concepto de estructura de control jerárquico, es el intento de manejar problemas complejos, descomponiéndolos en problemas más pequeños y re ensamblando sus soluciones en una estructura. Normalmente esta estructura consiste en una capa superior que optimiza las condiciones de operación (RTO) para poder actualizar los puntos de ajuste de los controladores dinámicos locales de la capa inferior [15], estos, con frecuencia, son controladores predictivos basados en modelo (MPC).

Una de las principales desventajas en la implementación tanto de estructuras multicapa

como de controladores predictivos, es la potencia requerida por los cálculos formulados en estas estrategias, por lo tanto, se dificulta su estudio y ejecución en controladores industriales de campo. El presente trabajo plantea la implementación de una estructura de control jerárquica en un entorno hardware In The Loop Simulation (HILS). El entorno de simulación permitirá leer y escribir los datos de las variables de proceso, en plantas químicas virtuales de complejidad considerable (columna de destilación y separador trifásico FWKO) a medida que los algoritmos de control son ejecutados por un controlador industrial. El entorno de simulación dinámica de los procesos se desarrolló en el software Aspen Hysys, mientras que los controladores industriales utilizados fueron los PLC Siemens de las series 1200 y 1500.

En el lazo de simulación se integró el software de procesamiento matemático MatLab a manera de pasarela de comunicación, diseño de controladores y generación de código compatible con PLC y como herramienta para la ejecución de la capa de optimización. La comunicación entre el PLC y el entorno de simulación se realizó vía conexión Ethernet y el enlace e intercambio de datos entre el PLC y el conjunto MatLab - Hysys por medio de servidor OPC.

1.1. Motivación

La investigación de las estrategias y técnicas que están a la vanguardia en la ingeniería de control son desarrolladas a menudo por grandes empresas de la automatización o por laboratorios de grandes centros universitarios, los cuales tienen en común la disponibilidad de presupuestos altos de inversión y de investigación y desarrollo. El actual trabajo pretende minimizar la brecha entre las prácticas industriales sofisticadas y las prácticas académicas al mostrar un camino para lograr un entorno de ingeniería de control completo que integre las herramientas y metodologías que están al alcance de los investigadores universitarios y mostrando el potencial que representan estas herramientas para evaluar la viabilidad de grandes proyectos con recursos limitados o escasos, conservando características determinantes como lo es el control en tiempo real.

1.2. Objetivos

Evaluar la viabilidad de implementación en un PLC industrial, de una estrategia de control jerárquico de dos fases.

Parar lograr este objetivo, se presentan a continuación los objetivos específicos con los que se pretende cumplir el objetivo general del presente trabajo.

1. Crear un entorno HILS para simular y controlar un proceso químico, caso de estudio, que permita incorporar un PLC industrial en el lazo.

2. Implementar una estructura jerárquica de dos fases para la fijación de consignas óptimas, en el entorno de simulación construido.

3. Determinar las condiciones bajo las cuales, todos o parte de los algoritmos de optimización requeridos en la capa superior del supervisor, pueden implementarse en un PLC industrial específico.

1.3. Estructura del documento

El presente documento está dividido en cinco capítulos, incluido el actual, que siguen un orden de desarrollo determinado, empezando por la introducción de la base conceptual hasta llegar a los elementos que constituyen la implementación de un esquema de control jerárquico multicapa en un controlador de campo mediante un entorno HILS, terminando con conclusiones y trabajos futuros.

En el Capítulo II, se expone la base teórica de los conceptos necesarios para lograr la implementación del esquema de control jerárquico. En este capítulo están documentados conceptos de la estructura de control como optimización en línea (RTO), controladores predictivos basados en modelo (MPC) y la información referente a los entornos de simulación HILS, sus características y consideraciones para el desarrollo de trabajos como el presente.

En el Capítulo III, se realiza la construcción del entorno de simulación integrando el software de simulación dinámica Aspen Hysys, la herramienta de cómputo numérico MatLab y un controlador industrial PLC de Siemens. En esta sección se describe detalladamente los procedimientos para la interconexión de las diferentes herramientas, tanto físicas como virtuales, así como la función que desempeña cada uno en el funcionamiento global del sistema creado.

La implementación de la estructura de control en el entorno creado se realiza en el Capítulo IV, en donde se realiza el control de dos plantas utilizadas con frecuencia en la industria química, a saber, una columna de destilación de metanol-agua y un separador trifásico FWKO, donde correrán los algoritmos que permiten la optimización en tiempo real, el control predictivo basado en modelo y el control básico en el nivel inferior de la estructura.

En el capitulo V se describe la identificación del modelo y los objetivos de control para la columna de destilación metanol-agua y el separador trifásico FWKO. Asimismo, se detallan los objetivos de cada capa en cada uno de los procesos mencionados anteriormente. Finalmente se da a conocer el desempeño de la estructura de control a nivel de simulación (Matlab/Simulink) y en el entorno HILS (PLC incluido en el lazo)

para cada proceso.

En el Capítulo VI, se presentan las condiciones y requerimientos necesarios para la implementación de los algoritmos en el PLC, y funcionamiento en tiempo real del entorno HILS.

Finalmente, en el Capítulo VII, se presentan las conclusiones resultantes del proceso de implementación y simulación, a la vez que se plantean las líneas de investigación del proyecto y algunas propuestas para trabajos futuros.

Capítulo 2

Marco conceptual

Desde la década de 1970 y, especialmente, en los últimos años, diversas estructuras de control distribuido han sido desarrolladas debido a las notables tendencias en ingeniería química, lo cual conlleva un aumento en la complejidad, diseño y operación de las plantas. Como consecuencia, se ha impulsado la demanda de mejores técnicas y estrategias de control, ya que "un mejor control representa un menor uso de recursos, mayor productividad, mayor seguridad y menor contaminación" [16].

Este aumento en la complejidad de los procesos, junto con el objetivo de eficiencia económica planteado generalmente en las industrias, son tareas difíciles de traducir a puntos de operación para los controladores o estados de las variables en los procesos. Sin embargo, para hacer frente al diseño de controladores en este tipo de situaciones, una idea bien establecida en la práctica industrial y discutida en muchos artículos y monografías es la implementación de estructuras de control jerárquico.

En este capítulo se presenta la base conceptual necesaria para el entendimiento de las estrategias de control jerárquico y los diferentes elementos que la componen, así como la formulación matemática del controlador predictivo basado en modelo y el problema de optimización de consignas en tiempo real.

2.1. Estructuras de control jerárquico

Las industrias de procesos están controladas por estructuras jerárquicas de control donde, en la parte superior, la producción de toda la planta, así como la calidad de los procesos y la utilización de recursos, son determinados por criterios económicos [17].

Alrededor de la literatura existente es probable encontrar la descomposición de las estructuras bajo diferentes criterios, como lo pueden ser la descomposición funcional, espacial y, en algunos casos, una descomposición temporal sustentada en la diferenciación de los periodos que determinan las dinámicas a lo largo del control de un

proceso en sus diferentes tareas, bajo la premisa de lograr el objetivo general de control descomponiendo este en objetivos parciales o subsistemas con tareas específicas acordes a los objetivos de producción. Tatjewski en [16] define la estructura de control jerárquico como el resultado de una descomposición funcional del objetivo de control general, debido a que la realización del objetivo económico del control en línea de una planta industrial se puede expresar como la realización de tres objetivos parciales:

- Garantizar el funcionamiento seguro de los procesos en la planta controlada.
- Mantener las características requeridas de las salidas del proceso (calidad de los productos, etc).
- Optimizar la operación del proceso.

Generalmente para abarcar la tarea de control con una estructura jerárquica, es posible explicar esta a partir de dos capas principales según [16] y [1], a saber: una capa de optimización encargada de calcular los puntos de ajuste o set point de las capas inferiores y una capa de control reguladora, la cual se encarga de mantener los puntos de operación del proceso en estos valores determinados y proveer las características de robustez en general. Cabe añadir que la variación y adición de diferentes características alrededor y en medio de estas capas principales, da lugar a diferentes estructuras de control.

En los sistemas de control más sofisticados, la capa de control de regulación se compone de dos sub capas: una capa de control básico, generalmente compuesta por controladores PID, y una capa superior de control con restricciones [18]. En esta última, comúnmente son implementados algoritmos embebidos de control predictivo basado en modelo MPC, por sus siglas en inglés, los cuales tienen la capacidad única de realizar el control del proceso teniendo en cuenta restricciones en las entradas y salidas de este. Más adelante, en este capítulo, se explica detalladamente el funcionamiento y las características de estos algoritmos.

2.1.1. Optimización de régimen permanente en la estructura de control jerárquica

Como ha sido mencionado anteriormente, el objetivo de la optimización en una estructura de control jerárquico es proporcionar, a los controladores de regulación de las capas más bajas, las mejores trayectorias dinámicas o valores constantes óptimos que conduzcan a alcanzar los criterios económicos establecidos para la operación del proceso, mientras son tenidos en cuenta los diferentes tipos de restricciones de seguridad y límites de operación [1]. Aunque la ubicación de los controladores de regulación, es decir, tanto del controlador con restricciones como del controlador directo no son determinantes para la capa de optimización, la información que esta envía a cada uno de

los controladores si es diferente en cuanto a variables de proceso.

Para explicar la organización de la estructura de control y mostrar el intercambio de información en esta, en primer lugar, se considera el problema de optimización para una estructura sin la capa de control con restricción, es decir, solo con las capas de control directo y optimización como se muestra en la figura 2.1.

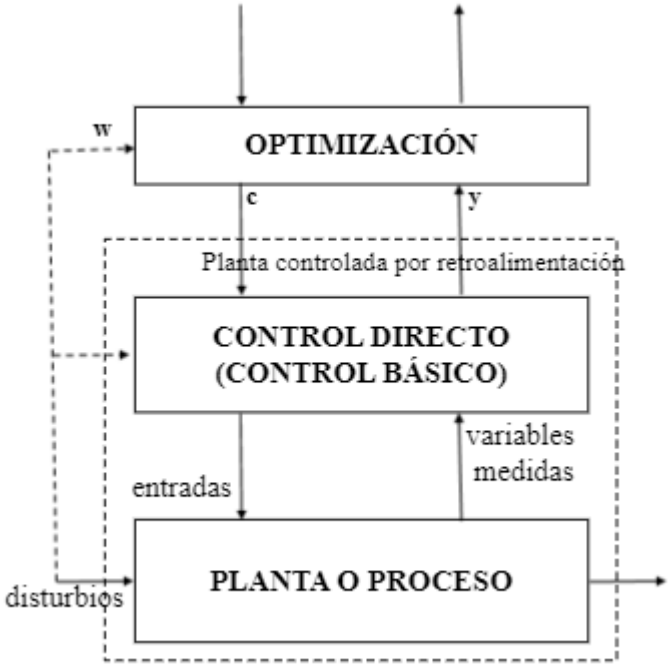

Figura 2.1: Estructura de control jerárquica de dos capas (básica) para optimización de consigas. Tomado de [1]

En la formulación del problema actual, Piotr Tatjewski en [1], aclara que los valores de los disturbios (medidos o estimados) w, son considerados valores constantes, aunque estos sean usualmente valores de variación suave. Sin embargo, los valores de punto de ajuste deben adaptarse a estos cambios para que el proceso se mantenga cerca del funcionamiento óptimo, lo cual se logra resolviendo un problema de optimización a intervalos de tiempo regulares, determinados por las dinámicas de las perturbaciones, o después de cambios significativos de los valores de estas perturbaciones irregulares pero medibles o estimables, para después actualizar los puntos de ajuste de la capa de control directo.

Ahora bien, si el modelo de proceso estático explícito está disponible (hallado o desarrollado), el problema de optimización toma la forma de la ecuación 2.1.

$$\begin{aligned} \text{mín} \quad & Q(c,y) \\ s.a: \quad & y = F(c,w) \\ & c \,\epsilon\, C \\ & y \,\epsilon\, Y \end{aligned} \tag{2.1}$$

En este planteamiento, 'F' denota el modelo de proceso en régimen permanente, 'c' es el vector de variables de decisión o de puntos de ajuste para las variables controladas en el problema de optimización, 'y' es el vector de resultados significativos para el proceso, en términos económicos, y para la satisfacción de las restricciones. 'C' y 'Y' son conjuntos de restricciones [1]. Cabe resaltar en este punto que, gracias a este planteamiento, es posible resolver el problema de "traducir" los objetivos de beneficios económicos de producción en trayectorias o estados de las consignas de los controladores, dado que $Q(c,y)$ de 2.1, asume la dependencia lineal entre los costos de los materiales y la estabilización de los flujos de materia prima y de energía, es decir, entre los requerimientos de planeación y las acciones de control necesarias para lograr los objetivos. Luego, la minimización de la función de rendimiento se puede formular de forma lineal simple como a continuación:

$$Q(c,y) = \sum_{j=1}^{n^J} p_j c_j - \sum_{j=1}^{m^J} q_j y_j \tag{2.2}$$

Donde p_j denota los precios de los flujos de entrada, q_j los precios de los productos de salida y n^J es el número de flujos de entrada controlados. Las variables de salida del proceso 'y' (también denominadas variables de restricción) pueden representar tanto tasas de productos y materiales de desecho, como componentes que no entran en la función de desempeño 2.3, "pero son relevantes para la formulación de las restricciones del proceso", como añade Tatjewski. Por lo tanto, cuando $m^J < dimensión\ de\ y$, la situación es posible y se puede encontrar.

Sin embargo, como existen restricciones significativas capaces de determinar los parámetros de calidad de los productos, no es suficiente tener en cuenta únicamente las restricciones en los valores de salida del proceso para la formulación del problema de optimización restringida, por lo tanto, es cada vez más común la implementación y el diseño de una capa de control de punto de ajuste, jerárquicamente de un nivel superior a la de control directo, para el control de restricción [16].

2.1.1.1. Estructura de control jerárquico con restricciones

En procesos con exigencias de mayor complejidad, la capa de control de regulación adiciona ahora una capa de control de restricción dinámica (denominada también capa de control de punto de ajuste o capa de control avanzado). Esta estructura presenta dos ventajas principales: las restricciones son mantenidas en periodos de transitorios dinámicos en el proceso, como ante cambios en las perturbaciones, y la aplicación de retroalimentación elimina los efectos de la no coincidencia entre el modelo y los errores de estimación en las perturbaciones consideradas para la optimización [1]. En la capa

de control avanzado, generalmente se implementan algoritmos de control predictivo basado en el modelo (MPC), los cuales poseen la capacidad de tener en cuenta las restricciones en las entradas y salidas del proceso. La inclusión de la capa de control MPC en la estructura de control, así como el nuevo intercambio de información entre capas, se puede observar en la figura 2.2.

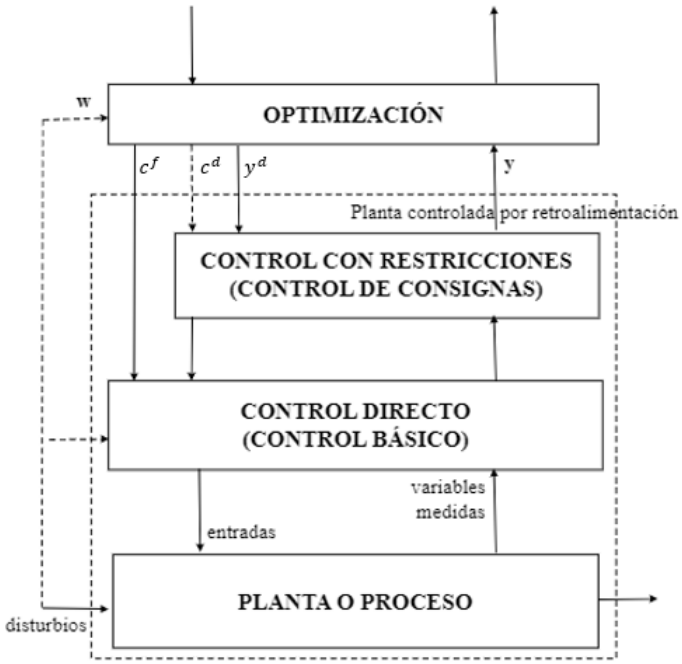

Figura 2.2: Estructura de control jerárquica de dos capas (avanzada) para optimización de consigas con restricciones. Tomado de [1]

Como es posible observar en la figura 2.2, ahora el vector de las variables de decisión del problema de optimización se dividió en dos partes: $c = c^f$, c^d, mientras el vector de salidas del proceso es denotado como 'y^d' como un sub-vector del vector de salidas global 'y'. El modelo de régimen permanente utilizado en la capa de optimización, el cual incluye la información disponible de los valores actuales de perturbaciones 'w', permite el calculo de los puntos de ajuste para la capa de control directo (c^f) y del valor óptimo de régimen permanente para las salidas de los controladores con restricciones (c^d). Por lo tanto, el problema de optimización puede ser replanteado así:

$$\begin{aligned} \text{mín } & Q(c^f, c^d, y^f, y^d) \\ s.a : & (y^f, y^d) = F(c^f, c^d, w) \\ & c^f \,\epsilon\, C^f, \;\; c^d \,\epsilon\, C^d \\ & y^f \,\epsilon\, Y^f, \;\; y^d \,\epsilon\, Y^d \end{aligned} \tag{2.3}$$

En este planteamiento, el valor óptimo de las salidas con restricciones y^d, se transmite

al controlador de restricciones como su punto de ajuste después de cada solución del problema de optimización. A su vez, en este planteamiento el vector w representa valores de entradas no controlados, o perturbaciones, significativos para la solución óptima del proceso, por lo que desde el punto de vista del problema de optimización, estos valores son considerados parámetros y, por consiguiente, deben ser conocidos, medidos o estimados para lograr determinar los puntos de ajuste óptimos del proceso real [1]. Se asumen estas perturbaciones, en una estructura multicapa, de variación más lenta que la dinámica del proceso controlado, lo que significa que el problema de optimización de régimen permanente de la capa superior se puede resolver con una frecuencia mucho más baja que la frecuencia de muestreo de los controladores subordinados de las capas inferiores [19].

2.1.2. Optimización local de régimen permanente LSSO

Existen situaciones en las que, debido a la complejidad del proceso, el problema de optimización también requiere dividirse en capas para llevar a cabo optimizaciones locales de régimen permanente (LSSO), mientras una capa superior se encarga de la optimización global de todo el proceso, apuntando, esta última, a maximizar el rendimiento económico [16] determinando las configuraciones óptimas de régimen permanente para cada unidad de la planta.

Como se ha mencionado, cada capa opera a diferentes frecuencias su tarea de control. Generalmente en la capa más baja de control directo, el muestreo es realizado aproximadamente cada segundo, en la capa de control con restricción o capa MPC, este muestreo es realizado cada un minuto, en el LSSO la repetición de la ejecución puede ser cada una hora y, finalmente, para la optimización global de toda la planta puede ser de un día entero [19].

La capa denominada LSSO se encarga de realizar una optimización con restricciones de una función de objetivo económico mediante un modelo no lineal, ajustando los valores de referencia según la variación de los disturbios y la variación de los requisitos de la capa de optimización superior, sin embargo, como señalan Tatjewski en [19] y Pang en [20], la tarea de optimización es llevada a cabo con efectividad si la variación de las perturbaciones es más lenta que la dinámica del proceso controlado, condición que no se cumple en su totalidad en la mayoría de procesos o no se cumple durante periodos de tiempo significativos, dando origen a pérdidas económicas [13]. Estos problemas se deben a que en la siguiente capa el controlador MPC, en cada instante de muestreo, requiere de los puntos de ajuste deseados que son suministrados por la capa LSSO, para calcular los ajustes óptimos de las futuras variables de salida del controlador y, de esta manera, evaluar los errores de control predichos en su determinado horizonte de predicción [19] (conceptos del controlador MPC que serán expuestos con detalle en la siguiente sección de este capítulo). Por lo tanto, mantener el punto de referencia mientras se espera que la capa LSSO realice un nuevo cálculo, a pesar de cambios en las perturbaciones, da lugar a las pérdidas mencionadas.

Como se menciona en [19], se piensa que la solución adecuada sería ejecutar el LSSO con tanta frecuencia como sea necesaria, incluso a la frecuencia de muestreo del MPC, pero la dificultad de esta solución radica en la diferencia entre los modelos de la capa LSSO y la capa MPC, a saber: mientras el controlador MPC utiliza un modelo de proceso dinámico, la LSSO requiere un modelo de régimen permanente no lineal del proceso para realizar la optimización no lineal con restricciones. Por estas condiciones, no es posible que la capa LSSO realice su tarea a la misma frecuencia que es requerida por el controlador MPC.

Como consecuencia de lo anterior, en estas estructuras jerárquicas, para reducir la brecha entre la frecuencia considerablemente rápida de la capa de control MPC y la capa de optimización de régimen permanente LSSO de baja frecuencia, se divide la capa MPC en dos: una capa de cálculo de objetivo de régimen permanente SSTC (también denominada optimización de objetivos de régimen permanente SSTO) y una capa denominada de optimización dinámica (OD) [16].

2.1.3. Optimización de objetivos de régimen permanente SSTO

“La tarea de esta capa es calcular los puntos de ajuste, tanto para las variables controladas como para las variables manipuladas del LSSO cada vez que se ejecuta el MPC” [18]. Como se explicaba al final de la sección 2.1.2, la razón principal radica en la diferencia de frecuencia de ejecución de la capa LSSO y la capa MPC, por lo que ahora, las perturbaciones y la nueva información de entrada al sistema puedan modificar la ubicación de los puntos óptimos de régimen permanente, sin generar pérdidas de optimalidad por retrasos de la capa LSSO.

En [21], los autores mencionan el concepto de valores de reposo ideales (IRV) como el cálculo que realiza ahora la capa SSTC cuando ocurre el retraso, por parte de la capa de optimización superior LSSO, en proveer estos al controlador MPC, es decir, que la capa MPC, de dos etapas, ahora se encuentra en capacidad de recalcular los puntos óptimos de ajuste según la función de rendimiento económico original. Esta optimización de régimen permanente es un problema de programación lineal o programación cuadrática y fue llamado cálculo de objetivo de régimen permanente por Kassman en [17]. La estructura de control resultante, con la frecuencia típica de ejecución por capa es la que se observa en la figura 2.3.

Como se ha venido explicando, al incluir la subcapa SSTO para mejorar la integración entre la LSSO y el controlador MPC se espera:

- Minimizar las inconsistencias entre el modelo no lineal de la capa LSSO y el modelo lineal utilizado en la capa MPC.
- Tener una reacción más rápida de la estructura de control, como respuesta a la aparición de nuevas perturbaciones.

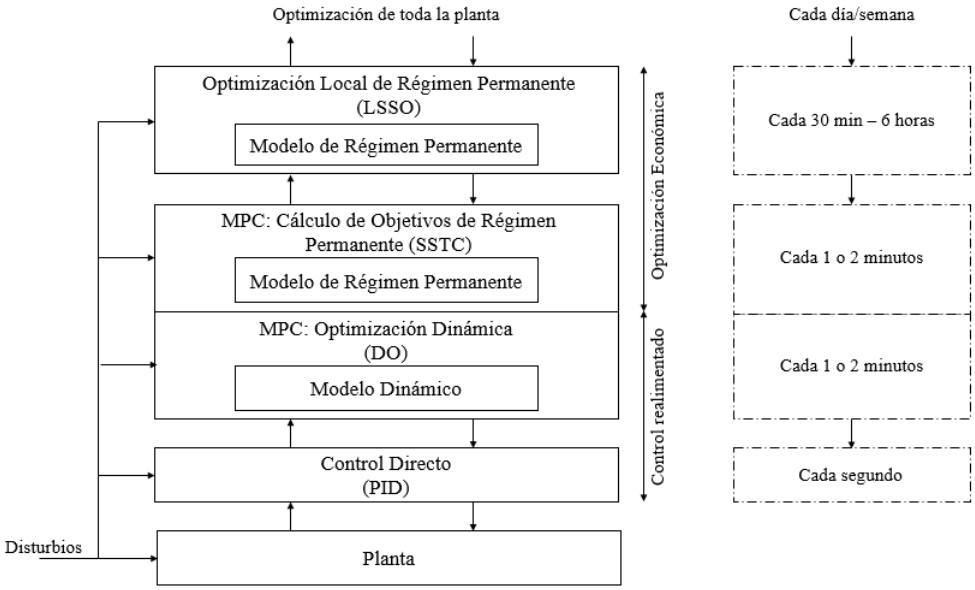

Figura 2.3: Estructura de un sistema de control jerárquico con optimización de objetivos de régimen permanente SSTO. Tomado de [4]

- Evitar grandes pasos en los cambios de punto de ajuste que pueden causar inestabilidad.
- Lidiar con la compensación de los objetivos deseados de una forma controlada y optimizada [22].

Como el cálculo de los estados estables disponibles de la LSSO es realizado por la SSTC cada vez que se ejecuta el controlador MPC, esta capa debe usar un modelo de proceso simple, el cual en la práctica suele ser un modelo lineal [17]. Este modelo, usualmente, es una versión de régimen permanente del modelo dinámico lineal (matriz de ganancia) utilizada en el algoritmo MPC.

En el caso de que la función del objetivo económico sea lineal, (en [19] se menciona que lo es en la mayoría de los casos) la solución se convierte en un problema de programación lineal (PL), el cual, aunque se puede resolver en cada instante de muestreo, puede tener como consecuencia pérdidas en la optimalidad económica debido a que los modelos, tanto el lineal mencionado y el utilizado en la capa LSSO, pueden ser considerable y significativamente diferentes y, por lo tanto, las predicciones obtenidas a través de ellos pueden no ser consistentes cuando el proceso está operando cerca del punto de linealización [22].

Para la implementación de la subcapa SSTO, generalmente se implementa con uno de los siguientes enfoques, el primero, como se menciona en [20] y [16], es usar la aproximación cuadrática del modelo de proceso no lineal propuesto en [21] y [23], sin embargo, este método también presenta el problema de inconsistencias entre el

modelo de régimen permanente no lineal y el modelo dinámico del MPC. El segundo enfoque se trata de recalcular los puntos de ajuste para optimizar la misma función económica de la capa RTO utilizando un modelo lineal de régimen permanente más simple, como se muestra en la ecuación 2.4.

$$\begin{aligned} u_{SSTO}, y_{SSTO} &= argmin_{u,y}\, \Gamma(u,y) \\ & s.a: \\ y &= A_s u + \varepsilon(t|t) \\ y^{eq} &= y^s \\ y^L \leq\ & y^{in} \leq y^U \\ u^L \leq\ & u^{in} \leq u^U \end{aligned} \tag{2.4}$$

Es posible observar que este planteamiento del problema es una versión simplificada de la estructura LSSO, donde y_{SSTO} y u_{SSTO} son los objetivos ajustados por la subcapa SSTO; $\Gamma(u,y)$ es la misma función económica utilizada en LSSO; $y^{eq}\, \epsilon\, R^{n_y^{eq}}$ es el conjunto de salidas con restricciones de igualdad para las cuales y^s son los valores de punto de ajuste; $y^{in}\, \epsilon\, R^{n_y^{in}}$ es el conjunto de salidas restringidas de desigualdad para las cuales y^L y y^U son los limites inferior y superior, respectivamente; y u^L y u^U son los limites inferior y superior de las variables de decisión; A_s es la matriz de ganancia estática y $\epsilon(t \mid t)$ es la perturbación. Cabe resaltar que tanto A_s como $\epsilon(t \mid t)$ son tomados del algoritmo MPC, por lo que se garantiza que los modelos coincidan dentro de la capa MPC [22], ahora de dos etapas.

Con la SSTO, incluida ahora en la estructura de control jerárquica, los siguientes cálculos se realizan mediante un controlador-optimizador de restricciones MPC en cada instante de muestreo:

1. Adquisición de las salidas medidas.
2. Predicción de la trayectoria de salida libre para calcular las salidas futuras.
3. Solución del problema de programación lineal en la subcapa LSSO, para obtener los objetivos de régimen permanente en el problema de optimización dinámica.
4. Solución del problema de optimización dinámica QP, para obtener la salida actual del controlador como el primer elemento de la trayectoria de entrada de control óptima calculada [1].

2.2. Controlador basado en el modelo MPC

El control predictivo basado en modelo es una estrategia de control que hace uso del modelo del proceso para predecir las salidas futuras de la planta y, con base en ello,

optimizar las acciones de control futuras; de manera más concreta, un modelo matemático explícito de la planta.

Para lograr la tarea de optimización, en cada intervalo de control, el algoritmo resuelve una serie de problemas que resuelven tres programas no lineales, y los cuales, dan respuesta a las siguientes preguntas en el marco de control de procesos:

- ¿Dónde se encuentra el encabezado del proceso? (Estimación del estado)
- ¿A dónde debe ir el proceso? (Optimización de objetivo de régimen permanente)
- ¿Cuál es la mejor secuencia de ajustes para enviar el proceso a la optimalidad? (Optimización dinámica) [24]

Por estas razones, el control predictivo no se puede considerar como una estrategia de control independiente o aislada, sino que, por el contrario, integra toda una familia de métodos de control.

2.2.1. Introducción al algoritmo MPC

Durante mucho tiempo, el objetivo más importante en el control de procesos era alcanzar la estabilidad de la operación de la planta; sin embargo, en la actualidad, debido a la variación de los mercados y a la alta competitividad, las empresas se han visto en la obligación de mejorar sus procesos de producción. Otro factor que impulsado la investigación de estrategias de control fiables es el creciente interés de la sociedad por los problemas ambientales causados por procesos industriales ineficientes [2].

El control predictivo basado en modelo se ha convertido en una herramienta capaz de afrontar objetivos como: tomar en cuenta las restricciones en las entradas y salidas de proceso y generar entradas de proceso teniendo en cuenta las interacciones dentro del proceso mismo, debido al uso de un modelo [1]. En la actualidad estos objetivos se encuentran en el centro del desarrollo de estrategias de control avanzadas, debido a que pueden reflejar directamente los criterios más relevantes en la industria, a saber: el cumplimiento de los criterios de calidad en la producción, de los criterios económicos de planeación, con los criterios de seguridad y con los mencionados criterios medioambientales [2].

A grandes rasgos, a lo largo de la literatura, el control predictivo se ha desarrollado alrededor de un conjunto de principios comunes, como lo son:

- Uso explícito de un modelo del proceso para predecir la salida del sistema en instantes de tiempo futuros
- Cálculo de acciones control óptimas, como resultado de la minimización de funciones de costo y la inclusión de restricciones sobre las variables del proceso

Por lo tanto, la diferencia entre los algoritmos de tipo MPC se debe principalmente a dos razones: el tipo de modelo utilizado para representar el proceso y las perturbaciones, y en segundo lugar, la función de costo que se minimiza (con o sin restricciones) [2].

Esta tecnología surge primero en el contexto de la industria petrolera, específicamente en las refinerías. También en el control de centrales eléctricas, desde hace varias décadas [25]. Los resultados obtenidos conllevaron a una ola de aplicaciones industriales exitosas más adelante, por lo que, en la actualidad, los algoritmos MPC están embebidos en una amplia variedad de áreas como lo son la industria química, automotriz, aeroespacial y en el procesamiento de alimentos también. Según Quin y Badwell en [21], en el año 2003, el número de aplicaciones MPC en el mundo estaba estimado en 4.500. Puntualmente, existen anotaciones acerca de MPC implementado en diversos procesos como por ejemplo los mencionados en [2]: en la industria de cemento [26], generadores de vapor [25], columnas de destilación [27], anestesia clínica y robótica [28].

2.2.2. Ventajas y desventajas del algoritmo MPC

La estrategia de control predictivo presenta un conjunto de ventajas sobre los métodos de control tradicionales. Son parte de este conjunto:

- Puede manejar problemas de control mono variables, multivariables, lineales y no lineales de forma sencilla y utilizando la misma formulación del controlador
- Los principios de funcionamiento son intuitivos y los parámetros de diseño están orientados al desempeño, lo que permite acortar el tiempo de capacitación en operarios de planta [2]
- La formulación en el dominio del tiempo es flexible, abierta e intuitiva
- Es una aproximación natural al control con restricciones (restricciones en los actuadores, restricciones de seguridad en la operación y restricciones de parámetros de calidad)
- Permite controla procesos con dinámicas difíciles y poco usuales como procesos de fase no mínima, procesos oscilatorios o procesos inestables
- La ley de control responde a criterios óptimos
- La naturaleza predictiva del controlador permite compensar intrínsecamente los tiempos muertos y las perturbaciones medibles
- Es una estrategia escalable y flexible, como se ha venido explicando, en la incorporación de diferentes estructuras jerárquicas de control de procesos

Sin embargo, como se espera, esta técnica también presenta ciertas desventajas dentro de las cuales es posible citar las siguientes:

- Requiere el conocimiento de un modelo dinámico del sistema suficientemente preciso
- Como requiere un algoritmo de optimización, solo puede implementarse por un computador
- Requiere un alto coste computacional, por lo que su aplicación e implementación en sistemas rápidos se dificulta
- La sintonización de estos controladores, especialmente en el caso con restricciones, se realiza de forma heurística y sin un conocimiento demasiado específico de la influencia en el cambio de parámetros sobre la estabilidad del proceso

2.2.3. Formulación del MPC

El controlador predictivo basado en modelo está conformado por un conjunto de elementos generales, los cuales serán explicados, de manera introductoria y básica, a continuación. En esta sección se pretende enumerar y definir dichos componentes. Se presenta también junto con algunos conceptos, una formulación matemática que servirá de apoyo en la comprensión de estos, pero cabe aclarar que en la siguiente sección de este mismo capítulo serán explicados de manera más explícita.

2.2.3.1. Modelo de predicción

Este es el modelo que describe el comportamiento esperado del sistema, es cuál puede ser lineal o no lineal, continuo o discreto, en variables de estado o en función de entradas y salidas. Debido a que el problema de optimización y la técnica de horizonte deslizante con la que se aplica la solución son resueltos mediante computadoras, las explicaciones de los modelos son frecuentemente explicadas mediante modelos discretos más que con modelos continuos. De la misma forma, los modelos en espacio de estados son más generales que los modelos de entrada-salida, por lo que en esta sección se adoptan estas formulaciones.

Por lo anterior, el modelo de predicción tiene la forma:

$$x(k+1) = f(x(k), u(k)) \tag{2.5}$$

Donde $x(k)$ es el estado y $u(k)$ es son las actuaciones sobre el sistema en el instante de muestreo k. Cabe resaltar que cuando el sistema presente incertidumbres, estas pueden aparecer en el modelo de predicción, considerándose su efecto en la predicción del comportamiento futuro de la planta.

2.2.3.2. Función de coste

Es la función encargada de indicar el criterio de optimización, donde expresa el coste asociado a una determinada evolución del sistema a lo largo del horizonte de predicción Np. Debido a que la función de coste considera el comportamiento del sistema hasta el final del horizonte Nh, esta función depende del estado actual del sistema $x(k)$ y de la secuencia de N actuaciones aplicadas durante todo el horizonte de predicción Nc. Generalmente de considera constante la señal de control a través del horizonte de control.

2.2.3.3. Restricciones

Las restricciones indican los límites, tanto superiores como inferiores, dentro de los cuales debe ocurrir la evolución del sistema y de las señales, donde estas últimas no deben exceder estos límites, ya sea por motivos de seguridad o por limitaciones físicas de las variables. Por ejemplo, los límites de los actuadores forman parte de las restricciones.

La incorporación de estas restricciones surge de la necesidad de trabajar en puntos de operación que se ubican cerca de los límites físicos admisibles. Dichas necesidades, generalmente, son producto de directrices económicas de planeación para la optimización de recursos como materias primas y consumo energético [16].

Estas restricciones usualmente son expresadas como conjuntos X y U, generalmente cerrados y acotados, en los que deben estar contenidos los estados del sistema y las acciones de control en cada instante de muestreo k, de manera que quedan expresados así:

$$\begin{aligned} x(k) \,\epsilon\, X \; Vk \\ u(k) \,\epsilon\, U \; Vk \end{aligned} \tag{2.6}$$

2.2.3.4. Optimización

Como se ha venido explicando, el controlador MPC está asociado también a un problema de optimización de sus puntos de operación o trayectorias que determinan la eficiencia económica del proceso. Siguiendo el planteamiento de los elementos hasta aquí presentados, dicho problema del controlador debe resolver en cada instante de muestreo k el siguiente problema:

$$
\begin{aligned}
&min_{u(k)}\ J(x(k), u(x)) \\
&s.a: \\
&u(k+j \mid k)\ \epsilon\ U\ para\ j = 0, 1, ..., N_c - 1 \\
&x(k+j \mid k)\ \epsilon\ X\ para\ j = 0, 1, ..., N_p - 1 \\
&x(k+j \mid k)\ =\ f(k+j \mid k, u(k+j \mid k))\ \ para\ j = 0, 1, ..., N_p - 1 \\
&u(k+j \mid k)\ =\ u(k+j-1 \mid k)\ \ para\ j = N_c, N_c + 1, ..., N_p - 1
\end{aligned}
\tag{2.7}
$$

El anterior problema de optimización tiene como variables de decisión las actuaciones a lo largo del horizonte de control y depende de forma paramétrica del estado del sistema. Una vez obtenida la solución, según la estrategia del horizonte deslizante, se aplica la actuación obtenida para el instante siguiente y se resuelve nuevamente el siguiente periodo de muestreo $u(k \mid k)$ [1].

2.2.3.5. Principio de funcionamiento del MPC

A continuación, para explicar de forma detallada y general los principios, conceptos, parámetros y funciones del controlador MPC, se recurre, principalmente, al modelo y explicación de Piotr Tatjewski en el capítulo 3 de [1], debido a que en la mayoría de artículos y trabajos posteriores a esta publicación, el libro es mencionado como un referente base.

En primer lugar, este principio general del MPC se describe en cada instante de muestreo $k = 0, 1, 2...$, teniendo:

- Un modelo de proceso dinámico junto con un modelo supuesto de perturbaciones y modelos de restricciones
- Mediciones de las salidas actuales y pasadas del proceso, junto con los valores pasados de las entradas de control (variables manipuladas)
- Trayectorias conocidas o supuestas de puntos de ajuste para las variables controladas (salidas de proceso controladas) para un horizonte de predicción supuesto [1]

En segundo lugar, para poder explicar el paso a paso del funcionamiento, se describe la notación matemática utilizada a lo largo de la misma [2]:

- k: representa el índice del tiempo discreto $(k = 0, 1, 2, \dots,)$
- $u(k)$: representa la entrada del proceso (variable manipulada)
- $y(k)$: representa la salida del proceso (variable controlada)
- $w(k)$: representa el punto de operación (set-point)

- $u(k+j \mid k)$: representa los valores futuros de la entrada, calculados en el instante de tiempo k

- $y(k+j \mid k)$: representa los valores futuros de la salida con base en:
  Mediciones disponibles en el instante
  $k : y(k), y(k-1), \ldots, u(k-1), u(k-2), \ldots$
  Valores futuros de la entrada en el instante
  $k : u(k), u(k+1 \mid k), \ldots$

En la figura 2.4, es posible observar el principio básico de funcionamiento del control predictivo, el cual se caracteriza por la siguiente estrategia:

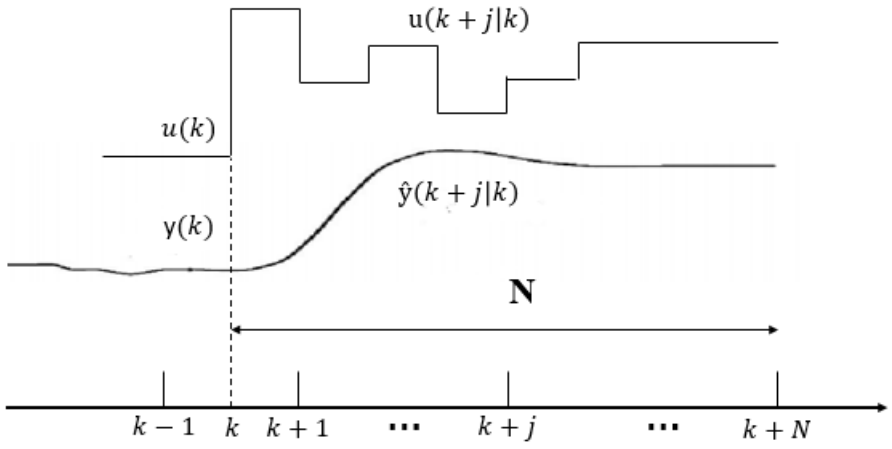

Figura 2.4: Estrategia del control basado en modelo MPC. Tomado de [2]

- En cada instante de tiempo k, se predice la salida de proceso $y(k+j)$ sobre un horizonte de tiempo $k = 1, 2, \ldots N$

$y(k+j \mid k)$ indica los valores futuros y el valor N es el denominado horizonte de predicción. Dicha predicción es realizada por medio del modelo del proceso, por lo que es fundamental que este se encuentre disponible; esta depende de las entradas y salidas pasadas, pero también del escenario de control futuro $u(k+j \mid k), k = 0, \ldots, N-1$; es decir, las acciones de control que serán aplicadas desde el instante de tiempo presente k en adelante.

- Se define una trayectoria de referencia $r(k+j \mid k), k = 1, \ldots, N$, sobre el horizonte de predicción, que comienza en $r(k \mid k) = y(t)$ y evoluciona hacia el punto de operación w. En caso de que el proceso presente tiempo muerto, es razonable iniciar dicha trayectoria después de este

- Se calcula el vector de control $u(k+j \mid k), k = 0, \ldots, N-1$ para minimizar una función de costo específica, dependiendo de los errores de control predichos $[r(k+j \mid k) - y(k+j \mid k)], k = 1, \ldots, N$
- El primer elemento $u(k \mid k)$ del vector de control óptimo $u(k+j \mid k), k = 0, \ldots, N-1$ se aplica al proceso real en el instante de tiempo actual. Los demás elementos del vector de control calculado pueden ser despreciados u olvidados ya que en el siguiente instante de muestreo todas las secuencias de tiempo se desplazan y se obtiene una nueva medición de la salida $y(k+1)$ repitiendo todo el procedimiento nuevamente [1]. Esto lleva a una nueva entrada de control $u(k+1 \mid k+1)$, la cual es generalmente diferente de la calculada previamente $u(k+1 \mid k)$; este concepto es denominado horizonte deslizante

El anterior planteamiento secuencial de la estrategia de control MPC, ha sido tomado de [2] y [1] para la explicación explicita y resumida de esta técnica. En la figura 2.5 se muestra la estructura de implementación del control predictivo, así como sus componentes necesarios.

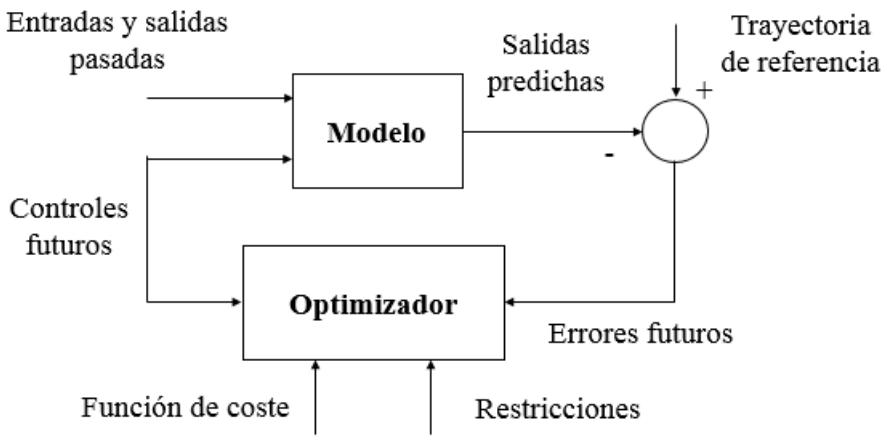

Figura 2.5: Estructura básica del MPC. Tomado de [2]

En la figura anterior, es posible observar el uso del modelo de la planta, con el objetivo de predecir la evolución de la salida del proceso a partir de las señales de estrada y salidas conocidas. Se calculan las acciones de control futuras con el optimizador, que considera la función de costo y las restricciones [2].

También cabe mencionar, que el modelo de proceso utilizado debe ser lo más cercano posible al modelo real; de esta forma es posible capturar la dinámica del proceso para obtener los mejores resultados de control logrando predecir de forma más precisa la evolución del sistema. Sin embargo, al mismo tiempo este modelo debe ser lo suficientemente simple de implementar para mitigar la carga computacional asociada al

proceso de predicción a lo largo de todo su horizonte finito. Existen diferentes tipos de controlador MPC, a saber, DMC, GPG, EPSAC, NEPSAC entre otros; los cuales difieren fundamentalmente en el tipo de modelo utilizado.

Por otra parte, se tiene en la estructura básica el optimizador, el cual provee al controlador de las acciones de control que deben ser aplicadas. Generalmente se utiliza un modelo lineal, lo cual deriva en una función de costo cuadrática, y si tampoco se tienen restricciones, la solución obtenida no genera un calculo computacional relativamente alto. Pero, por otra parte, si se incurre en un caso diferente en el que se acuda a algoritmos de optimización, la carga computacional aumenta considerablemente [2].

Finalmente, también se considera el hecho de que, aunque el tamaño del problema resultante depende de los parámetros del controlador MPC como número de variables del proceso, tamaños de los horizontes de control y predicción elegidos en el diseño y sintonización del controlador, y de las restricciones consideradas, generalmente, los problemas de optimización que resultan no suelen ser de una carga computacional demasiado pesada, sin embargo, como se verá más adelante, existen procesos con una complejidad exigente (especialmente en la industria química y petroquímica), lo que genera una clara desventaja en la implementación de estos algoritmos en controladores con poca capacidad de cálculo, como por ejemplo los PLC industriales.

2.2.4. Implementación de algoritmos MPC

Debido a la experiencia recogida y combinada de un gran número de aplicaciones MPC en las industrias, ha sido posible encontrar en la literatura un consenso cercano acerca de los pasos necesarios para lograr una implementación exitosa, en este caso enumeramos los que lograron sintetizar Thomas Badwell y Joe S. Qin.en [24] recientemente en el año 2015, como se muestra a continuación:

- Justificación: “justificar el caso económico de la solicitud”
- Prueba previa: diseñar los sensores y actuadores de control y probarlos
- Prueba de paso: generar señales de respuesta del proceso
- Modelado: desarrollar un modelo a partir de los datos de respuesta del proceso
- Configuración: configurar el software y probar el ajuste preliminar mediante una simulación
- Puesta en marcha: encender y probar el controlador
- Evaluación posterior: medir y comprobar el desempeño económico
- Mantenimiento: monitorear y mantener la aplicación MPC

Los autores de la anterior recopilación y sinterización de pasos para la implementación de un algoritmo MPC, resaltan el hecho de que, tanto en términos de costos por pérdidas de producción como de tiempo de ingeniería, el más caro de estos pasos es “la generación de señales a través de la prueba de pasos” [24]. Lo anterior se debe a que esta prueba es llevada a cabo haciendo ajustes significativos en cada una de las variables que el MPC ajustará mientras opera en lazo abierto para evitar la acción de control de compensación; por consiguiente, estos ajustes causarán respuestas anormales en los puntos operativos clave, lo que probablemente conduzca a un rendimiento más bajo de la planta y a que los productos queden fuera de las especificaciones normales. En los últimos años, con el objetivo de mitigar las dificultades de este paso en particular, se han obtenido avances mediante el uso de pruebas de pasos en lazo cerrado aproximadas [29].

2.3. Entorno Hardware-In-The-Loop-Simulation (HILS)

Los entornos de simulación virtual proveen a investigadores, ingenieros y estudiantes, de potentes herramientas informáticas con las cuales reproducir fenómenos físicos (mecánicos, eléctricos, químicos o una combinación de estos), evaluar la ejecución de sistemas de control, entre otros, con un alto grado de certidumbre en los resultados. No obstante, las dinámicas de los sistemas físicos poseen diversas no linealidades a tomar en cuenta y en muchos casos no es posible obtener una representación o modelo (generalmente matemático) que las reproduzca fielmente con el fin de observar y analizar su desempeño en un entorno de simulación específico. A pesar de lo anterior, las herramientas de estimación y modelado de fenómenos físicos, tanto analíticas como computarizadas, arrojan modelos lo suficientemente cercanos a la realidad, permitiendo que sistemas externos como dispositivos hardware interactúen con ellos a través de un medio de comunicación, conformando una estructura de simulación en la cual poner a prueba algoritmos de control, modelos computarizados de sensores y/o actuadores, equipos industriales, etc.

En la presente se sección se exponen trabajos realizados bajo un entorno Hardware-In-The-Loop-Simulation (HILS), con el fin de dar a conocer la arquitectura básica del entorno, las herramientas que lo articulan y los objetivos que satisface su implementación.

2.3.1. Aspectos generales de entornos HILS

Generalmente, en el diseño, instalación, puesta en marcha y mantenimiento de un sistema de control para una planta industrial, (que específicamente para este trabajo se trata de un sistema de control que adopta una estructura jerárquica para la optimización de consignas en línea) es necesario conocer a priori cómo va a comportarse bajo diferentes condiciones de funcionamiento, tales como variación en los parámetros de

sintonía de los algoritmos de control, disturbios en los flujos de energía y/o materia prima, latencia presente en las señales de control, cambios físicos en la planta, entre otros. Muchas veces, observar y analizar el comportamiento del sistema de control en una planta real, conlleva altos costos económicos y puede ocasionar daños en los equipos que conforman la planta debido a violaciones de los puntos de operación seguros del proceso [5]. Por ello, las simulaciones en softwares especializados en modelado matemático y numérico arrojan buenos resultados sobre el comportamiento de un sistema a bajo costo y sin poner en riesgo la planta. Sin embargo, al trabajar netamente en softwares de simulación, no es posible involucrar toda la dinámica del sistema real, perdiendo información relevante para la correcta configuración del mismo. En vista de lo anteriormente mencionado, implementar un entorno HILS se traduce en un mayor grado de certeza en la información obtenida, dado que gran parte de los elementos que articulan un sistema de control (actuadores, sensores, controladores, redes de comunicación industrial, etc) son susceptibles de funcionar en el lazo, dejando que el proceso industrial bajo estudio, se ejecute en un software de simulación especializado de altas prestaciones [30, 5, 31].

Principalmente los entornos HILS están articulados por 3 componentes [32]:

1. Simulador de altas prestaciones para el proceso o procesos bajo estudio.

2. Equipos para la ejecución de los algoritmos de control, los cuales pueden ser microcontroladores, FPGA's, PLC's, entre otros.

3. Sistema de comunicación para el intercambio de datos entre el simulador y el equipo de control.

Sin embargo, los entornos HILS también pueden involucrar otra clase herramientas hardware y/o software que enriquezcan la investigación y análisis del problema bajo estudio, con el objetivo de considerar la mayor cantidad de variables que inciden en el mismo y obtener resultados cercanos al comportamiento real del proceso (planta y sistema de control) [4]. En la figura 1 se pueden apreciar la implementación de algunos entornos HILS. (sacar imágenes de los artículos para hacer un collage).

En resumen, los entornos HILS se articulan de acuerdo a las exigencias del problema y a los resultados que se deseen evaluar por parte de los investigadores, convirtiéndose en enfoque versátil y flexible para enfrentar el estudio de grandes procesos industriales con inherente complejidad a un bajo costo en recursos físicos, económicos y talento humano.

2.3.2. Arquitectura de entornos HILS

La arquitectura de entornos HILS va ligada al proceso industrial bajo estudio sin dejar de lado los 3 componentes principales mencionados en el apartado 2.3.1, ver [33, 34].

En la arquitectura básica se encuentra un computador con un software de simulación numérica conectado a un equipo externo, que generalmente ejecuta algoritmos de control, por medio de herramientas de intercambio de datos (servidor OPC, tarjetas de adquisición de datos, protocolos de comunicación, etc) [4, 35]. En la figura 2.6 se puede observar la arquitectura de un entorno HILS, el cual fue concebido para el estudio del sistema de control de un reactor de presurizado de agua.

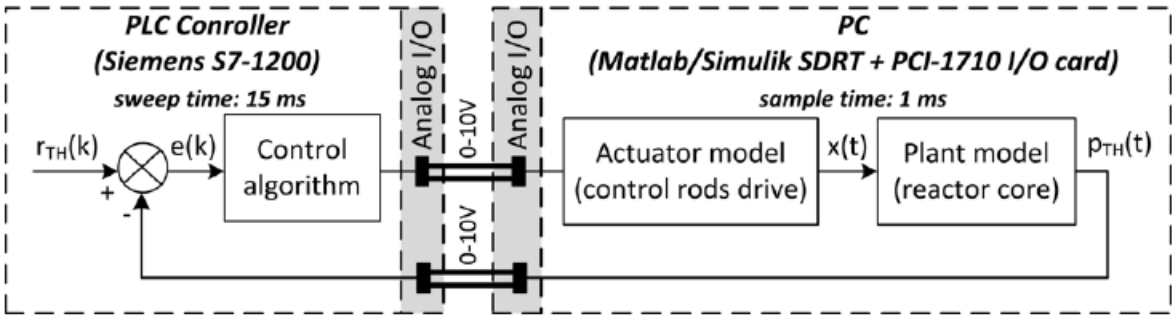

Figura 2.6: Entorno HILS para el sistema de control de un reactor de presurizado de agua. Tomado de [3]

Por otra parte, también se implementan arquitecturas HILS en las cuales se han añadido componentes tanto hardware y software adicionales a los 3 principales, para emular el comportamiento de un dispositivo específico. En la figura 2.7 se puede ver como dos motores son dispuestos para imitar el comportamiento de tuberías y motobomba. A su vez, en la figura 2.8 se aprecia la implementación virtual del sensor y actuador para el proceso de molienda de minerales delimitando los componentes del sistema de control (mundo real) y los componentes de simulación (mundo virtual).

En síntesis y de acuerdo a las arquitecturas vistas anteriormente, la arquitectura básica de un entorno HILS se puede observar en la figura 2.9. En ella destacan los 3 componentes principales que conforman dicho entorno.

2.3.3. Herramientas Hardware y Software para entornos HILS

Dentro de los entornos HILS no se definen herramientas hardware y software especiales para su construcción, encontrando variedad de dispositivos y programas informáticos inmersos en dicho entorno; de ahí su versatilidad y flexibilidad. Sin embargo, los programas de simulación y los dispositivos hardware usados en la implementación de un contexto HILS, poseen cualidades en común como, altas prestaciones en diseño y ejecución de modelos matemáticos, capacidad de intercambio de datos, posibilidad de adicionar más elementos conforme el problema bajo estudio lo requiera, entre otros; delegando al investigador la tarea de escoger los componentes hardware y software adecuados para las necesidades específicas del problema de investigación [5, 4, 3].

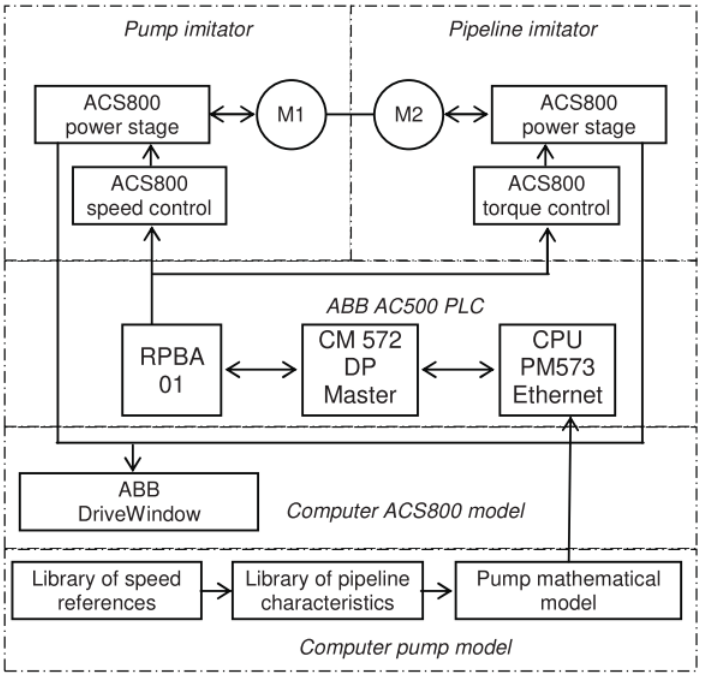

Figura 2.7: Arquitectura HILS para el estudio del sistema de control de una bomba centrífuga. Tomado de [4]

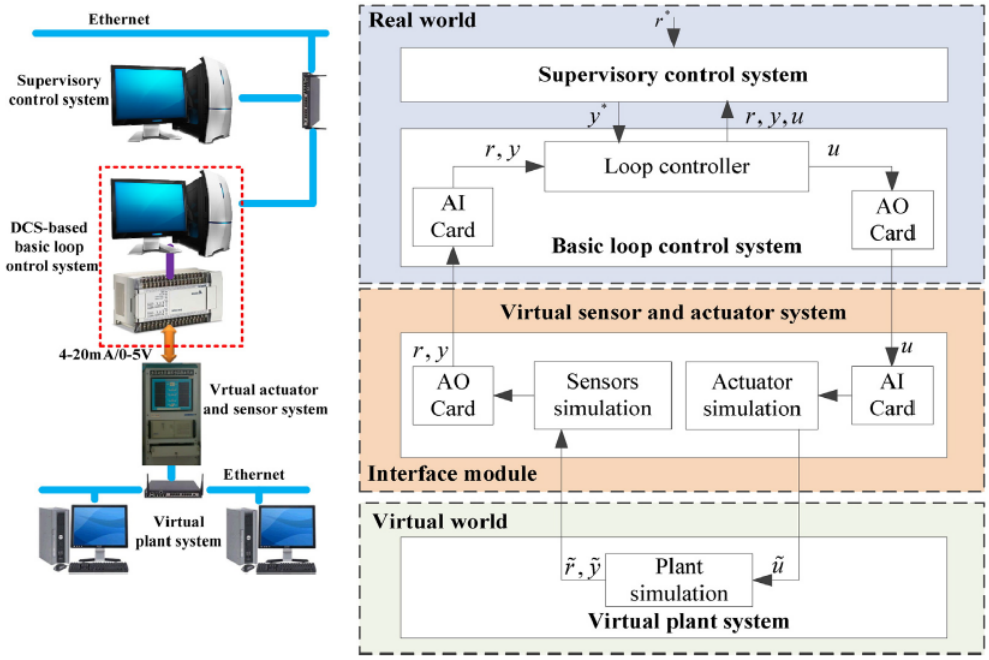

Figura 2.8: Arquitectura HILS con sensor y actuador virtual para el proceso de molienda de minerales. Adaptado de [5]

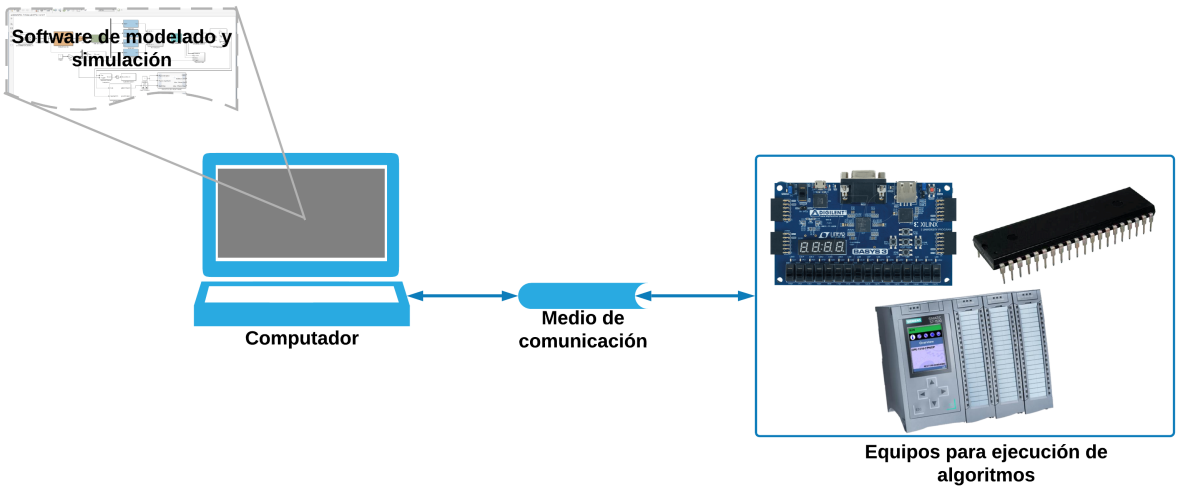

Figura 2.9: Arquitectura básica entorno HILS. Fuente: Propia

Entre las herramientas software más utilizadas para el diseño de algoritmos de control y simulación de modelos matemáticos podemos encontrar los siguientes paquetes: Matlab/Simulink, NI LabView. Cabe resaltar que el software Matlab/Simulink es ampliamente utilizado en el desarrollo de distintas investigaciones, entre ellas los sistemas de control, dada la gran cantidad de 'Toolboxes ' creados para facilitar el desarrollo y simulación de diversos algoritmos. En cuanto a herramientas de conexión e intercomunicación de los diferentes elementos que componen un entorno HILS para el estudio de procesos y sistemas de control industriales, existen estándares y protocolos como OPC (OLE for process control), ethernet, profibus, etc, asimismo, dispositivos como tarjetas de adquisición de datos, módulos de entradas/salidas analógicas y digitales, entre otros [34, 35, 36, 37].

Por otra parte, encontramos elementos hardware como PLC's (siemens, ABB, Rockwell), FPGA's, microcontroladores (Microchip, Atmel, etc), y dispositivos desarrollados específicamente para soportar algoritmos de alta complejidad como un controlador predictivo basado en modelo (MPC por sus siglas en inglés) [32].

2.3.4. Ventajas y desventajas en entornos HILS

Los entornos HILS presentan las siguientes ventajas:

- Bajo costo económico en su implementación.
- Escalabilidad tanto en hardware como en software.
- Realización ilimitada de pruebas con bajo riesgo para los equipos dispuestos en el lazo HILS.

Por otro lado, la mayor desventaja de un entorno HILS radica en el uso de modelos, principalmente matemáticos, para representar algún elemento del entorno, generalmente el proceso industrial bajo estudio. Lo anterior conlleva a perder cierto grado precisión en los resultados; sin embargo, la relación costo-beneficio de un entorno HILS es mayor a un entorno de simulación netamente virtual.

Capítulo 3

Implementación del entorno HILS incorporando un PLC industrial

El presente capitulo expone la selección y puesta en marcha de las herramientas software y dispositivos hardware que articularan el entorno HILS en el cual se llevara a cabo la implementación de un esquema de control jerárquico con optimización de consignas en línea. A partir de la literatura científica disponible en bases de datos, se propone la arquitectura del entorno, al igual que las herramientas software claves para su implementación; el proceso o planta industrial a controlar bajo el esquema anteriormente mencionado; el simulador para la construcción del modelo computacional del proceso seleccionado, y el PLC industrial a incorporar en el lazo HILS.

En la sección 3.1 se exhiben los trabajos y artículos de investigación que utilizan un entorno HILS con el objetivo de proponer la arquitectura del mismo; en la sección 3.1.4 se el simulador para su construcción; en la sección 3.1.5 se presenta una visión general del PLC industrial, las características del PLC seleccionado para este trabajo y las configuraciones pertinentes para su incorporación en el entorno; por último, en la sección 3.2 se describe la implementación del entorno.

3.1. Selección entorno HILS

Para proponer del entorno HILS a implementar en el presente trabajo, se realiza la correspondiente búsqueda sistemática, principalmente de trabajos y artículos en los cuales se incorpore un PLC industrial. No obstante, también se consultan trabajos adicionales en donde no se incluya un PLC en el lazo, dado que es importante observar los aportes en cuanto a los demás elementos que componen el entorno.

3.1.1. Búsqueda sistemática

La búsqueda sistemática es una herramienta que ayuda al investigador a identificar evidencia existente sobre un tema específico, vacíos en la investigación actual con el objeto de sugerir áreas para investigaciones futuras, y a su vez, provee un marco de trabajo y/o los antecedentes necesarios con el fin de posicionar nuevas actividades de investigación.

3.1.1.1. Criterios de búsqueda y selección de la información

En primer lugar, para realizar la búsqueda sistemática, se definen criterios de búsqueda y selección. Para el presente trabajo se establecen las siguientes cadenas de palabras para buscar información sobre entornos HILS.

- hardware in the loop AND control system
- hardware in the loop AND control system AND PLC

Por otra parte, se definen los siguientes criterios de selección para depurar la información encontrada.

- Artículos publicados en idiomas diferentes al inglés y español.
- Artículos con 10 años o más de antigüedad.
- Entorno de simulación diferente a HILS.

3.1.1.2. Información recolectada

La información encontrada a partir de los criterios de búsqueda y selección previamente establecidos en la sección 3.1.1.1 es consignada en las tablas 3.1 y 3.2 en las cuales se aprecian tópicos como el número de elementos que conforman el entorno, la complejidad de construcción y la información sobre el mismo. Dichos tópicos se definieron a partir de los elementos básicos que conforman el entorno HILS y se miden en básica, media y alta para los tópicos complejidad e información. Para llevar a cabo esta clasificación, se estudian los artículos previamente encontrados identificando los tópicos anteriormente mencionados.

3.1.1.3. Entornos de simulación encontrados

En la literatura revisada se destaca la aplicación de entornos HILS para el estudio de diversos procesos industriales y el uso de diferentes herramientas y dispositivos para su implementación. Asimismo, se hace hincapié en los algoritmos utilizados, principalmente algoritmos de control (PID, FOPID, MPC, etc), para el control del proceso y análisis del comportamiento general del sistema. Para el presente trabajo se realiza la clasificación de los entornos encontrados teniendo en cuenta los 3 componentes principales presentados en la sección 2.3.1.

Simuladores en entornos HILS:

Ítem	Título	Elementos que componen el entorno	Complejidad	Información al respecto	Ref
1	A Flexible Low Cost Embedded System for Model Predictive Control of Industrial Processes	3	Básica	Media	[32]
2	Application of IDEF Method in Hardware-in-the-loop Process Simulation	N/A	Básica	Básica	[38]
3	Design and Application of the Hardware-in-the-Loop Simulation System for Distillation Columns	3	Básica	Básica	[36]
4	Hardware-in-the-Loop Methods for Real-Time Frequency-Response Measurements fon-Board Power Distribution Systems	3	Alta	Básica	[37]

Tabla 3.1: Artículos encontrados con la cadena de búsqueda (hardware in the loop AND control system)

- Matlab/Simulink
- Aspen Hysys
- NI LabVIEW
- OPAL-RT simulator
- Dymola ThermoSysPro

Equipos para la ejecución de los algoritmos de control:

- PLC SIEMENS S7-1500, S7-1200, S7-400 y S7-300
- PLC HIMA HIMax y HIMatrix
- Dispositivos hardware para uso específico
- Computador industrial SIEMENS IPC427C
- PLC B&R X20
- PLC ABB AC500

Componentes del sistema de comunicación para el intercambio de datos en entornos HILS:

- Tarjetas de adquisición de datos DAQ (NIDAQ-6008, Advantech PCI-1710)

Ítem	Título	Elementos que componen el entorno	Complejidad	Información al respecto	Ref
1	PLC-Based Real-Time Realization of Flatness-Based Feedforward Control for Industrial Compression Systems	7	Alta	Media	[39]
2	PLC-Based Hardware-in-the-Loop Simulator of a Centrifugal Pump	6	Alta	Media	[4]
3	Implementation of the FOPID algorithm in the PLC controller-PWR thermal power control case study	3	Media	Alta	[3]
4	Fast Development of Controllers with Simulink Coder	3	Media	Alta	[33]
5	Hardware In The Loop (HIL) Simulation Of Wind Turbine Power Control	7	Media	Media	[34]
6	A Partially Automated HiL Test Environment for Model-Based Development Using Simulink and OPC Technology	3	Básica	Media	[40]
7	Real-Time Hardware-in-the-Loop Test Platform for Thermal Power Plant Control Systems	5	Media	Media	[6]
8	Hardware in the Loop Simulation of Railway Traffic Re-scheduling by Means of MILP Algorithm	8	Media	Media	[41]
9	Hardware and Software-in-the-loop Simulation for Parameterizing the Model and Control of Synchronous Condensers	8	Alta	Media	[7]
10	Application of OPC Protocol in Islanded Micro Grid Automation	5	Media	Media	[42]
11	Hydro power plant governor testing using hardware-in-the-loop simulation	4	Media	Alta	[43]
12	Using OPC technology to support the study of advanced process control	N/A	Alta	Alta	[44]

Tabla 3.2: Artículos encontrados con la cadena de búsqueda (hardware in the loop AND control system AND PLC)

- Servidor OPC
- Ethernet, MODBUS, Foundation Fieldbus

Cabe resaltar que la gran mayoría de los trabajos encontrados utilizan Matlab/simulink para el diseño de algoritmos, identificación de modelos matemáticos y ejecución de los mismos. En el mismo sentido, el software hace parte integral en la implementación de los entornos HILS dadas sus grandes prestaciones.

Cabe resaltar que la gran mayoría de los trabajos encontrados utilizan el software de cálculo numérico Matlab/simulink para el diseño de algoritmos, identificación de modelos matemáticos y ejecución de los mismos como se puede apreciar en la figura

3.1. En el mismo sentido, el software hace parte integral en la implementación de los entornos HILS dadas sus grandes prestaciones. De la misma forma, el PLC es ampliamente utilizado para la ejecución de los algoritmos de control y aplicación de la acción de control a los actuadores (reales y/o virtuales) como se puede observar en la figura 3.2. Por último, en la figura 3.3 se encuentra el porcentaje de utilización de los elementos clave utilizados para intercomunicar los entornos HILS. Destaca el uso de señales analógicas (voltaje y corriente) junto con interfaces para la adquisición, conversión y tratamiento de las mismas.

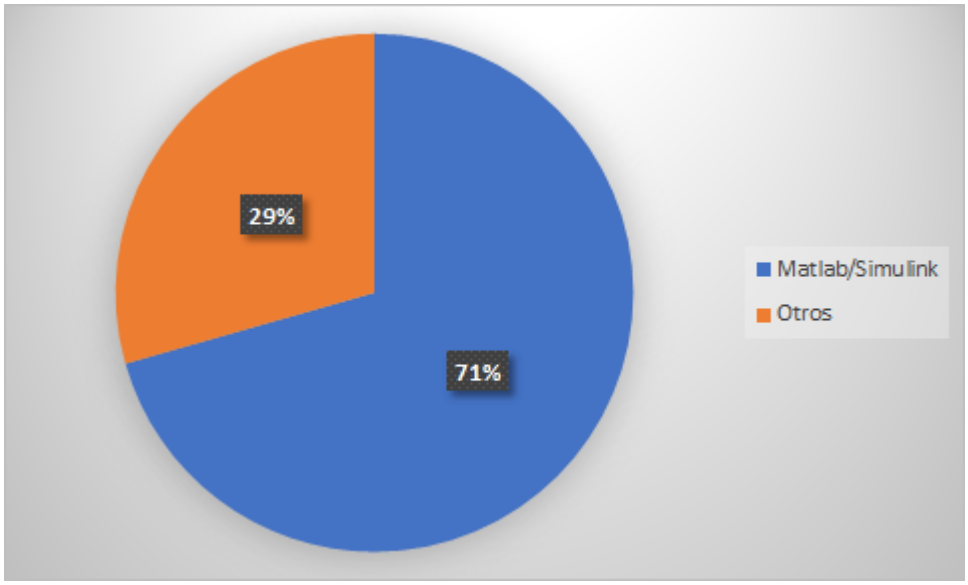

Figura 3.1: Porcentaje de utilización del software Matlab/Simulink frente a otros tipos de software similares

3.1.2. Arquitecturas HILS de mayor aplicación

TLas arquitecturas dentro de los entornos HILS encontrados, principalmente se diferencian en los elementos utilizados para intercomunicar los dispositivos que ejecutan el modelo matemático del proceso industrial y los dispositivos encargados de ejecutar los algoritmos de control. En la parte derecha de la figura 3.4 se observa la comunicación de los dispositivos que conforman el entorno HILS por medio de servidor OPC junto con protocolos de comunicación como industrial ethernet y profibus; en la parte izquierda de la misma, la comunicación se realiza a través de tarjetas DAQ.

Al utilizar señales analógicas (corriente y voltaje) para la comunicación entre los equipos que articulan el entorno HILS, se obtiene un mayor grado certeza en los resultados, dado que en un ambiente industrial se manejan este tipo de señales típicamente en un rango de 0 a 10 Volts o de 4 a 20 mA, aunque esto conlleva al incremento tanto en el número de elementos hardware como el aspecto económico. No obstante, utilizar software de tipo cliente/servidor para el intercambio de información entre los equipos,

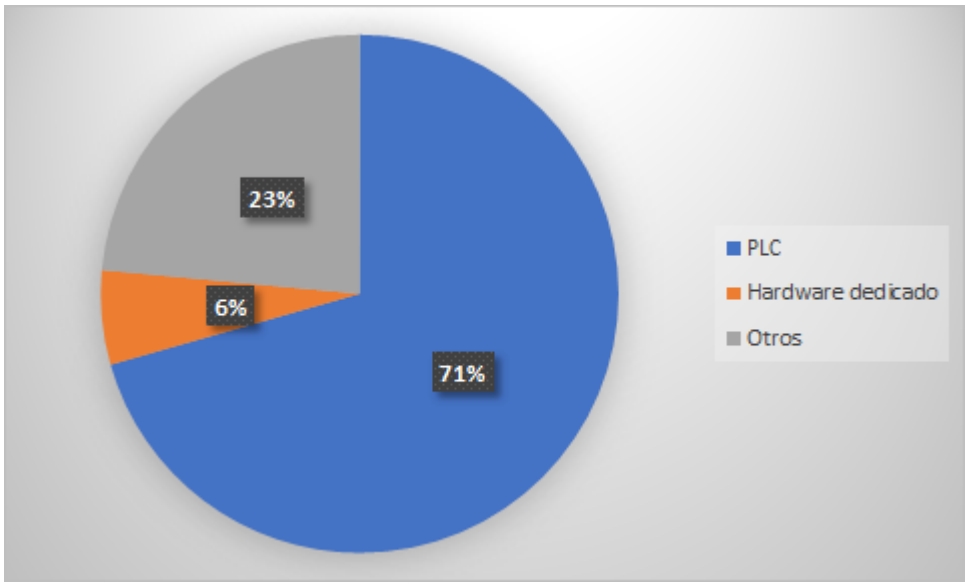

Figura 3.2: Porcentaje de utilización de equipos para la ejecución de algoritmos de control

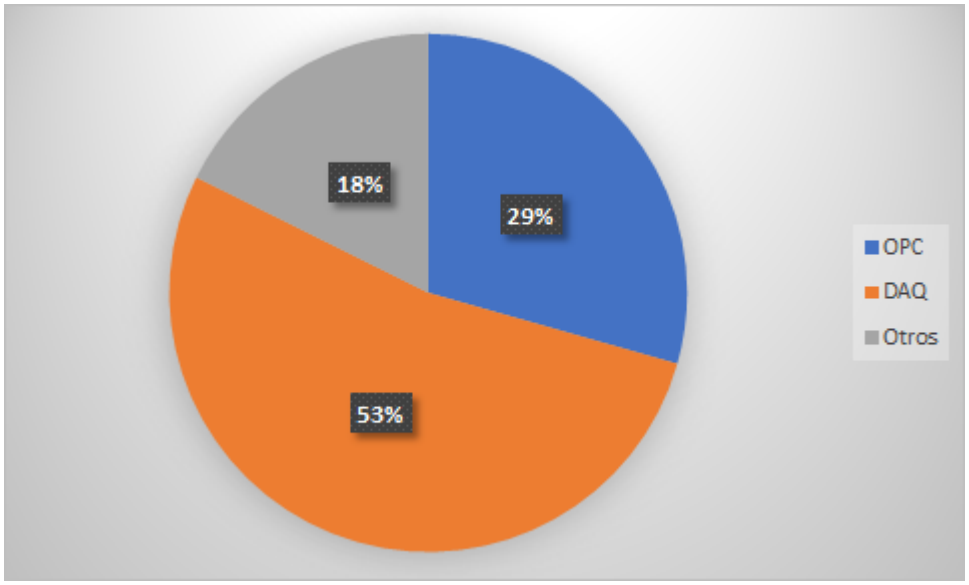

Figura 3.3: Porcentaje de utilización de componentes para el intercambio de datos en entornos HILS

es una manera rápida y económica de construir un entorno HILS para el desarrollo de cualquier investigación que contemple el uso del mismo.

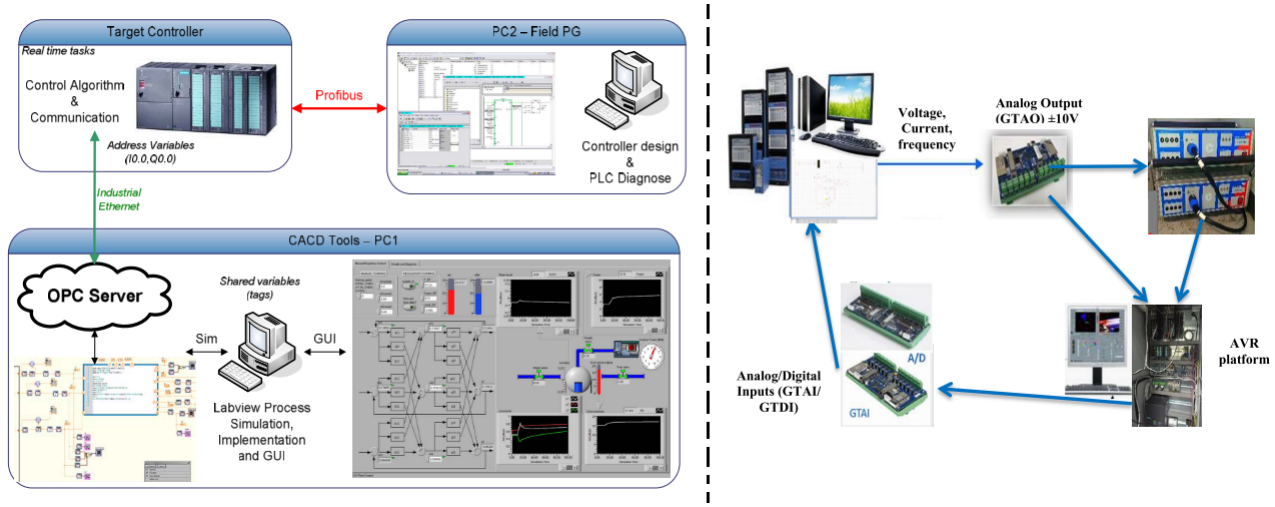

Figura 3.4: Arquitecturas HILS. Fuente: adaptado de [6], [7]

3.1.3. Propuesta arquitectura entorno HILS

Para plantear la arquitectura del entorno HILS, se toman en cuenta los siguientes aspectos, los cuales son sintetizados a partir de la literatura revisada:

- Disponibilidad de equipos para la ejecución de los algoritmos de control
- Software para el diseño de los algoritmos de control y modelado de procesos industriales
- Hardware y/o software para la comunicación entre los dispositivos que conformen el entorno HILS

Tomando en consideración los aspectos anteriormente planteados, para el presente trabajo se dispone de los siguientes elementos:

- PLC marca SIEMENS de la serie S7-1500 y S7-1200 en conjunto con su software de programación TIA PORTAL V14 SP1
- Matlab/Simulink
- Aspen Hysys
- Servidor OPC

En la figura 3.5 se aprecia la arquitectura propuesta con los elementos disponibles. El PLC se conecta físicamente a la tarjeta de red del computador a través de un cable RJ-45 y utiliza el protocolo de red Ethernet. Para el intercambio de datos entre Matlab/Simulink y el PLC se hace uso de un servidor OPC (estándar de comunicación comúnmente utilizado para el control y supervisión de procesos industriales), específicamente el software KepserverEx, el cual incluye los drivers necesarios para la lectura

y modificación de las variables dentro del PLC y demás programas conectados al mismo, en este caso, Matlab/Simulink. Por otra parte, el intercambio de información entre Aspen HYSYS y Matlab/Simulink se realiza vía ActiveX/COM gracias a la librería “HYCONNECT” creada por Olaf Trygve Berglihn. Es importante resaltar que el software Matlab/Simulink se utiliza tanto como pasarela de comunicación entre el simulador de procesos químicos Aspen HYSYS y el PLC, como para el diseño, sintonización y prueba de los algoritmos de control.

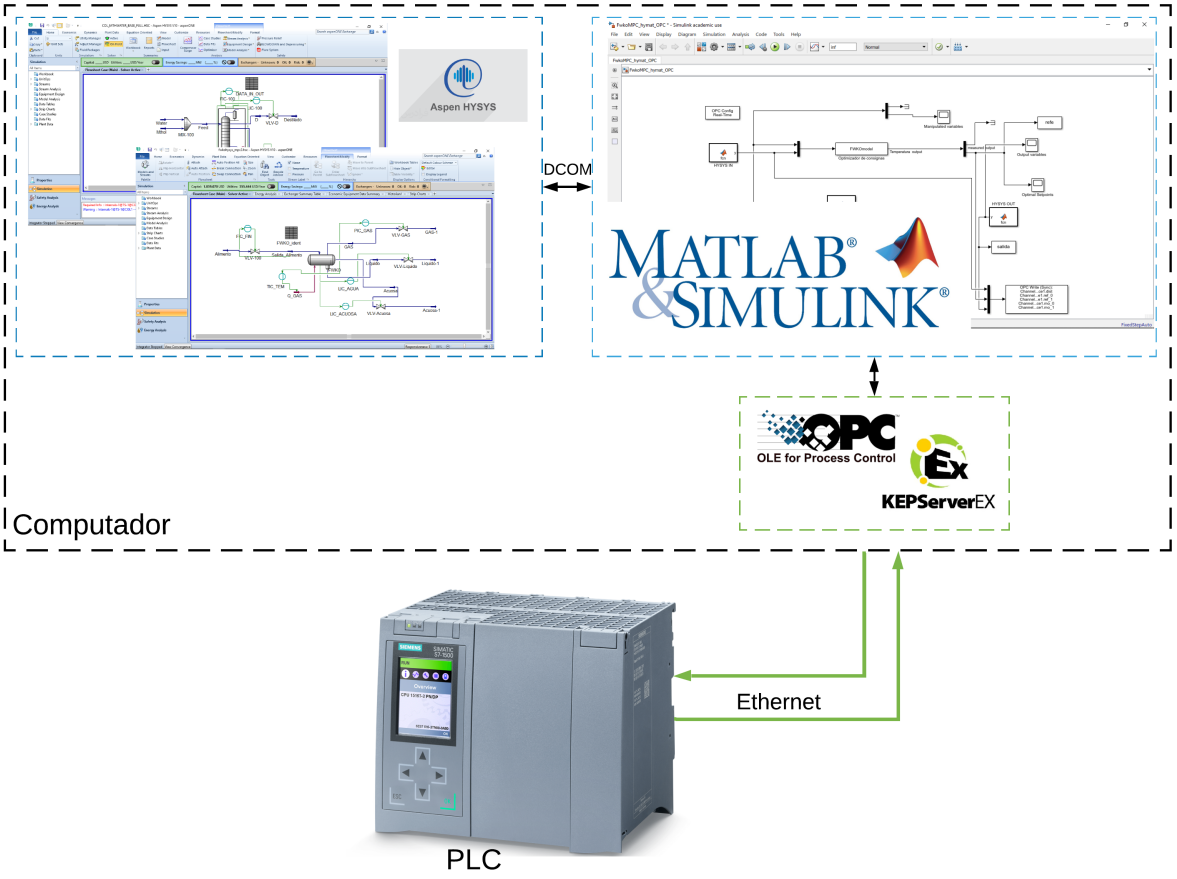

Figura 3.5: Arquitectura propuesta para el entorno HILS

3.1.4. Modelado de procesos industriales en Aspen HYSYS

Actualmente existen diversas herramientas software para el modelado y simulación de fenómenos físicos, económicos, y cualquier otro apto de ser abstraído e implementado en un instrumento computacional capaz de reproducir sus principales dinámicas. Lo anterior se realiza con el ánimo de evaluar el comportamiento de un fenómeno de especial interés en un ambiente controlado y modificable a gusto del investigador.

En el ámbito de la ingeniería, y en específico la ingeniería de control, cuenta con varias herramientas que proveen facilidades para la construcción de modelos de procesos industriales como los tratados en el capitulo V. Para la simulación de procesos químicos se encuentran varios tipos de software especializados en dicha materia como Aspen HYSYS, CADSIM Plus, UniSim, CHEMCAD, entre otros [45], los cuales incluyen objetos predefinidos (tanques, columnas, separadores, mezcladores, etc) para los que solo se necesita establecer parámetros de configuración (dimensiones del equipo, temperatura, presión, energía, etc) y paquetes que gobiernan el comportamiento de los fluidos químicos presentes en el modelo. Por otro lado, se tienen programas como Matlab/-Simulink, NI LabView, Scilab, etc, en donde el usuario crea a voluntad sus propios modelos y algoritmos.

La principal ventaja de utilizar algún tipo de software especializado, radica en la facilidad de crear el modelo que se desea estudiar a partir de los bloques o modelos de equipos disponibles en el mismo. No obstante, el usuario se ve restringido a los servicios que tenga el software, que en la mayoría de ocasiones no son escalables. Sin embargo, para el presente trabajo, las herramientas que ofrece la aplicación aspen HYSYS, especializada en modelamiento de procesos químicos, como lo son balances de masa y energía, cálculo de transferencia de calor, presión, relaciones vapor-líquido y comunicación con otras aplicaciones como Matlab/Simulink [12], son suficientes para abordar la construcción de los modelos de los procesos descritos en la sección 3.1.4.

3.1.4.1. Visión general Aspen HYSYS

Aspen HYSYS es un simulador de procesos químicos que utiliza modelos matemáticos de los mismos para llevar a cabo los cálculos y soluciones de las ecuaciones que los modelan. Este software fue creado en 1996 por investigadores canadienses de la Universidad de Calgary, quienes fundaron la compañía Hyprotech. En el año 2002 la empresa AspenTech adquiere la compañía Hyprotech, incluyendo el código fuente del programa HYSYS que continúa desarrollando hoy en día.

Aspen HYSYS dispone de una gran biblioteca para operaciones unitarias y varios paquetes llamados “Fluid Package” para el tratamiento analítico de la composición los fluidos que se añadan a la simulación. Por otra parte, este software presenta dos tipos de análisis para el estudio de los procesos modelados:

- Análisis en régimen permanente o estacionario
- Análisis dinámico

El análisis en régimen permanente o estacionario se realiza con el fin de introducir los principales parámetros de configuración a las unidades de operación seleccionadas (columnas, tanques, mezcladores, etc), definir la composición de cada fluido (fracción de agua, fracción de etileno, etc) y las conexiones entre los fluidos y las unidades.

Aspen HYSYS calculará los parámetros restantes a partir de la información introducida por el usuario, mostrando el comportamiento del sistema (magnitudes físicas como temperatura, presión, flujo, composición, entre otras).

Por otra parte, para el análisis dinámico es necesaria la introducción de equipos como motobombas, válvulas, controladores o intercambiadores de calor que generen diferentes dinámicas en los equipos y fluidos con el objetivo de observar los cambios de magnitudes físicas de interés a través del tiempo.

En las figuras 3.6 y 3.7 se puede apreciar la interfaz gráfica del software y el modelo de una columna de destilación realizado dentro del mismo.

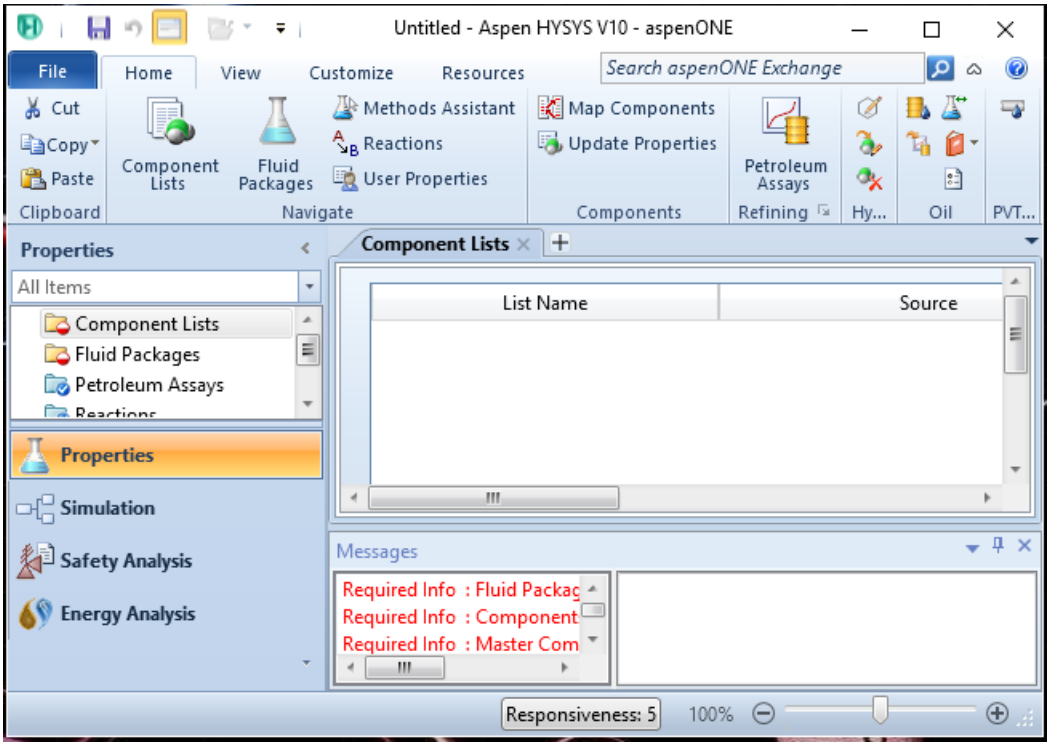

Figura 3.6: Interfaz gráfica Aspen HYSYS

3.1.5. PLC Industrial

Los PLC son dispositivos ampliamente utilizados en la industria para desempeñar acciones de monitoreo y control de procesos industriales. Estos dispositivos reemplazaron la lógica cableada dado que ofrecen programación flexible, implementación en espacios reducidos, sencilla detección de errores, escalabilidad supeditada a la complejidad del proceso a controlar, entre otras. El primer PLC, llamado 084, fue desarrollado a finales de la década de 1960 por Bedford Associates para la división Hydramatic de General Motors [REF].

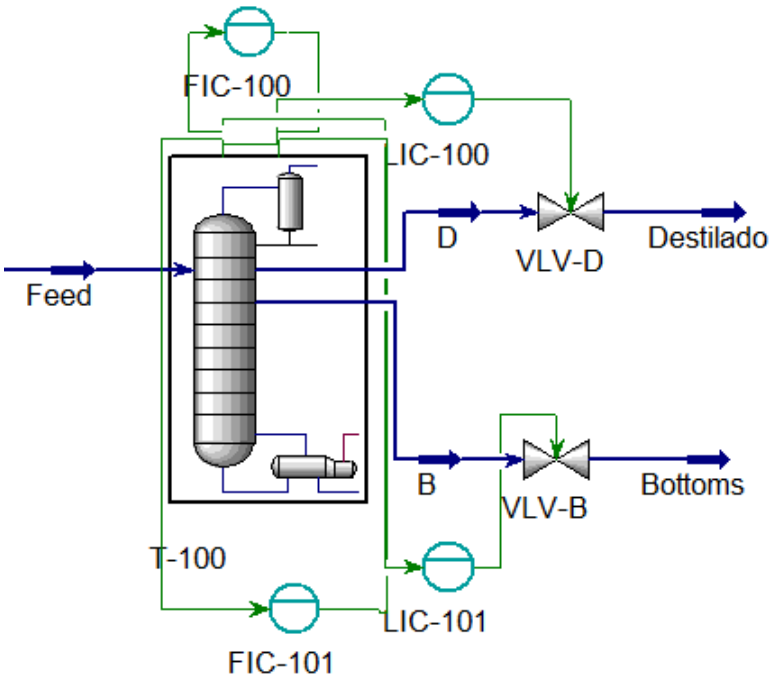

Figura 3.7: Columna de destilación modelada en Aspen HYSYS

- **Robustez:** Diseñados para operar continuamente y en condiciones ambientales hostiles.

- **Flexibilidad:** Es posible modificar la lógica de control y añadir módulos para la ejecución de otras tareas como comunicación de datos y visualización de información por medio de pantallas (HMI). Asimismo, disminuye los tiempos de desarrollo en proyectos de control y supervisión.

Por otro lado, la estructura interna de un PLC básicamente cuenta con interfaces de entrada/salida, unidad de procesamiento o CPU, memoria de almacenamiento de datos y estado de las entradas y salidas, fuente de poder, batería y bus interno para la transmisión de datos. En la figura 3.8 se puede observar el diagrama de bloques general de la estructura interna de un PLC.
Por otra parte, un aspecto importante dentro de los PLC es el denominado ciclo de scan, el cual es un proceso repetitivo donde se obtienen los estados de las entradas y salidas del PLC, los cuales son volcados a la memoria interna destinada para tal fin. Luego se ejecuta el programa descargado por el usuario y finalmente se actualiza el estado de las salidas. El tiempo que tarda el PLC en ejecutar un ciclo de scan, dependerá de la potencia de la CPU y la complejidad del programa que el usuario descargué al mismo. En la figura 3.9 se puede apreciar gráficamente el ciclo de scan.
Para el presente trabajo se cuenta con la disposición de dos PLC industriales de la marca SIEMENS, específicamente el PLC *S7-1200 1214C DC/DC/DC* y el PLC *S7-1500 1512C-1 PN*, los cuales se incorporan por separado en el entorno HILS propuesto

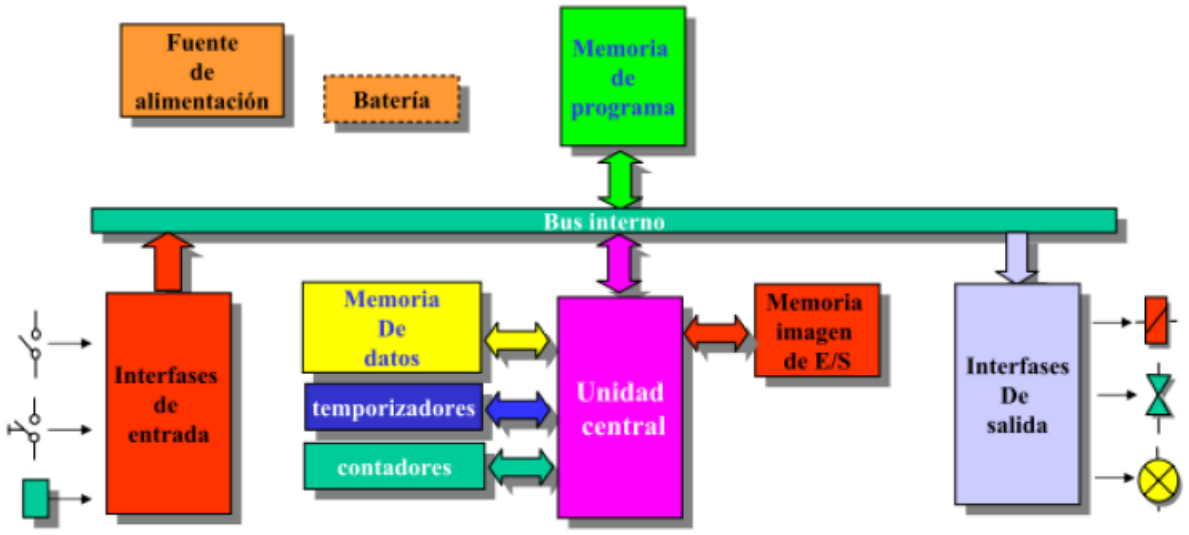

Figura 3.8: Diagrama de bloques estructura interna PLC. Fuente: Tomado de [REF]

en la sección 3.1.3 con el objetivo de evaluar el desempeño de los algoritmos de control que articulan la estructura presentada en la sección 2.1.1.1.

3.1.5.1. Características principales

Las principales características de los PLC a utilizar se pueden encontrar en la tabla 3.3.

Características / PLC	**S7-1200 1214C**	**S7-1500 1512C-1**
Memoria:		
Usuario	75 Kbyte	250 Kbyte
Trabajo	75 Kbyte	1 Mbyte
Carga	4 Mbyte	32 Gbyte (Max)
Tiempos de ejecución de operaciones:		
Bit	0.085 us/instrucción	48 ns/instrucción
Palabra	1.7 us/instrucción	58 ns/instrucción
Aritmética de coma fija	N/A	77 ns/instrucción
Aritmética de coma flotante	2.5 us/instrucción	307 ns/instrucción

Tabla 3.3: Principales características PLC

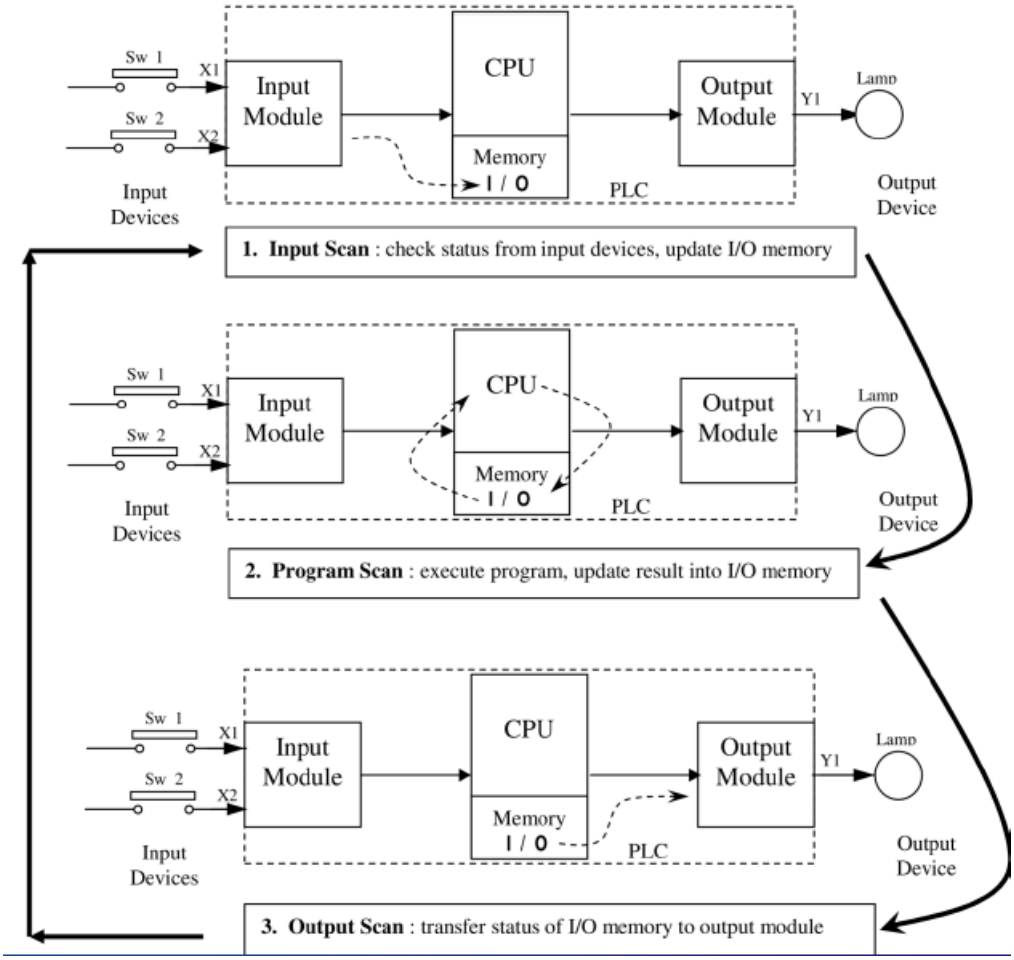

Figura 3.9: Diagrama de bloques estructura interna PLC. Fuente: Tomado de [REF]

3.2. Implementación entorno HILS

Tomando como referencia la arquitectura propuesta en la sección TAL se dispone a la conexión de cada componente (Aspen HYSYS, Matlab/Simulink, PLC SIEMENS).

En primer lugar, se realiza la conexión entre el PLC y Matlab/Simulink por medio del servidor OPC KepserverEx v6. Para lograr lo anterior, se toma como ejemplo el control de temperatura de la chaqueta de un tanque de agitación continua. El algoritmo de control, que este caso es un algoritmo MPC sin restricciones, se despliega en el PLC y la función de transferencia que modela el comportamiento de la chaqueta se programa en Matlab/Simulink. En la figura 3.10 se puede ver el bloque MPC dentro del entorno de programación para PLC SIEMENS TIA PORTAL V14 el cual cuenta con 3 entradas y 1 salida:

- **ssMethodType:** Variable de entrada que toma el valor de 0 o 1. Por defecto se inicializa en 0 para asignar valores a variables que serán utilizadas en la ejecución

del algoritmo. Cuando toma el valor de 1 se da paso a la ejecución del algoritmo MPC. Cabe resaltar que está variable es manipulada por el usuario.

- **ref:** Variable de entrada que indica el punto de ajuste o set-point deseado. El valor de esta variable es asignado por el usuario, que para este caso es la temperatura en grados centígrados se desea fijar en la chaqueta.
- **mo:** Variable de entrada que indica la magnitud de la salida del proceso, en este caso, la temperatura de la chaqueta.
- **mv:** Variable de salida que hace referencia al esfuerzo de control aplicado por el controlador a la entrada de la planta.

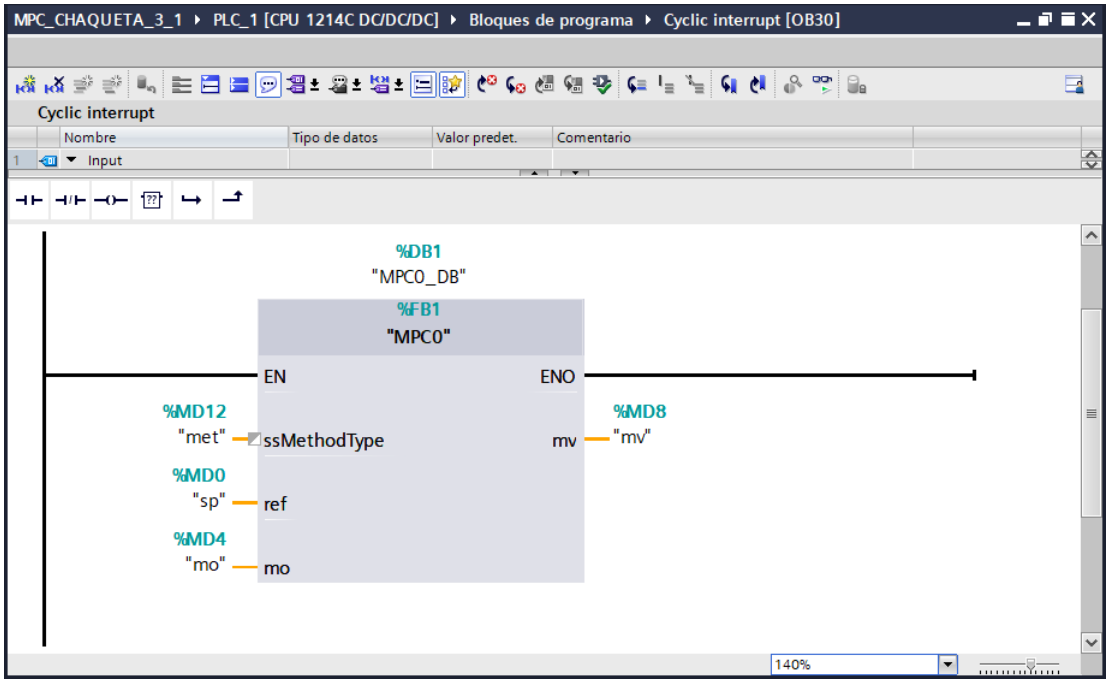

Figura 3.10: Bloque controlador MPC en TIA PORTAL V14. Fuente: Propia

El modelo y los parámetros de sintonización del controlador MPC son los siguientes:

```
%% MATRICES DEL SISTEMA
A               = [-0.8792    -0.0011          0
                    36.7077   -20.7578    20.8333
                          0   156.3445  -169.3055];

B               = [0;0;-1.7184];

C               = [0  1  0];

```

```
10 D             = 0;
11
12 %% TRANSFORMACION DE ESPACIO DE ESTADOS A FUNCION DE TRANSFERENCIA
13 [num,den]     = ss2tf(A,B,C,D,1);
14 sys           = tf(num,den);
15
16 %% PARAMETROS DEL MPC
17 Ts = 1;
18 p = 70;
19 m = 2;
20 mpcobj = mpc(sys, Ts, p, m);
```

Por otra parte, en la figura 3.11 se puede observar el bloque para la función de transferencia que representa la chaqueta junto con los bloques para lectura, escritura y configuración de la librería OPC presente en Matlab/Simulink.

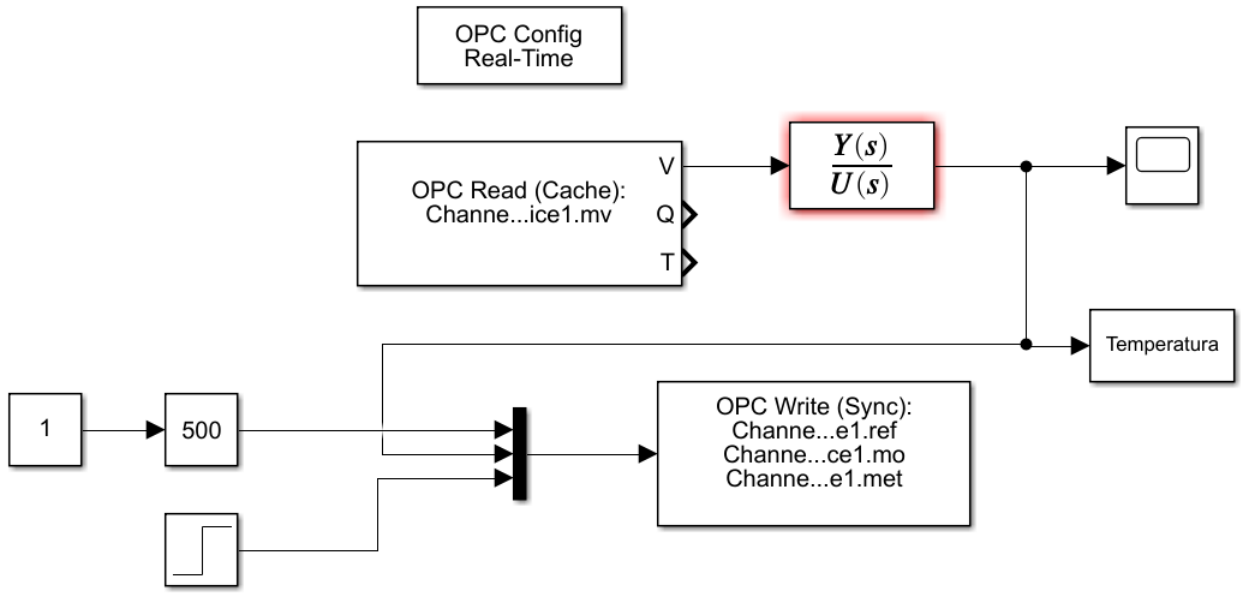

Figura 3.11: Modelo para simulación de la chaqueta de un tanque de agitación continua en Matlab/Simulink. Fuente: Propia

Para realizar la conexión entre el PLC y Matlab/Simulink se hace uso del servidor OPC KepserverEx V6 en el cual se crean las tags necesarias para lograr un control adecuado teniendo especial cuidado en colocar la misma dirección de memoria y tipo de dato de la variable creada en el PLC en el servidor OPC (Ver figura 3.12).
Cuando la conexión es satisfactoria, se procede a colocar en modo RUN tanto el PLC como el modelo realizado en Matlab/Simulink. En la figura 3.13 se puede apreciar la respuesta del controlador a diferentes consignas, las cuales fueron variadas en línea por medio del bloque “Slider Gain” dispuesto en la librería de Matlab/Simulink. (En el anexo A se detalla la configuración realizada)

Variables OPC

Tag Name	Address	Data Type	Scan Rate	Scaling
met	MD12	Float	100	None
mo	MD4	Float	100	None
mv	MD8	Float	100	None
ref	MD0	Float	100	None

Variables PLC

	Nombre	Tabla de variables	Tipo de datos	Dirección
1	sp	Tabla de variables e.	Real	%MD0
2	mo	Tabla de variables e.	Real	%MD4
3	mv	Tabla de variables e.	Real	%MD8
4	met	Tabla de variables e.	Real	%MD12

Figura 3.12: Tags en servidor OPC y en PLC. Fuente: Propia

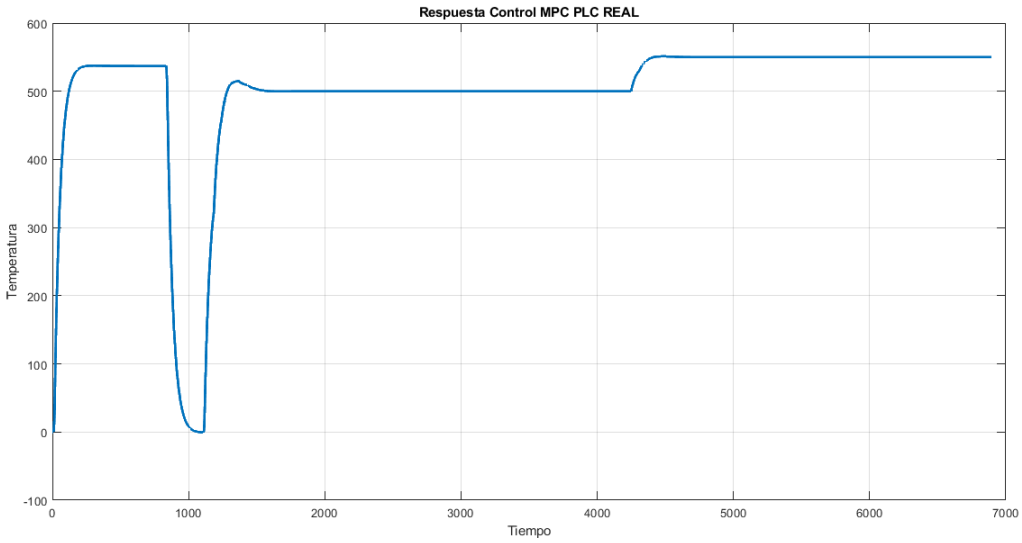

Figura 3.13: Respuesta controlador MPC temperatura chaqueta tanque de agitación continua. Fuente: Propia

Ahora se procede a conectar Aspen HYSYS y Matlab/Simulink. Para ello, se hace uso de la librería denominada “hyconnect” la cual trae todo lo necesario para realizar la conexión entre los dos programas. Dicha librería utiliza controles ActiveX para crear el intercambio de datos. Para esta parte se toma como ejemplo un modelo creado en Aspen HYSYS en donde se controla el nivel de un tanque por medio del porcentaje de apertura de una válvula que gobierna un controlador PID. El modelo creado en el simulador se puede observar en la figura 3.14 en el cual es importante destacar el objeto llamado “CONEX”, dado que en este se agregan las variables a modificar y leer desde Matlab/Simulink.

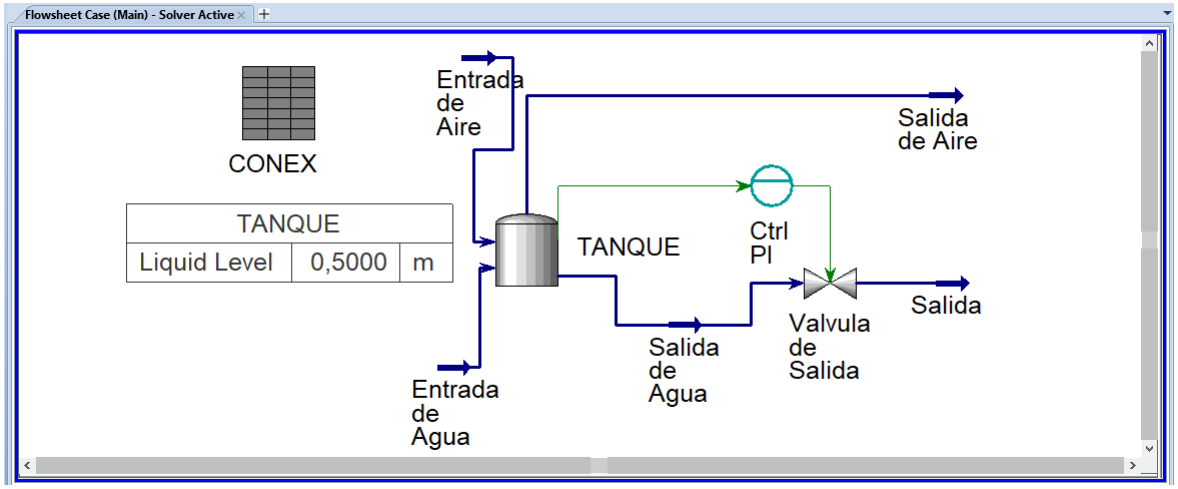

Figura 3.14: Modelo en Aspen HYSYS. Fuente: Propia

En la figura 3.15 se puede ver la configuración del objeto “CONEX” en donde se ha agregado la variable OP del controlador PI.

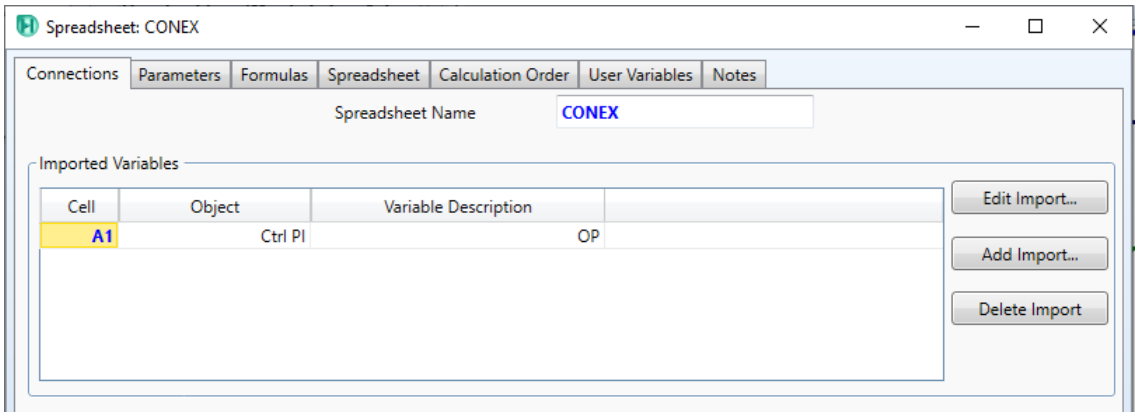

Figura 3.15: Configuración objeto CONEX. Fuente: Propia

Por otro lado, en Matlab/Simulink se crea un objeto a partir de la función “hyconnect” con el objetivo de obtener acceso a las variables agregadas en el objeto “CONEX”. Asimismo, con las funciones hycell y hyset se puede leer y escribir respectivamente desde Matlab/Simulink. En las siguientes líneas se pueden apreciar los comandos utilizados para generar la conexión de dichos software. (En el anexo A se detalla la configuración realizada)

```
1 %% Establecer la conexion con hysys
2 % la simulacion debe estar abierta
```

```
hysys = hyconnect;
sprd = hyspread(hysys,'CONEX');

% Recibe los valores de la celda A1 y los almacena
% en el espacio de trabajo
In = hycell(sprd,{'A1'});

% Constante de conversion de datos
Cns = 2.777777777777778e-04;

% Se fija el valor deseado en la celda de interes
hyset(In{1}, 90);
```

Como resultado final, fue posible conectar Aspen HYSYS, Matlab/Simulink y el PLC (tanto PLC S7-1200 y S7-1500) conformando el entorno HILS en el cual se evaluará el desempeño de algoritmos de control de una estructura jerárquica de dos fases con optimización de consignas en línea.

Capítulo 4

Implementación de la estructura de control jerárquico en el entorno HILS creado

Una vez que el entorno de simulación ha sido creado y conectado con éxito, es necesario evaluar la estrategia de control jerárquico de dos capas en este. Es decir, realizar un proceso de selección de las posibles formas de implementar los algoritmos requeridos en un PLC industrial. En este capítulo se desarrolla el procedimiento utilizado para encontrar las herramientas y metodologías más adecuadas que permitan hacer viable una implementación de una estructura de control jerárquico en un entorno HILS.

4.1. Estructura de control

Como fue mostrado en el capitulo 2, las estructuras de control jerárquico varían dependiendo del número de capas implementadas para realizar un control óptimo. A lo largo de la literatura y desde la primera aparición de una descomposición funcional de la tarea de control, se han incrementado las posibles combinaciones de algoritmos para llevar a cabo la denominada optimización en tiempo real en procesos de una complejidad considerable.

En primer lugar, la estructura más completa, abordada en el capítulo 2, presenta un número de 3 capas en total: la capa de optimización de régimen permanente LSSO, la capa MPC en dos fases, la cual está dividida a su vez en una subcapa de optimización dinámica SSTO y en el controlador predictivo MPC; y, por último, la capa de control básico PID.

Sin embargo, el actual trabajo tiene como premisa los resultados obtenidos del trabajo denominado “Diseño de controladores para la estrategia de control jerárquico en el problema de fijación de consignas óptimas en línea” desarrollado en la Universidad del Cauca para el programa de ingeniería en automática industrial, en el cual se realizó

el procedimiento de sintonización y desarrollo de una estrategia de control jerárquica para una columna de destilación, logrando realizar un control óptimo con la implementación de dos capas: la capa de optimización en régimen permanente LSSO y la capa MPC sin la subcapa de optimización dinámica SSTO. Por consiguiente, la búsqueda llevada a cabo será realizada para implementar esta estructura de control jerárquica en particular.

En la figura 4.1, se realiza una comparación entre la estructura más general de control jerárquico y la estructura que se busca implementar en el actual trabajo. Lo anterior se hace con el propósito de diferenciar la búsqueda y así poder desarrollar esta específicamente por cada capa, a la vez que cada una es asociada con la herramienta más adecuada. Cabe mencionar que para un entendimiento mayor de esta implementación para el caso de estudio de la columna de destilación, se remite al lector a la bibliografía correspondiente en [16].

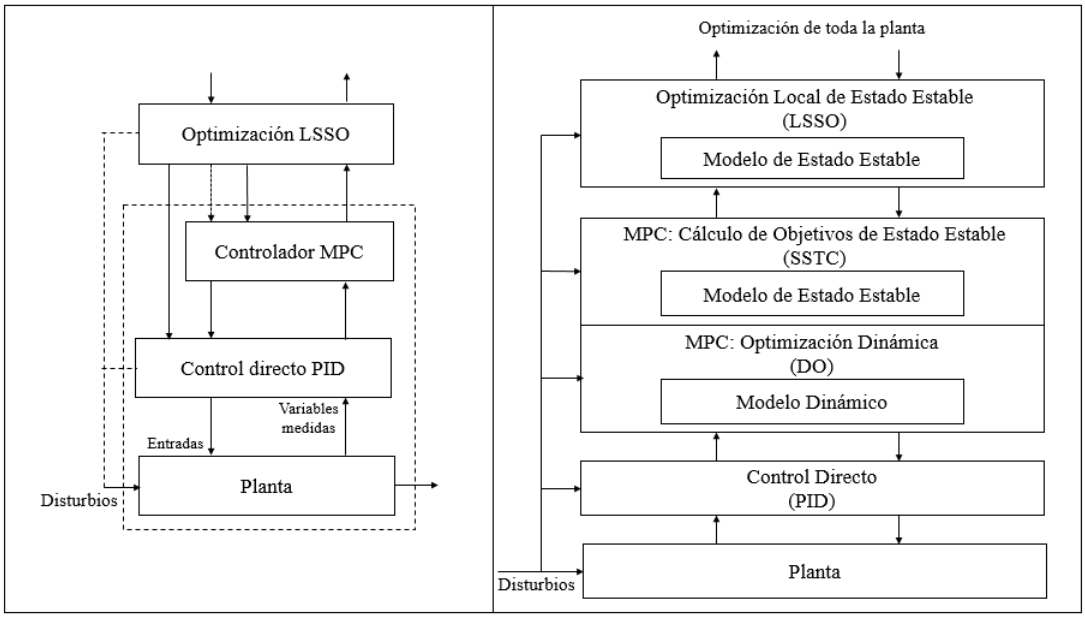

Figura 4.1: Comparación entre el esquema de control jerárquico general y el del caso de estudio

En la figura 4.1, el esquema del lado izquierdo resume, a grandes rasgos, la estrategia de control que se pretende implementar en el entorno HILS. A su vez, el esquema del lado derecho de la imagen permite evidenciar la capa que será omitida en el presente trabajo, a saber, la subcapa de cálculo de objetivos de régimen permanente SSTC. Dicha omisión es justificada por el autor en [12], al utilizar un modelo lineal de régimen permanente en la capa de optimización LSSO en lugar de uno no lineal. De esta forma, las diferencias en los tiempos de ejecución entre la capa de optimización y la capa de control MPC no generan mayor inconveniente, impidiendo así que se pierda optimalidad en los puntos de referencia entregados al controlador y finalmente permitiendo

el esquema de control jerárquico del lado izquierdo de la imagen, con las características requeridas para satisfacer esta técnica en el caso de estudio de la columna de destilación y, probablemente, para otros procesos y plantas a controlar óptimamente mediante esta estrategia.

Teniendo claro el esquema a implementar, se procede a analizar capa por capa las características de cada una de estas con el objetivo de hallar la mejor herramienta y técnica que permita hacer viable una implementación de dicho esquema en el entorno HILS creado, teniendo como prioridad la compatibilidad necesaria con un PLC industrial incluido en el lazo.

4.2. Implementación de una estrategia de control compleja en un PLC

Generalmente, los controladores lógicos programables son usados en la industria para solucionar una gran variedad de problemas de control, desde lazos de control de sistemas SISO hasta sistemas multivariables y algoritmos de supervisión más complejos. Para estos últimos, la codificación de las lógicas deseadas se convierte en una tarea compleja debido a la cantidad de parámetros a calcular y a supervisar. Este procedimiento, además de complejo, implica un riesgo considerable de daño en los equipos y un consumo de tiempo nada despreciable.

Debido a las razones anteriores, usar un entorno HILS que incluya un PLC en el lazo, en el que se pretende implementar la estrategia de control jerárquica de dos capas mostrada, requiere contemplar la metodología que mayor viabilidad aporte al objetivo de este trabajo, es decir, la que mejor se adapte a la implementación del esquema propuesto (o a parte de este), según las características y limitaciones de un PLC.

4.2.1. Generación de código

Las características consideradas más relevantes, para seleccionar el mejor procedimiento de generación de código, en la posible implementación de la estrategia de control jerárquica en un PLC son:

- Los lenguajes de programación soportados por el PLC
- El tiempo de desarrollo e implementación del algoritmo
- La flexibilidad, compatibilidad y estandarización de la técnica

A continuación, se muestran las opciones consideradas para la implementación de los algoritmos requeridos por la estrategia de control jerárquica.

4.2.1.1. Desarrollo manual de algoritmos de programación cuadrática

Como se explicó en el capitulo 2, debido a las características de los modelos utilizados para resolver el problema de optimización en la estrategia de control, las soluciones se hallan en problemas de programación cuadrática. Por esta razón se realizó una investigación de los algoritmos de programación cuadrática más usados en PLCs.

- Algoritmos Active Set (Algoritmos de Conjunto Activo)

Los métodos de conjunto activo se basan en el hecho de que, en un problema de optimización con restricciones, algunas de estas pueden ser ignoradas debido a que solo algunas se encuentras activas en el punto óptimo; ignorando, claro está, las que se encuentran inactivas en estos instantes de tiempo. Por lo tanto, es posible reducir la dimensión del problema, al descartar temporalmente las restricciones inactivas y manteniendo las que se encuentran activas [46].

Se resalta también, que al inicio del algoritmo se desconocen cuáles restricciones son activas y cuáles inactivas en los puntos de operación, por lo tanto, cada uno de estos métodos debe contar con una estrategia para escoger el conjunto de restricciones activas y cambiarlo por cada iteración [46].

- Métodos de punto interior (Interior Point Methods)

Los métodos de punto interior han sido aplicados con éxito a problemas de programación lineal desde su aparición en la década de 1984. Existen diversas implementaciones de carácter práctico, tales como el sistema LoQo (Linear Optimization, Quadratic Optimization).

También es capaz de resolver problemas de programación cuadrática sin importar el tamaño de este, logrando dichas soluciones en un rango de iteraciones entre 25 y 80. Este método se hizo conocido a partir del desarrollo del Karmarkar's interior-point method y es hasta ahora un área de investigación muy activa [46].

- Algoritmo Hildreth

Este algoritmo es uno de lo más usados para aplicaciones con control predictivo embebidos en PLCs. El algoritmo de Hildreth ha se caracteriza por su número limitado de líneas de código, lo que facilita su implementación. Es posible desarrollarlo en código C para ser embedido en un PAC o en S7-SCL para el PLC.

Este algoritmo calcula la solución en dos pasos. Primero, se calcula la solución sin restricciones y, si no se infringen restricciones, se adopta esta solución. Si se viola una restricción, se resuelve un QP restringido. La solución del QP se pasa luego a las entradas de la configuración. Si no se puede encontrar una solución para el QP, la solución no restringida se compara con la restricción. Si se viola una restricción, entonces esa entrada de la solución no restringida se limita a la restricción [47].

4.2.1.2. Generación basada en máquinas de estados

En la investigación de la generación de código para PLC, fue encontrada un método formal para la generación automática de programas para estos controladores industriales. El método comienza a partir del modelado del comportamiento deseado del sistema bajo diseño por medio de una máquina de estados con la capacidad de medir el tiempo y termina con un programa completo escrito en un lenguaje de diagrama de escalera. El lenguaje de destino (LD) ha sido escogido, para este método, por el uso generalizado de entornos de programación certificados ofrecidos por la mayoría de los fabricantes de PLC. A continuación se explican los pasos a seguir del algoritmo propuesto en [48].

1. *Codificación de estados.* En el primer paso, se asigna un grupo separado de flip-flops para codificar los sub-estados de cada superestado. Estos estados son el resultado de un análisis de entradas y salidas del proceso, por lo tanto, se les asigna un componente y una etiqueta de identificación. Es decir, que este paso define una semántica para la máquina de estados y para sus transiciones correspondientes.

2. *Implementar la función de temporizador.* Los temporizadores de la máquina de estados son asignados en el diagrama Ladder del código del PLC, manteniendo las condiciones de entrada y salidas de los estados definidos previamente.

3. *Implementar la función de transición.* Una secuencia de expresiones booleanas define las condiciones de set y reset de los flip-flops. Estas expresiones son dependientes de los temporizadores, las señales de entrada y los estados codificados en el primer paso.

4. *Rellenar el espacio de estado (recuperación de error).* Se definen los estados relacionados con las posibles entradas en fallo durante la ejecución del programa.

5. *Implementar la función de salida.* Las expresiones booleanas calculadas al final del ciclo del programa como resultado de los estados codificados son asignadas a las salidas físicas del controlador.

6. *Construir un programa.* Cada expresión booleana generada por los pasos anteriores define en su conjunto un programa detallado para PLC. Cada instrucción es convertida en una línea de código Ladder donde se asignan elementos lógicos a contactos normalmente cerrados o abiertos dependiendo de la lógica definida por los estados y sub-estados.

Para una mayor comprensión de este método de generación de código automático, y de su implementación en un caso de estudio, se remite al lector al trabajo de investigación realizado en [48].

4.2.1.3. Simulink PLC Coder

El toolbox de Matlab Simulink PLC Coder es una de las herramientas de generación de código para controladores industriales más usados por los investigadores en los últimos años. Esta técnica genera texto estructurado IEC 61131 independiente del hardware a partir de modelos de construidos en Simulink, cuadros de Stateflow y funciones Matlab embebidas.

El texto estructurado se genera en PLCopen XML y otros formatos de archivo compatibles con los entornos de desarrollo integrado (IDE) ampliamente utilizados. Como resultado, puede compilar e implementar su aplicación en numerosos dispositivos de controlador lógico programable (PLC) y controlador de automatización programable (PAC). Simulink PLC Coder genera bancos de prueba que lo ayudan a verificar el texto estructurado utilizando las IDE de PLC y PAC y las herramientas de simulación.
Algunas de sus características clave son:

- Generación automática de texto estructurado IEC 61131-3
- Compatibilidad con Simulink, incluidos subsistemas reutilizables, bloques de controladores PID y tablas de búsqueda
- Soporte de Stateflow, incluidas funciones gráficas, tablas de verdad y máquinas de estado
- Compatibilidad con Matlab incrustado, incluidas sentencias if-else, construcciones de bucle y operaciones matemáticas
- Compatibilidad con múltiples tipos de datos, incluidos booleanos, enteros, enumerados y de punto flotante, así como vectores, matrices, buses y parámetros ajustables
- Soporte IDE, incluyendo Automation Studio, PLC open XML, Rockwell Automation RSlogix 5000, Siemens Simatic Step 7 y 3S-Smart Software Solutions CoDeSys
- Creación de banco de pruebas

4.3. Implementación del esquema de control jerárquico por capas

Con el objetivo de determinar la mejor estrategia posible para la implementación del esquema de control de dos capas, se resume la información determinante para el actual trabajo en la siguiente tabla 4.1.
Con la información sintetizada, se optó por realizar la generación de código para el PLC con la herramienta PLC Coder de Simulink/Matlab. El primer criterio de elección

Método	Lenguajes soportados	Tiempo de implementación
Algoritmo de conjunto activo	Texto estructurado, C.	Días
Algoritmo de punto interno	Texto estructurado, C.	Días
Algoritmo Hildreth	Texto estructurado, C.	Días
Máquina de estados	Ladder, C.	Horas
Simulink PLC Coder	Texto Estructurado C, C++	Minutos

Tabla 4.1: Características relevantes de las técnicas de generación de código

se fundamenta en la eficiencia del lenguaje de programación para desarrollar una técnica de control avanzada como lo es, en principio, el controlador MPC, es decir, que el texto estructurado es mucho más eficiente para tareas que impliquen cálculos matemáticos más complejos gracias a su naturaleza escrita (a diferencia del espacio que requiere el Ladder y a su dificultad para la comprensión de la misma lógica de control).

En segundo lugar, se ha resaltado la complejidad de sintonización y puesta en marcha de una técnica como el control basado en modelo MPC, por lo que el factor “tiempo” es determinante en la agilización de los pasos requeridos para la implementación del esquema estudiado y, por consiguiente, otorga mayor viabilidad al proyecto, al menos, en su etapa de desarrollo.

Finalmente, es evidente que las características soportadas por esta herramienta imposibilitan la implementación total de las capas superiores de la estrategia de control jerárquica. Debido a esto, el PLC Coder solo servirá como herramienta de implementación de la capa MPC. Sin embargo, una solución factible es desarrollar la capa de optimización LSSO restante en el mismo software Matlab, ya que este se encuentra incluido en el entorno HILS creado, y tendrá la capacidad de trabajar en conjunto con el PLC y el software de simulación Aspen Hysys.

De esta forma, la implementación del esquema de control jerárquico de dos capas se realiza como se muestra en la figura 4.2, donde a cada capa le corresponde un software de desarrollo y generación de los algoritmos; y un software o hardware donde estos son implementados.

Para la generación de código se sigue la metodología expuesta en la figura 4.3, en donde se crean y/o sintonizan los algoritmos de control en Matlab/Simulink, luego se emplea la herramienta PLC Coder para generar código en texto estructurado, compatible con el PLC. Posteriormente, el código generado es trasladado a la plataforma de programación del PLC, desplegado en el mismo y finalmente llevado al entorno HILS.

Con la arquitectura y las interfaces más adecuadas para la implementación del esquema de control definidas, en el siguiente capítulo se presentan los resultados de

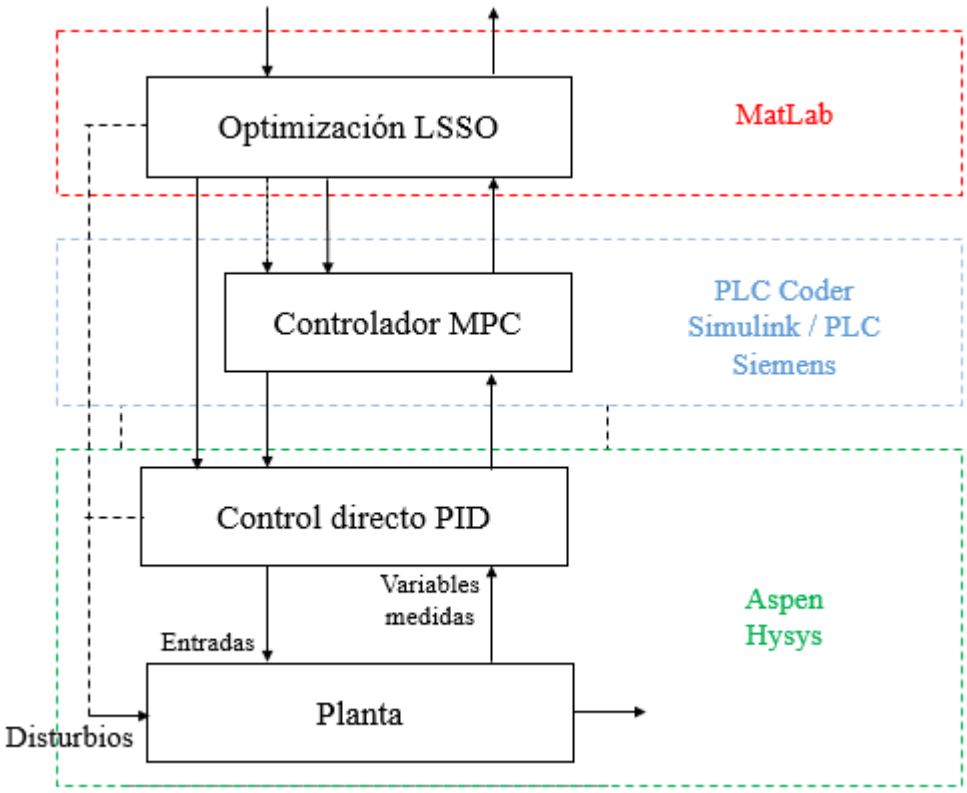

Figura 4.2: Implementación del esquema de control de dos capas junto con las herramientas seleccionadas. Fuente: Propia

desempeño entregados por el controlador en la simulación de tiempo real para los casos de estudio.

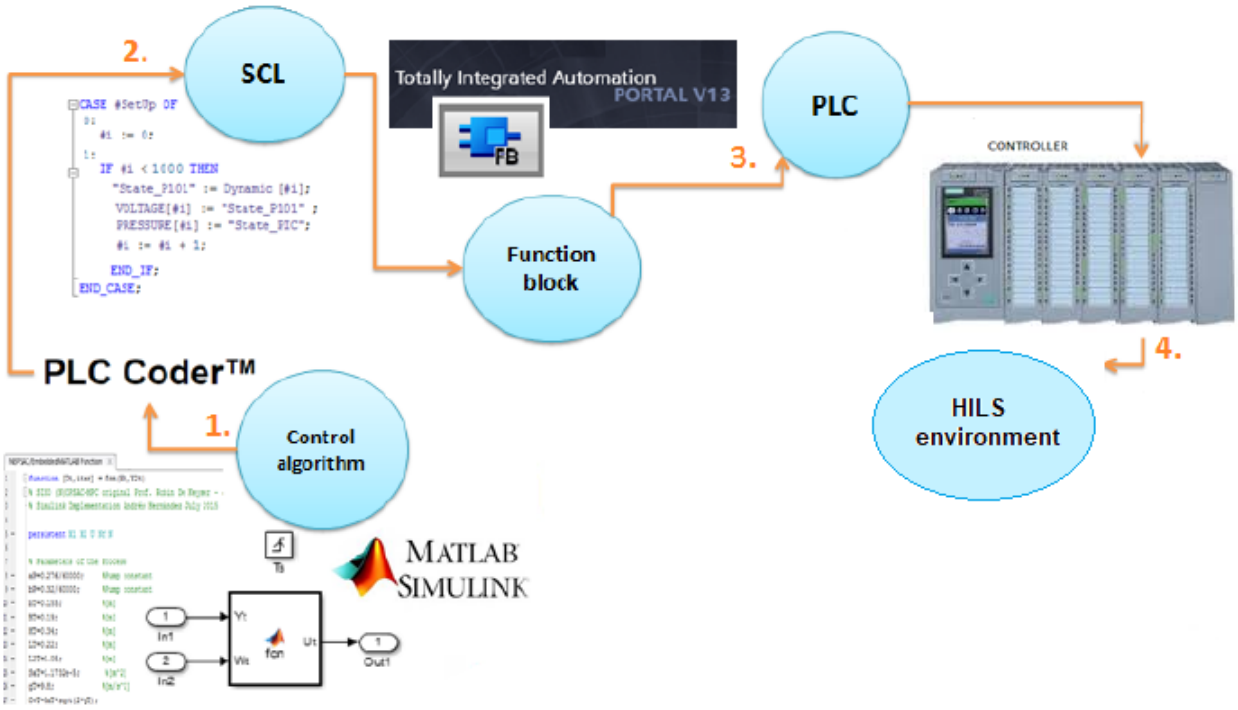

Figura 4.3: Metodología generación y despliegue de código en el PLC. Fuente: adaptado de [8]

Capítulo 5

Simulación y desempeño de la estructura de control

Con el objetivo de evaluar la metodología de implementación propuesta y su correcto funcionamiento, se presentan a continuación dos procesos diferentes en los que se será ejecutada la estructura de control de dos capas con el controlador predictivo corriendo en un PLC Siemens de la serie S7-1500 y la capa de optimización ejecutándose en Matlab como se había determinado en la sección anterior.

Para los casos de estudio se presentan, en primer lugar, sus características generales como variables manipuladas, objetivos de control y esquema de la planta. En segundo lugar se exponen los procedimientos de identificación, sintonización, generación de controlador y ejecución de la simulación en el entorno HILS creado. Finalmente se contrastan los resultados de simulación en tiempo real para el desempeño de control óptimo en cada caso de estudio con respecto a los resultados fuera de linea esperados.

Según Muñoz en [12], Zambrano y Alegría en [49], algunos de los procesos a los cuales se aplica la estrategia de control jerárquica con optimización de consignas en línea, dada la complejidad y variables de proceso a controlar, son los siguientes:

- Tanque-Reactor de agitación continua (CSTR por su sigla en inglés)
- Columna de destilación
- Aspen Hysys
- Separador trifásico de un tren de tratamiento de crudo (FWKO por su sigla en inglés)

A continuación, se da una breve descripción de cada uno de los procesos anteriormente mencionados.

Tanque-Reactor de agitación continua: En este caso, una serie de reactivos son vertidos al tanque-reactor para su continua agitación, presentándose una reacción exotérmica reversible. Las principales variables a analizar y controlar en este proceso son

concentración, nivel, presión, temperatura, flujo molar y peso molecular. En la figura 5.1 se puede observar un modelo básico del tanque-reactor de agitación continua.

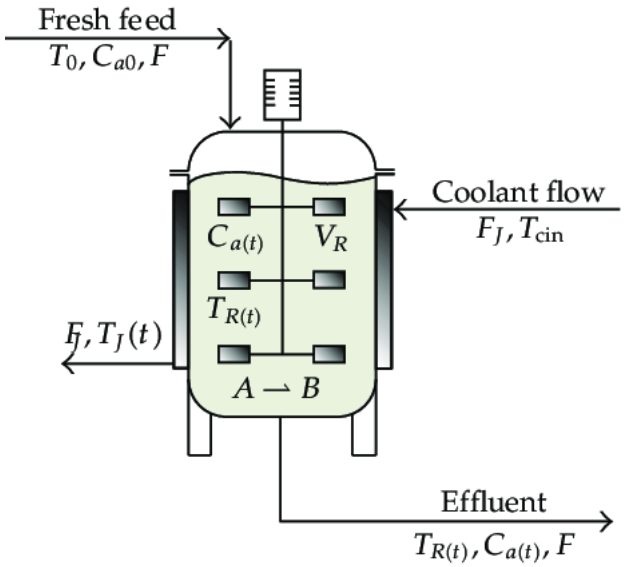

Figura 5.1: Esquema tanque-reactor de agitación continua. Fuente: Tomado de [9]

Columna de destilación: Este proceso tiene como objetivo separar los diferentes elementos líquidos y sólidos presentes en una mezcla por medio de vaporización (aplicando calor) y condensación (extrayendo calor). Las principales variables a estudiar y controlar son nivel, flujo, presión, temperatura, concentración, corrientes de refrigeración y corrientes caloríficas. En la figura 5.2 se puede observar un bosquejo básico de una columna de destilación.

Separador trifásico de un tren de tratamiento de crudo: Este proceso se aplica al crudo proveniente de los pozos petroleros con el fin de separar las fases existentes en él (agua, petróleo y gas). En algunas ocasiones, los separadores trifásicos poseen un tratador de calor para posibilitar el debilitamiento de la coalescencia entre las gotas de petróleo y agua, absorbiendo la energía producida por algún tipo de combustible. En la figura 5.3 se puede ver el esquema simplificado de un separador trifásico.

De los procesos anteriormente descritos, se toman como posibles casos de estudio la columna de destilación y el separador trifásico, considerando que existe información suficiente (parámetros físico-químicos) para el modelado de los mismos en el software Aspen HYSYS. De igual forma, las variables a manipular y controlar permiten la aplicación de una estrategia de control jerárquico de dos fases. Por otra parte, es importante mencionar que si bien el proceso llevado a cabo en un tanque-reactor de agitación continua califica para la aplicación de una estructura de control jerárquica, el software de simulación Aspen HYSYS no contiene los paquetes de simulación para polímeros, los cuales son fundamentales para el proceso [12].

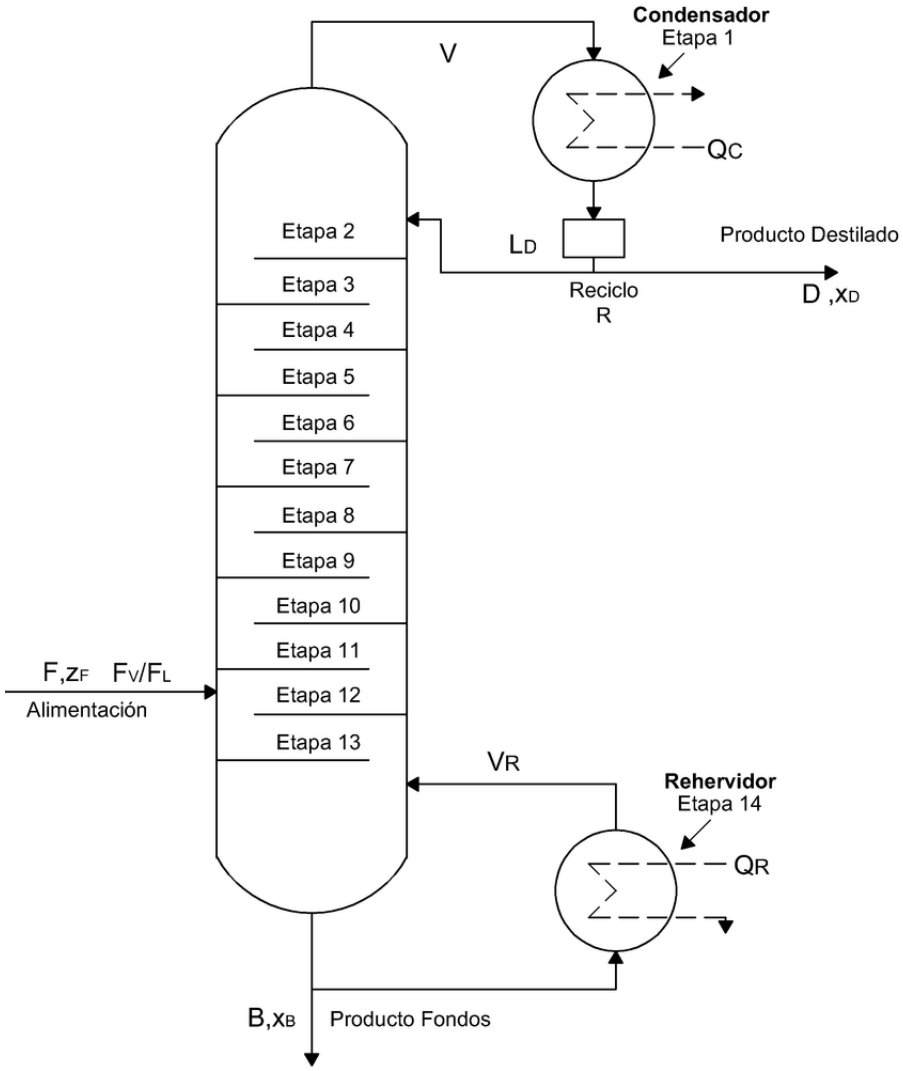

Figura 5.2: Esquema columna de destilación. Fuente: Tomado de [10]

5.1. Desempeño del controlador en la estructura de control para la columna de destilación

En la presente sección se describen los objetivos de optimización y control para los casos de estudio columna de destilación metanol-agua y separador trifásico en un tren de tratamiento de crudo, con el objetivo de apreciar el desempeño de la estructura de control jerárquica en un entorno HILS, en donde la capa de control avanzado se ejecuta en un PLC industrial.

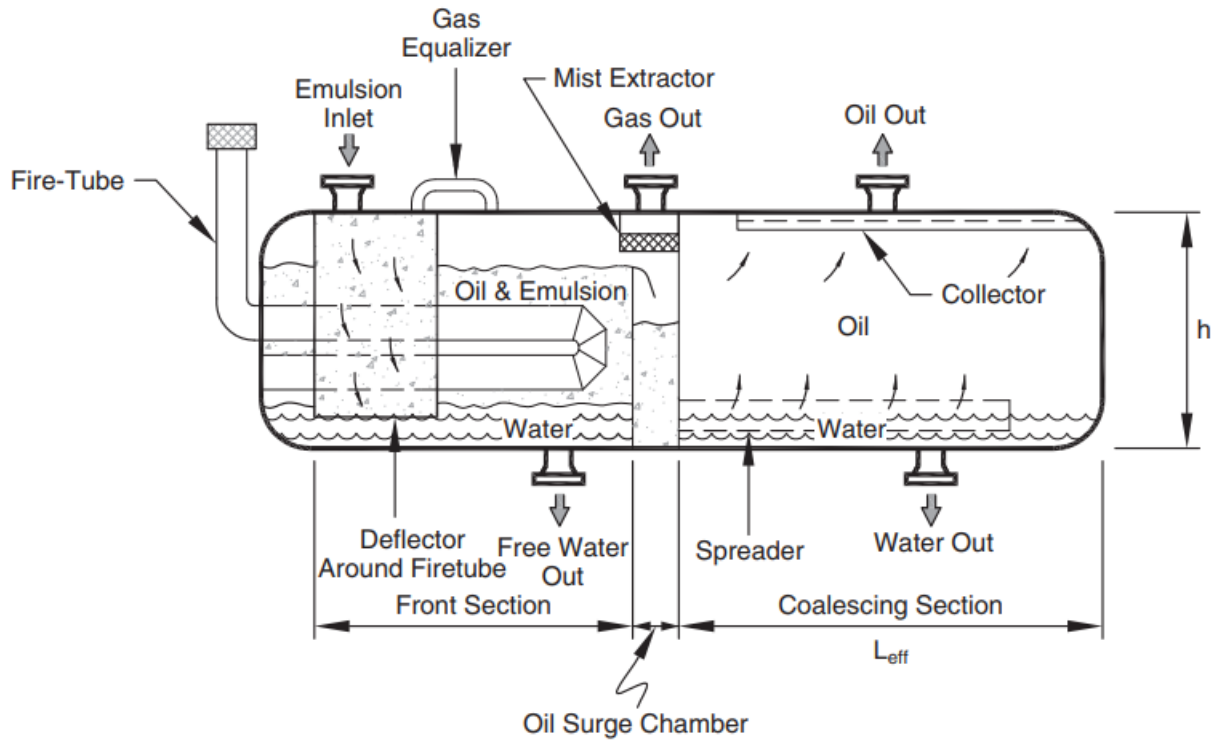

Figura 5.3: Esquema simplificado separador trifásico. Fuente: Tomado de [11]

5.1.1. Objetivo de control columna de destilación

El objetivo de control planteado para una columna de destilación metanol-agua en [12] es controlar la composición de metanol en el tope y el fondo de la columna, manipulando los flujos molares de reflujo Flow_R y vapor Flow_V. En la figura 5.4 se puede apreciar los controladores de flujo, y las entradas y salidas de la columna. Cabe resaltar que la columna de destilación se encuentra modelada en el software de simulación de procesos químicos Aspen HYSYS.

Dado que este proceso se pretende controlar bajo una estructura de control jerárquica, se debe identificar un modelo del mismo, el cual servirá para el diseño del problema de optimización y el control MPC.

5.1.2. Identificación del modelo para la columna de destilación

Para la identificación del modelo se realiza la inyección de señales binarias pseudo-aleatorias o PRBS en las entradas de la columna de destilación de metanol-agua, a saber, flujo de reflujo, flujo de vapor, flujo molar y composición de metanol en el flujo molar. Asimismo, se obtiene la magnitud de la composición de metanol en el tope y en el fondo, los cuales varían de acuerdo al comportamiento de las señales de entrada.

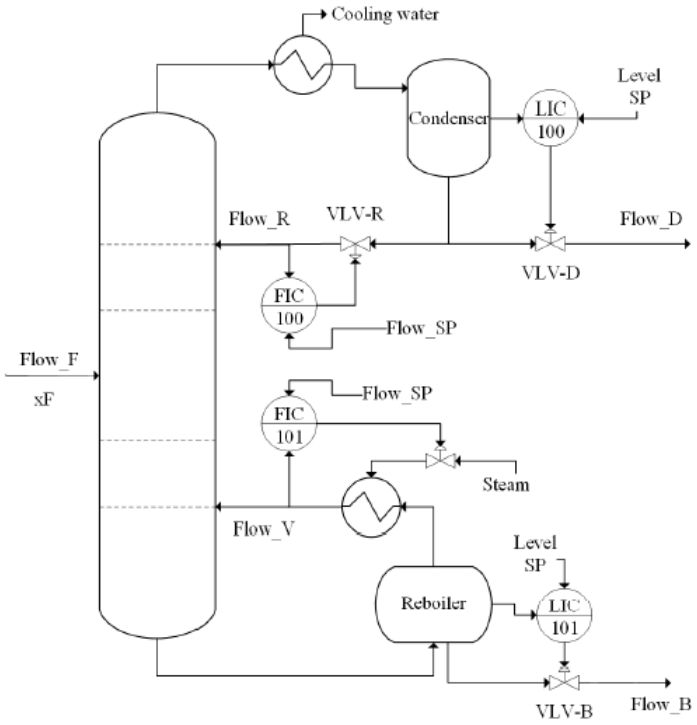

Figura 5.4: Controladores para columna de destilación. Fuente: Tomado de [12]

Lo anterior se ejecuta comunicando Matlab/Simulink y Aspen HYSYS como se explica en el capítulo III. En la figura 5.5 se puede observar el esquema propuesto para la identificación del modelo [12].

El modelo lineal identificado es el descrito en las matrices 5.1:

$$\mathbf{A} = \begin{bmatrix} 0.77050 & 0.05148 & -0.01572 & -0.01811 & -0.01479 & 0.0106 \\ -0.47410 & 0.67360 & -0.15860 & 0.17210 & 0.02090 & 0.17230 \\ -0.13250 & 0.14490 & 0.11640 & -0.03624 & 0.07266 & 0.16250 \\ 0.18310 & 0.29860 & -0.62560 & 0.63180 & -0.09473 & 0.13650 \\ -0.01256 & 0.37680 & 0.29120 & -0.41440 & 0.46460 & 0.30610 \\ -0.00809 & -0.01087 & 0.10200 & 0.02984 & -0.02390 & 0.99940 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0.004814 & -0.003201 & 0.015470 & 0.966500 \\ 0.012010 & -0.012160 & 0.029660 & 0.867500 \\ 0.065920 & -0.009914 & 0.299800 & 0.097320 \\ 0.049210 & -0.004593 & 0.313800 & 3.794000 \\ -0.026500 & -0.000530 & 0.066310 & 12.75000 \\ -0.009080 & 0.000980 & -0.039660 & 0.244300 \end{bmatrix} \tag{5.1}$$

$$\mathbf{C} = \begin{bmatrix} 0.283700 & -0.084530 & 0.027980 & -0.022840 & -0.002035 & 0.035890 \\ 0.812700 & -0.011600 & -0.044060 & 0.005270 & -0.005246 & -0.116300 \end{bmatrix}$$

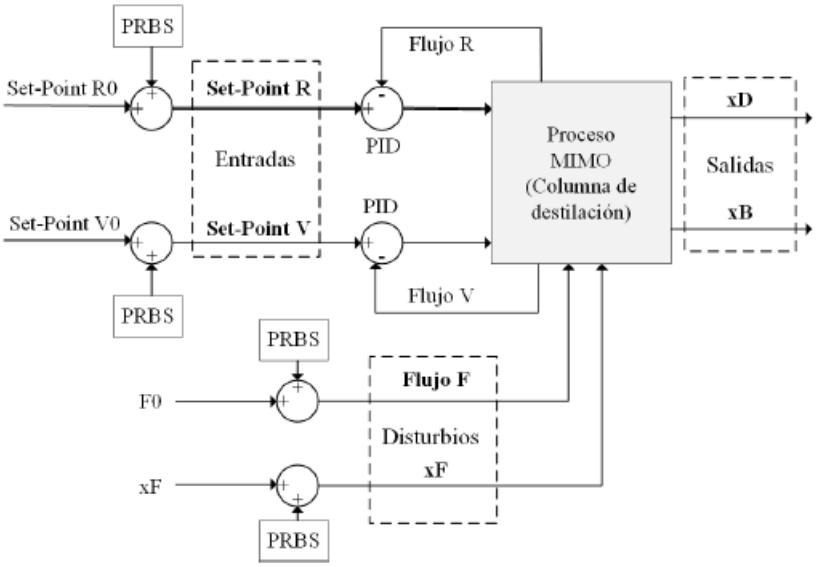

Figura 5.5: Esquema para identificación columna de destilación. Fuente: Tomado de [12]

Con el modelo identificado se procede a diseñar la capa de optimización y la capa de control dinámico MPC.

5.1.3. Capa de optimización LSSO

En esta capa se plantea maximizar la cantidad de metanol destilado en el tope de la columna bajo el siguiente criterio 5.2:

$$\begin{aligned} J_{E1} &= -_{P_D} D^{ss} + _{P_B} B^{ss} + _{P_R} R^{ss} + _{P_V} V^{ss} \\ sujeto\ a: \quad & R_{min} \leq R^{ss} \leq R_{max} \\ & V_{min} \leq V^{ss} \leq V_{max} \\ & x_D^{ss} \geq 0.95 \\ & x_B^{ss} \leq 0.05 \end{aligned} \tag{5.2}$$

En donde:

D^{ss} y B^{ss} son las tasas de salida de flujo en Kgmole/h en tope y fondo de la columna respectivamente.

P_D y P_B precios asociados a las tasas de salida en $/Kgmole.

R^{ss} y V^{ss} hacen referencia a los caudales de realimentación en Kgmole/h.

P_R y P_V precios asociados a los caudales de realimentación en $/Kgmole.

El criterio JE1 busca minimizar el flujo de salida en el fondo de la columna y en los flujos de realimentación, en especial el flujo de reflujo R.

Cabe resaltar que el modelo en régimen permanente es utilizado para calcular las salidas optimas del proceso, respetando las restricciones impuestas para las mismas, que este caso la concentración o fracción de metanol en el tope sea ≥ 0.95 y en el fondo sea ≤ 0.05.

5.1.4. Capa de control avanzado MPC

Esta capa es la encargada de llevar de manera segura a que el proceso alcance las consignas optimas dadas por la capa de optimización, puesto que el control avanzado MPC impone restricciones físicas, logrando una operación segura.

La sintonización del controlador se realiza a partir del modelo identificado previamente con los siguientes parámetros de sintonización.

- Horizonte de predicción $N = 20$
- Horizonte de control $N_u = 5$
- Matrices de ponderación $\mathbf{W}_y = diag([10, 10])$ y $\mathbf{W}_u = diag([1, 1])$
- Escalar de ponderación especifico $\lambda = 1$
- Tiempo de muestreo $T_s = 1$ segundo

5.1.5. Resultados implementación estructura de control jerárquica en entorno HILS para columna de destilación

Para la implementación de la estructura de control en el entorno HILS, se toma en cuenta lo explicado en el capítulo IV y se procede a implementar la capa de control avanzado en el PLC.

En primer lugar, se sintoniza el controlador a partir del modelo en espacio de estados de la columna de destilación y se configuran las restricciones para las variables manipuladas y controladas como se observa en los parámetros de sintonización.

Los parámetros de sintonización junto con el modelo en espacio de estados son introducidos dentro del bloque MPC en Matlab/Simulink con ayuda del toolbox “mpcDesigner” como se puede observar en la figura 5.6.

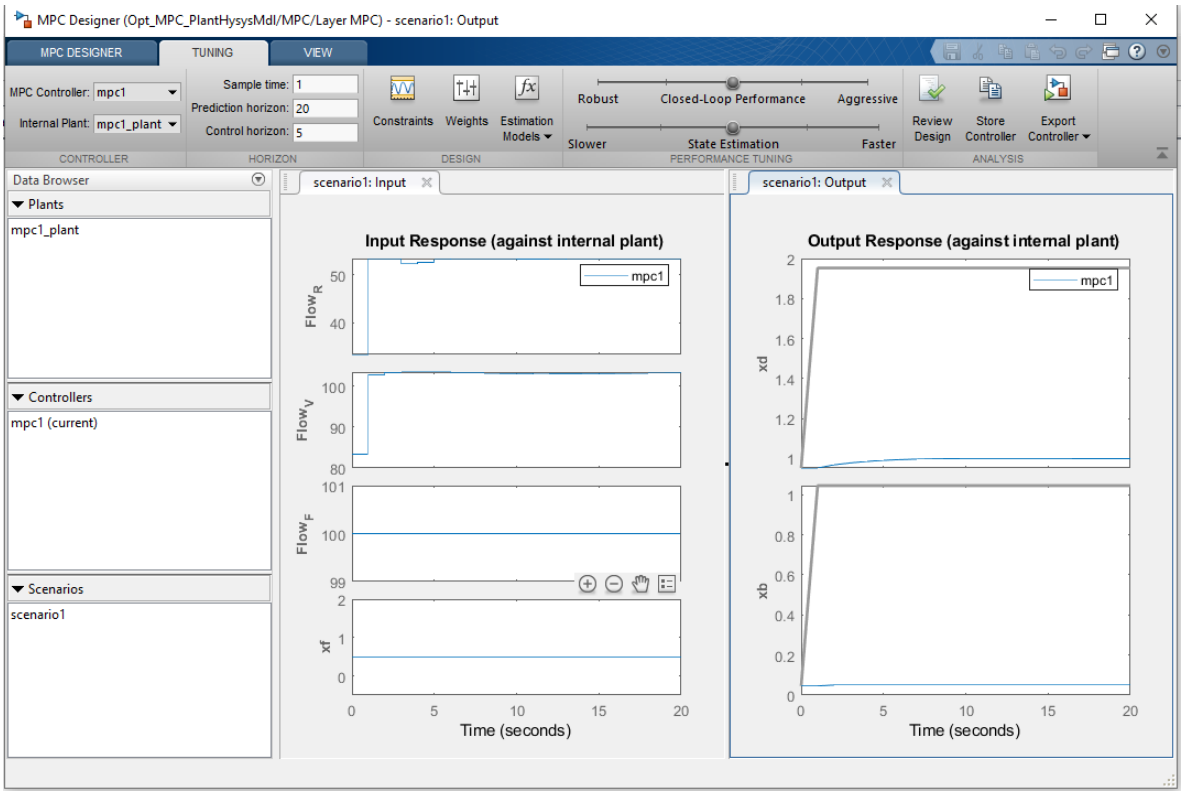

Figura 5.6: Interfaz visual mpcDesigner. Fuente: Propia.

Al terminar de introducir la información necesaria en dicho toolbox se procede a realizar una prueba rápida con el fin de observar la respuesta del controlador a nivel de simulación sin incluir el PLC en el lazo, es decir, Aspen HYSYS y Matlab/Simulink. En la figura 5.7 se puede apreciar el seguimiento de consigna de las variables controladas y en la figura 5.8 se puede observar el comportamiento de la función de optimización.

Tanto el controlador MPC como la optimización representada en la figura 5.8 en donde se evidencia que los beneficios de producción aumentan y los costos disminuyen muestran buenos resultados se procede a generar el código correspondiente con ayuda de la herramienta PLC Coder. Para ello, se crea un subsistema del bloque de control MPC con el fin de fijar un tiempo de muestreo, que en este caso es 1 segundo al igual que el tiempo de muestreo del controlador. Ahora se procede a ejecutar el toolbox PLC Coder en el cual se configura la plataforma de destino del código generado (SIEMENS TIA PORTAL) y el directorio de salida (Ver figura 5.9). (ver anexo B.1)

El toolbox también posibilita configurar la generación de comentarios en el código ge-

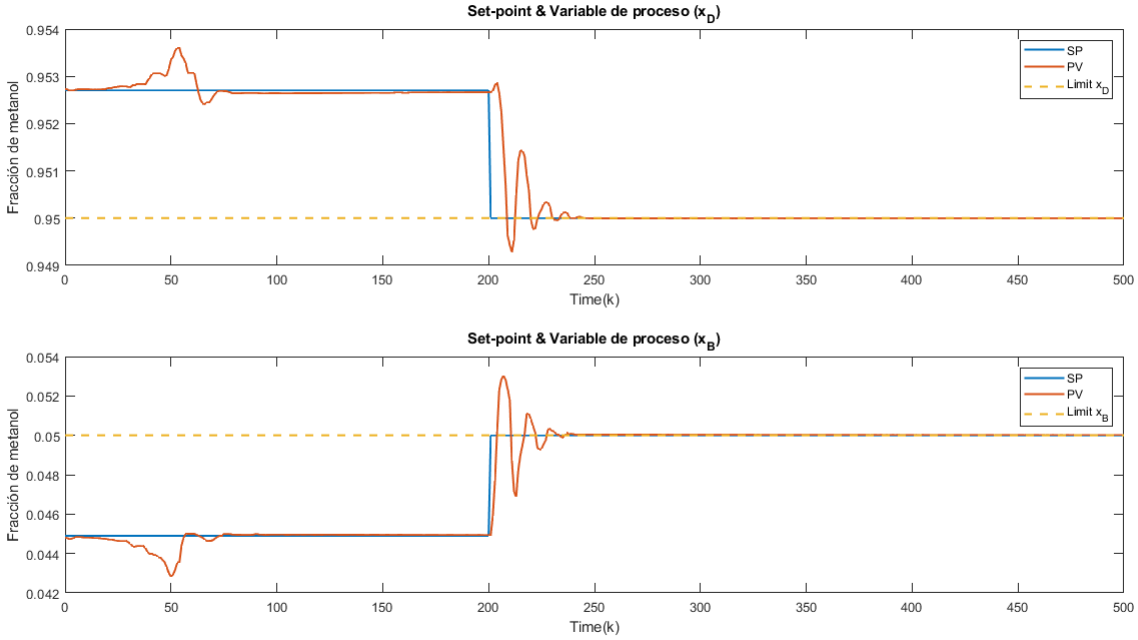

Figura 5.7: Respuesta esquema de control en la columna de destilación a nivel de simulación. Fuente: Propia

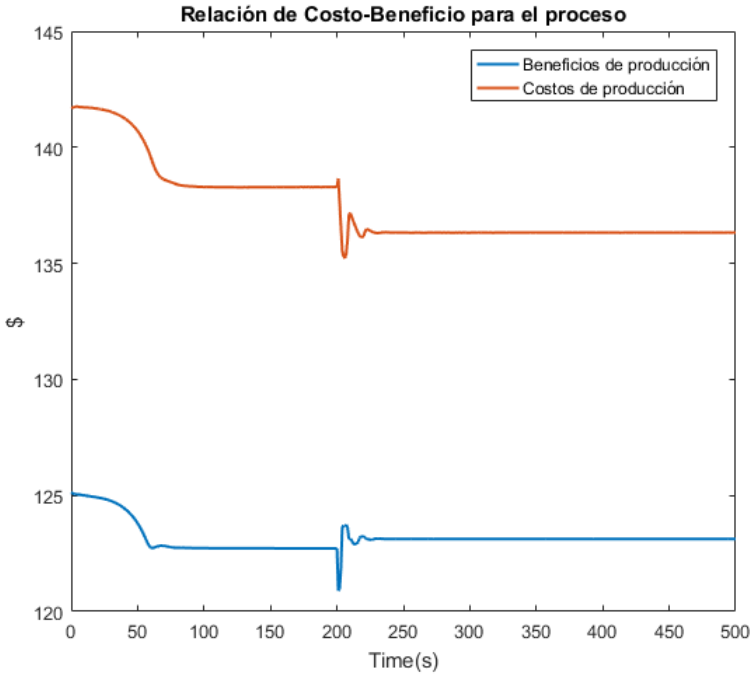

Figura 5.8: Costos y beneficios de producción alcanzados por la capa de optimización. Fuente: Propia

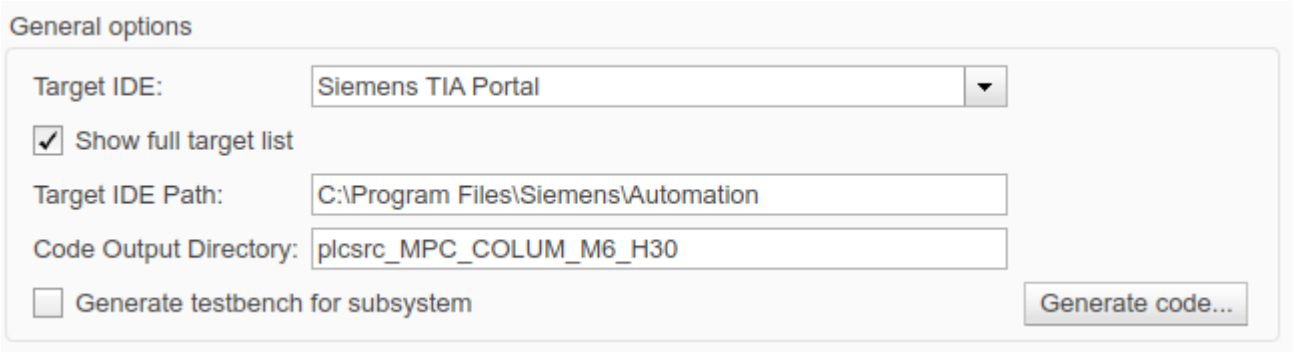

Figura 5.9: Configuración general PLC Coder. Fuente: Propia

nerado, opciones de optimización y generación de reportes para el mismo.

Para generar el código del bloque MPC, se da clic en el botón “Generate code. . . ” dando como resultado un archivo con la extensión .scl listo para ser desplegado en el PLC por medio del entorno de programación TIA PORTAL.

En el entorno de programación TIA PORTAL se importa el archivo .scl creado previamente y se procede a generar los bloques correspondientes. La generación de los bloques dentro del entorno de programación TIA PORTAL se realiza de manera automática, dando como resultado una serie de bloques función que hacen posible el funcionamiento del controlador MPC dentro del PLC. En la figura 5.10 se pueden ver los bloques generados en TIA PORTAL donde destaca el bloque llamado MPC0 [FB8], el cual contiene las funciones que articulan el algoritmo MPC. Los demás bloques son funciones auxiliares que se utilizan en tiempo de ejecución y no es necesario colocarlas en los bloques de organización, ya que la llamada de los bloques secundarios es automática al configurar el bloque que contiene los parámetros de entrada y salida del MPC (ver anexo B).

Para el correcto funcionamiento del controlador MPC dentro del PLC, se debe asegurar que la ejecución del mismo se realice en tiempos de muestreo regulares e iguales al tiempo de muestreo del controlador. Para ello, TIA PORTAL dispone de un bloque de organización llamado “Cyclic interrupt”, el cual puede ser configurado para que su ejecución se realice a intervalos de tiempo regulares como se puede observar en la figura 5.11.

Con el bloque MPC dentro del bloque de organización “Cyclic interrupt” se procede a realizar las pruebas en el entorno HILS obteniendo los siguientes resultados (ver figura 5.12).

En la figura **??** se puede observar que el controlador MPC es incapaz de seguir la consigna asignada, mostrando un comportamiento oscilatorio. Lo anterior se debe a que

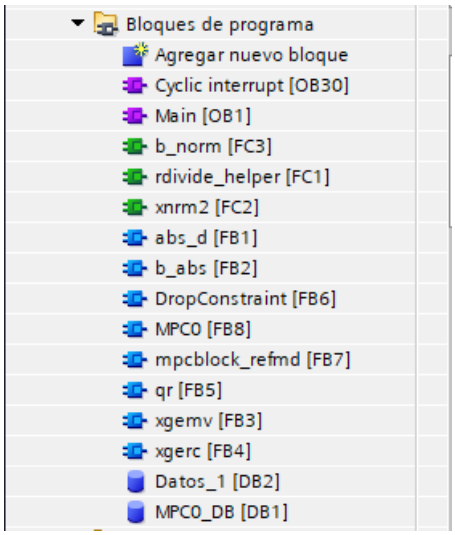

Figura 5.10: Bloques generados por TIA PORTAL para el controlador MPC. Fuente: Propia

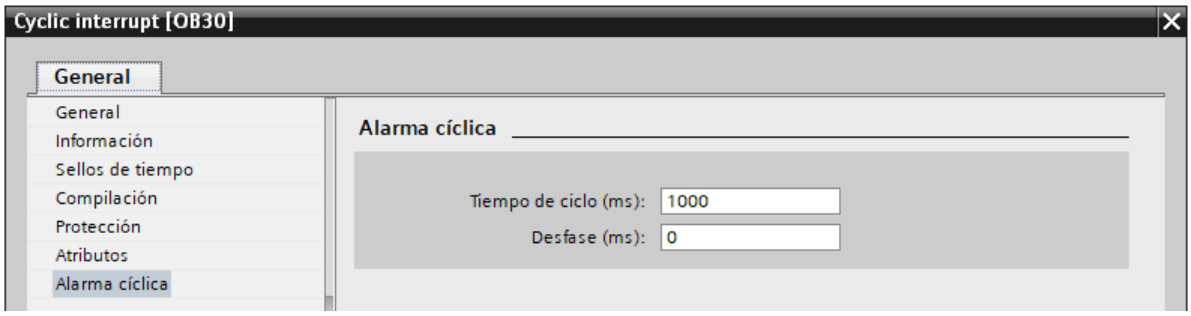

Figura 5.11: Configuración bloque Cyclic interrupt. Fuente: Propia

la columna de destilación que se ejecuta en el simulador Aspen HYSYS no se está ejecutando a intervalos regulares de tiempo, por lo cual el controlador recibe información de los estados de la planta que no corresponden a las predicciones realizadas y a las leyes de control aplicadas, por consiguiente, no es posible alcanzar el objetivo de control con esta planta. Por otra parte, en la figura 5.13 se puede ver la respuesta a la relación entre los costos y beneficios de producción otorgada por la capa de optimización cuya respuesta es similar a la esperada en la simulación de sintonización fuera de línea observada en la figura 5.8, lo cual evidencia que a pesar de que el seguimiento de las referencias entregadas por la capa de optimización presentaba un error creciente a medida que la simulación avanzaba en el tiempo, el controlador operaba con base en los criterios económicos predeterminados y, al mismo tiempo, toda la estructura implementada se encuentra recibiendo y enviando las señales correspondientes a las demás capas; por lo tanto, es posible deducir que el esquema de control implementado en el entorno de simulación funciona a pesar de que el elemento de simulación de la planta se encuentre no sincronizado por la ausencia de la configuración previa de tiempo real correspondiente.

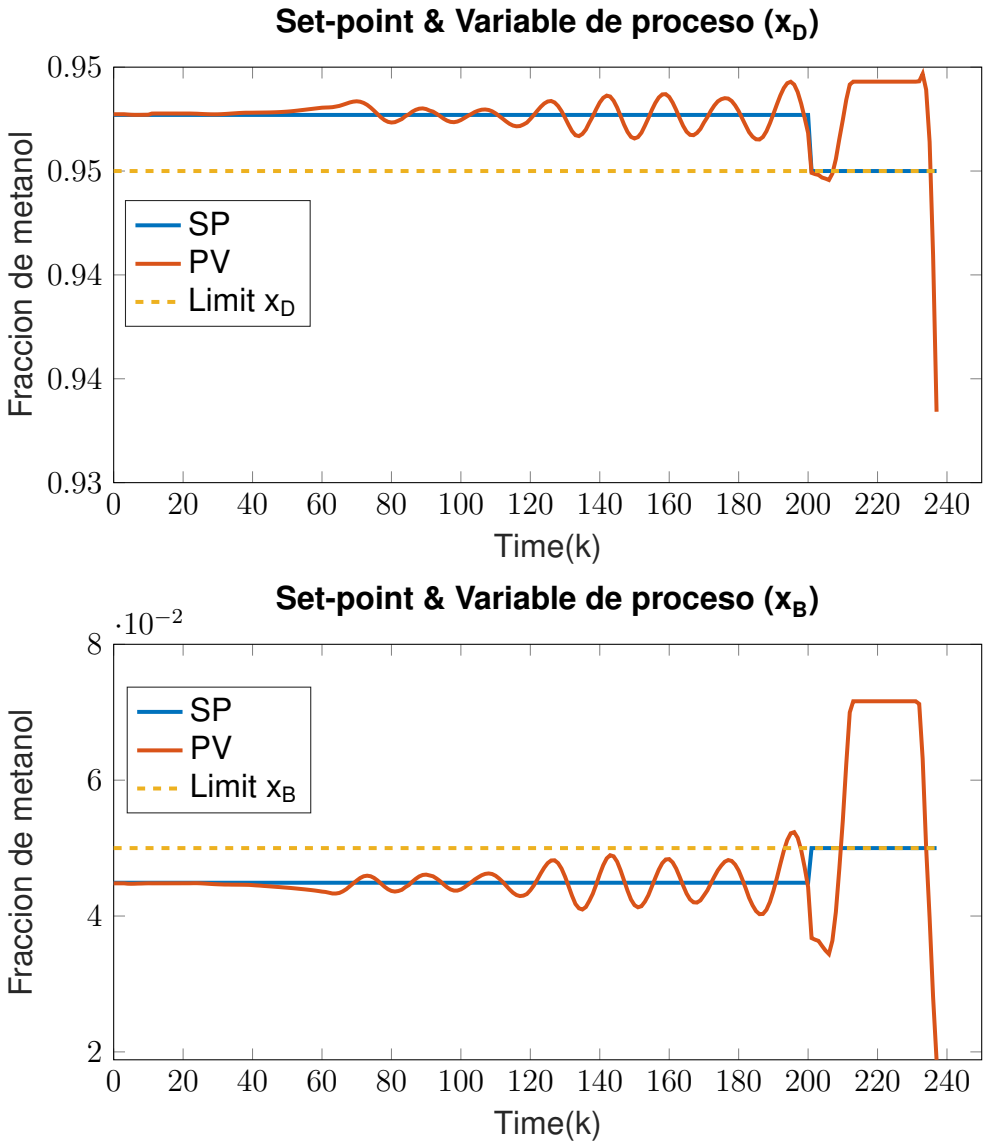

Figura 5.12: Respuesta esquema de control en la columna de destilación en entorno HILS. Fuente: Propia

Con el objetivo de contrastar los resultados para una simulación en la que todos los elementos se encuentren sincronizados bajo un mismo tiempo de ejecución, a continuación se repite la metodología de implementación expuesta para un caso de estudio diferente, a saber, un separador trifásico FWKO configurado para ejecutarse en tiempo real en su diseño.

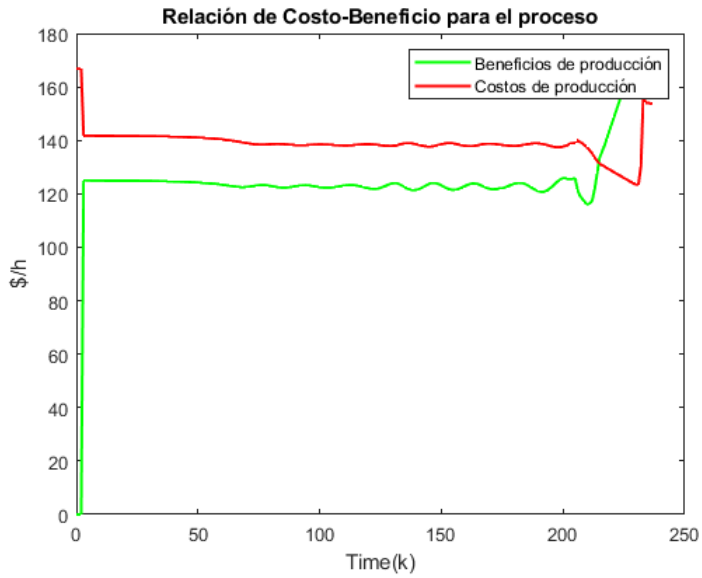

Figura 5.13: Costos y beneficios de producción alcanzados por la capa de optimización. Fuente: Propia

5.2. Desempeño del controlador en la estructura de control para el separador trifásico FWKO

5.2.1. Objetivo de control separador trifásico FWKO

El objetivo de control planteando para un separador trifásico FWKO en [49] es controlar la presión y la temperatura manipulando el flujo de energía y salida de gas. En la figura 5.14 se puede apreciar el separador junto con los controladores y las entradas y salidas al mismo. Cabe resaltar que el FWKO está modelado en software de simulación Aspen HYSYS.

5.2.2. Identificación del modelo para el FWKO

El proceso de identificación del modelo lineal para el FWKO se realizó de manera similar al proceso llevado a cabo para la obtención del modelo de la columna de destilación de metanol agua, aplicando señales binarias pseudoaleatorias en las entradas del FWKO y registrando los valores de las salidas. El modelo obtenido está representado por las matrices TAL.

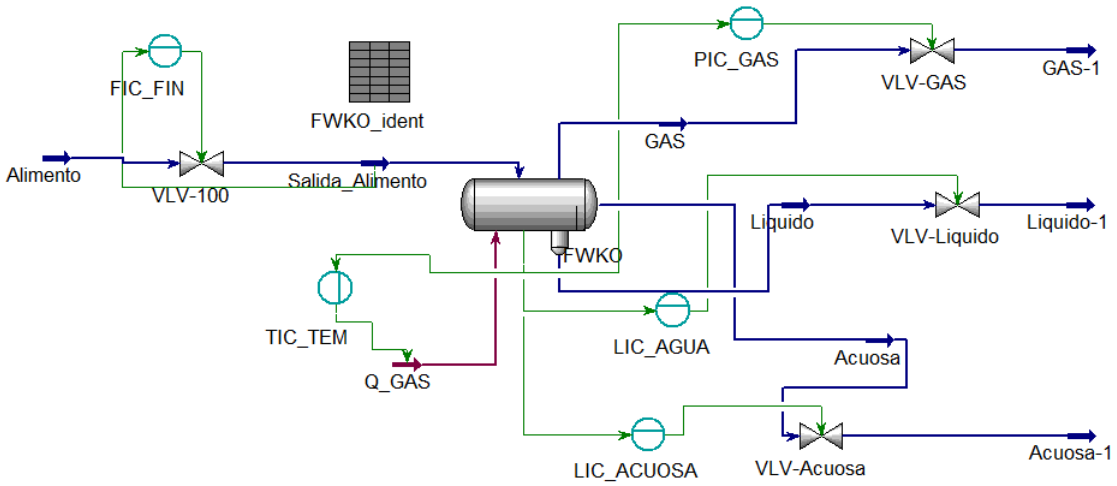

Figura 5.14: Separador trifásico FWKO. Fuente: Propia

$$\mathbf{A} = \begin{bmatrix} 0.9861 & -0.0787 & 0.0160 & -0.0974 & -0.0652 & -0.0321 & -0.0155 & 0.0024 & 0.0395 & 0.0011 & 0.0769 & -0.1229 \\ 0.0250 & 0.8396 & -0.1040 & -0.1054 & -0.0390 & -0.0510 & 0.1437 & -0.0840 & -0.0543 & -0.1315 & 0.1819 & -0.0639 \\ -0.0534 & -0.0598 & 0.8226 & -0.3779 & 0.0029 & 0.1197 & 0.2120 & 0.2027 & 0.2236 & -0.1007 & 0.1296 & -0.1710 \\ 0.0068 & -0.0207 & 0.3193 & 0.7416 & -0.1247 & 0.1210 & -0.2877 & -0.1460 & 0.0672 & 0.0331 & -0.0288 & -0.1435 \\ 0.0178 & -0.0287 & 0.1154 & 0.0136 & 0.6324 & -0.2058 & 0.3445 & -0.0740 & -0.0422 & -0.0375 & -0.1832 & 0.0664 \\ -0.0004 & 0.0281 & -0.1277 & -0.1036 & -0.0143 & 0.8097 & 0.2272 & 0.0619 & 0.0274 & 0.1606 & -0.1323 & 0.1483 \\ -0.0345 & 0.0140 & 0.2061 & 0.0987 & -0.1898 & -0.1589 & 0.3499 & 0.2883 & 0.2220 & 0.0450 & -0.0944 & -0.0482 \\ 0.1208 & -0.1133 & 0.1314 & -0.1215 & -0.3470 & -0.1890 & -0.2413 & 0.0697 & 0.1011 & 0.3120 & 0.1051 & -0.1725 \\ -0.0284 & 0.1334 & -0.2502 & -0.0226 & 0.4732 & 0.0810 & -0.2188 & 0.0756 & 0.7207 & 0.2641 & 0.0169 & 0.1213 \\ -0.0532 & 0.0977 & 0.0030 & 0.2661 & -0.0292 & 0.0551 & 0.0485 & 0.0131 & -0.1742 & 0.6277 & 0.2196 & 0.3183 \\ 0.0419 & 0.0195 & 0.0353 & 0.0979 & 0.1037 & 0.0080 & 0.0464 & -0.2421 & -0.2560 & -0.2605 & 0.2596 & 0.2087 \\ -0.0651 & -0.1096 & 0.1321 & -0.2161 & -0.0739 & 0.0063 & -0.2845 & 0.4596 & -0.2245 & -0.0467 & 0.0016 & 0.4571 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} -0.0000 & -0.0255 & 0.0095 \\ 0.0006 & -0.0108 & 0.0030 \\ 0.0002 & -0.0279 & 0.0112 \\ 0.0003 & -0.0385 & 0.0185 \\ 0.0007 & -0.0900 & 0.0347 \\ 0.0008 & -0.0503 & 0.0205 \\ -0.0005 & 0.0614 & -0.0236 \\ 0.0028 & -0.0834 & 0.0393 \\ -0.0003 & -0.0083 & 0.0042 \\ -0.0016 & 0.1726 & -0.0705 \\ 0.0006 & -0.0698 & 0.0301 \\ -0.0012 & -0.1411 & 0.0516 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} -20.0555 & 5.1471 & 0.8072 & 0.5919 & -3.5039 & 4.8357 & -0.3268 & 1.8783 & -3.7762 & 0.4087 & -0.7305 & 2.9369 \\ -125.9750 & 10.2831 & 3.4032 & 5.3413 & -19.2578 & 25.7129 & -2.5051 & 10.2201 & -23.3693 & 2.9036 & -5.8103 & 17.7695 \end{bmatrix}$$

5.2.3. Capa de optimización LSSO

El objetivo principal a optimizar por esta capa es el consumo de flujo de energía que requiere el intercambiador de calor presente en el FWKO. Para lograr la optimización

se hace uso del modelo en régimen permanente, el cual se obtiene a partir del modelo lineal identificado, y de funciones de optimización como lo es fmincon y linprog presentes en el software Matlab/Simulink. El criterio de optimización JE se define de la siguiente forma:

$$JE = -U$$

Donde U representa el flujo de energía que ingresa al intercambiador de calor.

5.2.4. Capa de control avanzado MPC

Para la capa de control avanzado, los autores en [2] proponen los siguientes parámetros de sintonización.

- Horizonte de predicción $N = 60$
- Horizonte de control $N_u = 2$
- Matrices de ponderación $\mathbf{W}_y = diag([10, 10])$ y $\mathbf{W}_u = diag([1, 1])$
- Escalar de ponderación especifico $\lambda = 1$
- Tiempo de muestreo $T_s = 1$ segundo

5.2.5. Resultados implementación estructura de control jerárquica en entorno HILS para separador trifásico FWKO

El procedimiento para la sintonización de la capa de otpimzación y control avanzado se realiza de igual forma a la presentada en la sección tal. Por sonsiguiente, los resultados obtenidos se pueden evidenciar a continuación:

Resultados a nivel de simulación:
Se puede observar en la figura 5.15 que el controlador alcanza el objetivo de control tanto en la variable presión como en la variable temperatura.

Resultados en el entorno HILS
En la figura 5.16 se puede ver como el controlador es capaz de alcanzar el objetivo de control alrededor de los 1500 segundos, dado que, tanto el controlador como el simulador Aspen HYSYS en donde se encuentra el modelo del FWKO, se ejecutan a intervalos de 1 segundo. Por lo tanto, se debe tener especial cuidado en la configuración de los tiempos de muestreo y ejecución de los algoritmos que entren en juego en los entornos de simulación donde se incluyan dispositivos hardware como lo es un PLC, puesto que estos, al considerarse sistemas en tiempo real, interactúan de forma activa con elementos que posean dinámica conocida (deterministas) en relación con sus salidas, entradas y restricciones de tiempo.

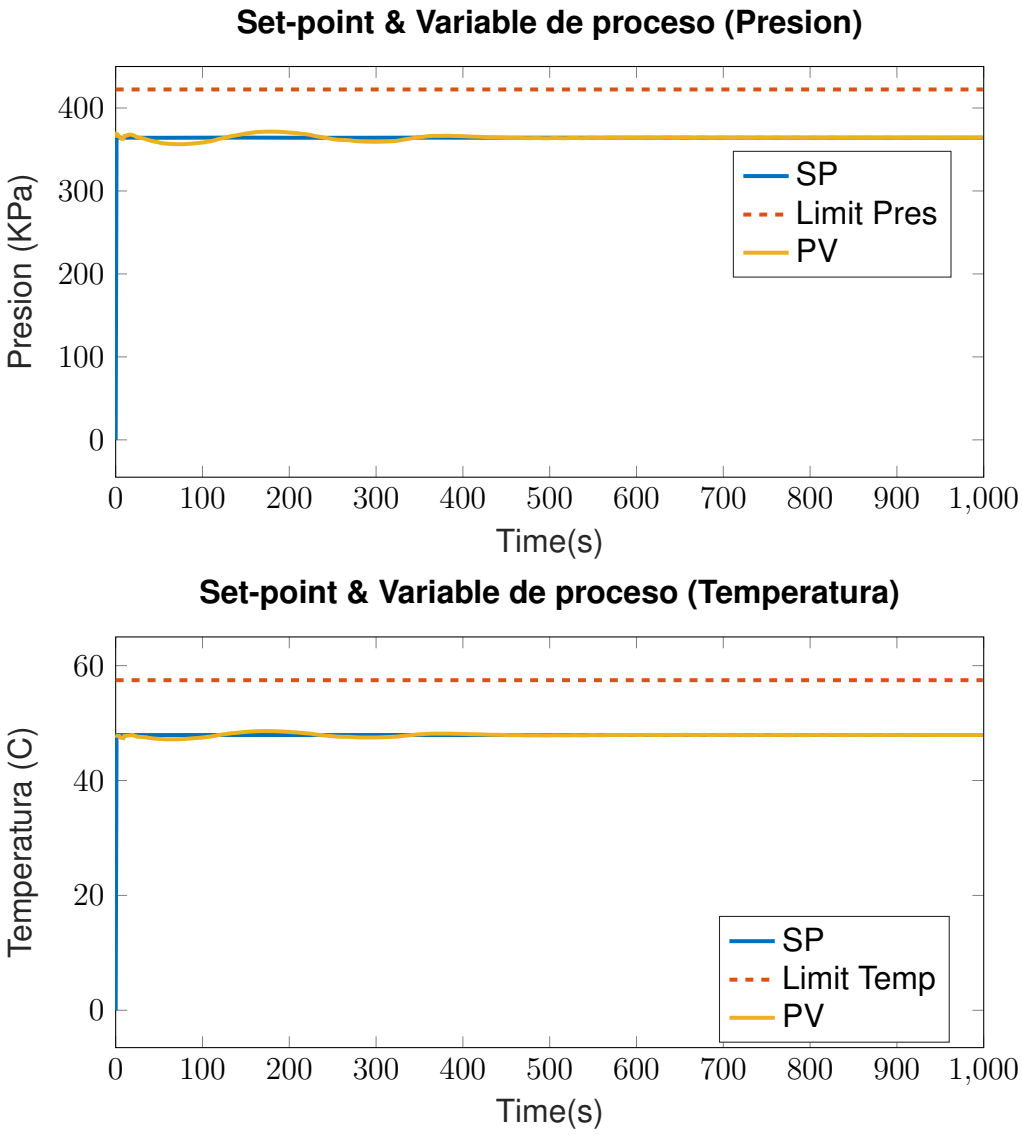

Figura 5.15: Respuesta de control MPC para seguimiento de consigna de presión y temperatura para un separador trifásico FWKO entorno simulación. Fuente: Propia

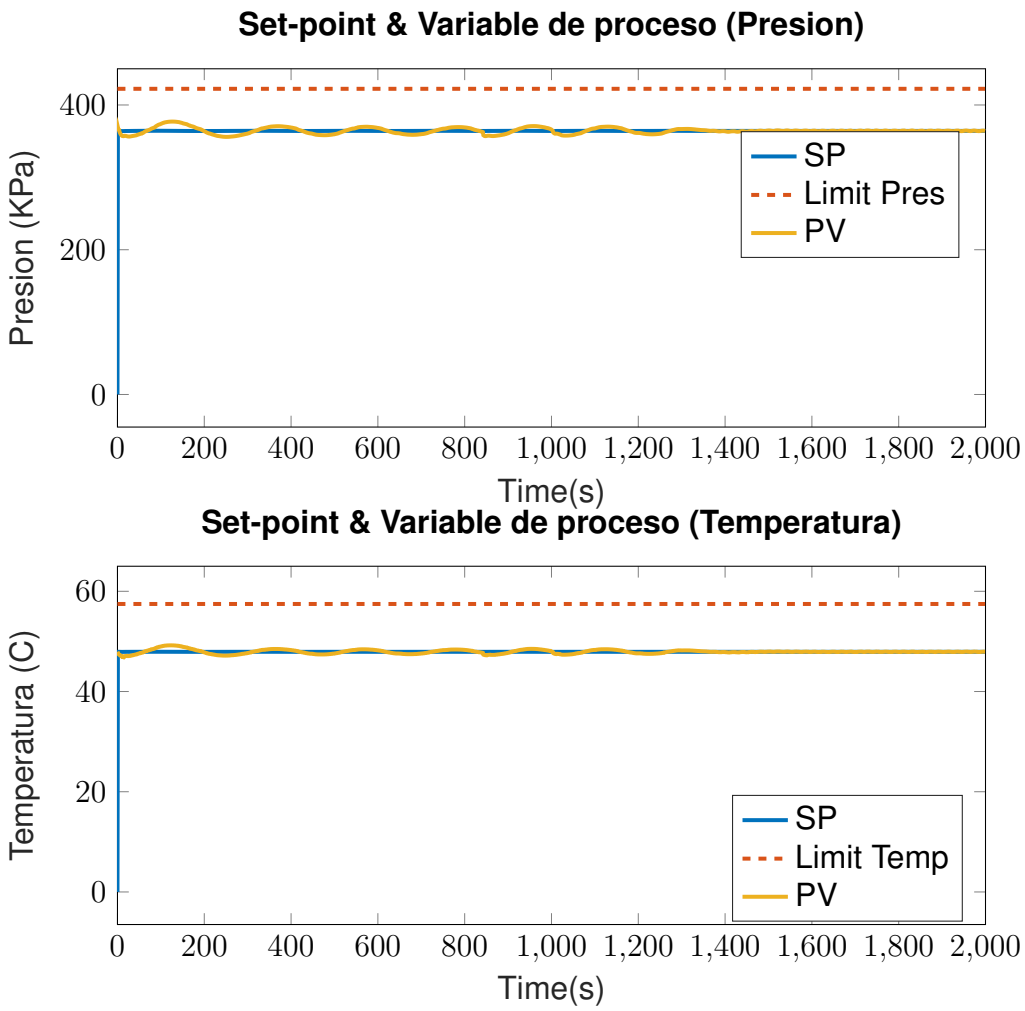

Figura 5.16: Respuesta de control MPC para seguimiento de consigna de flujo y temperatura para un separador trifásico FWKO entorno HILS. Fuente: Propia

5.2.6. Validación de la capa de optimización

En la figura 5.17 se puede ver la superficie creada para distintos valores de temperatura y presión los cuales, en conjunto, requieren un mayor o menor flujo de energía entrante al intercambiador de calor.

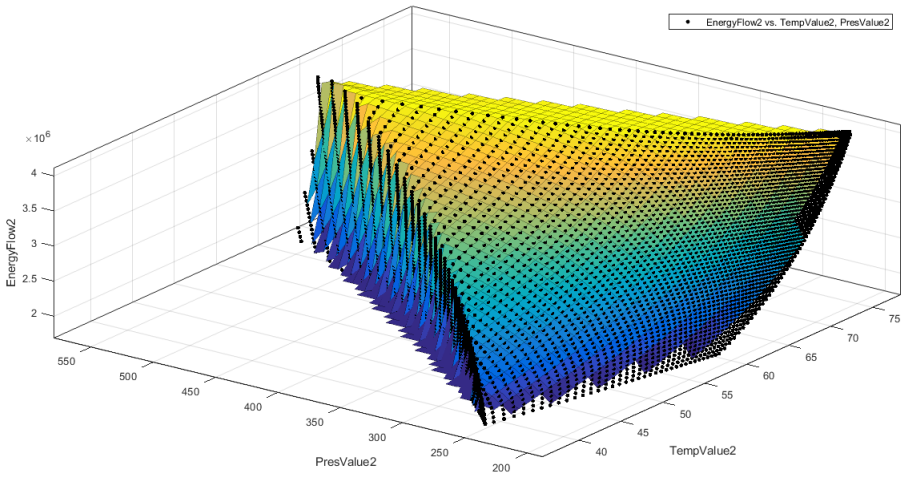

Figura 5.17: Flujo de energía para distintos valores de presión y temperatura. Fuente: Propia

Según los resultados obtenidos y plasmados en la figura 5.16, la capa de optimización está fijando como consignas óptimas una presión de 364.24 KPa y una temperatura de 47.92 grados centígrados. Este punto óptimo da como resultado un requerimiento de $1.8x10^6 KJ/h$ para el intercambiador de calor. Como se puede apreciar en la figura 5.18, este requerimiento de energía es un valor mínimo dado por la capa de optimización.

Como se pudo observar en este capítulo, la implementación y simulación de la estructura de control jerárquica planteada en el presente trabajo funcionó y desempeñó la tarea de control avanzado para los casos de estudio presentados. Sin embargo, fue necesario recurrir a una diferenciación de la configuración de tiempo real en las plantas simuladas para determinar la importancia de esta característica en la implementación del esquema en un entorno HILS, puesto que en la primera simulación para la columna de destilación, aunque se ejecutaba a medida que el tiempo corría, esta primera experimentación presentaba un error considerable y creciente en el seguimiento de la referencia óptima entregada, derivando finalmente en la parada forzada por parte del PLC cuando los puntos óptimos cambiaban debido a cambios en los disturbios medidos. Luego, en el segundo caso de estudio para el separador trifásico FWKO la implementación de la estructura también pudo ejecutarse en el entorno HILS con el

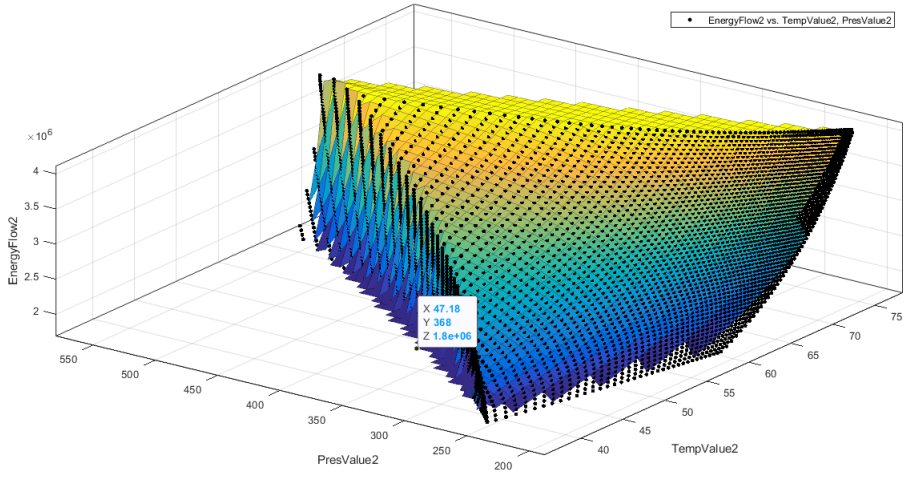

Figura 5.18: Punto optimo dado por la capa de optimización. Fuente: Propia

algoritmo de control predictivo corriendo en el PLC, con la diferencia de que el desempeño de la estrategia de control fue mucho más cercano a los resultados esperados por la simulación off-line debido a la sincronización de la planta simulada con los demás elementos de la estructura y del entorno, presentando un seguimiento de las consignas óptimas con error cero después del periodo transitorio y respondiendo ante los cambios en los disturbios.

Finalmente, en el siguiente capítulo se realizan una serie de experimentaciones y análisis de los resultados de control, en particular para los fallos presentados en el primer caso de estudio, con el objetivo de entender mejor las desventajas de la implementación propuesta y sintetizar las condiciones y requerimientos que pueden afectar dicha implementación.

Capítulo 6

Condiciones y requerimientos

Con el objetivo de evaluar los diferentes requerimientos de memoria en la implementación, los cuales deben ser soportados por el controlador industrial, se realizaron una serie de pruebas para las plantas simuladas en Aspen Hysys (columna de destilación y FWKO) con variaciones en el tamaño de los horizontes de predicción y en el orden del modelo de la planta a controlar, debido a que estos parámetros son los más incidentes en el consumo de memoria.

La implementación presentada en este trabajo se ha desarrollado en un PLC S7-1500 (en particular, la CPU 1512C-1 PN) de Siemens. Sin embargo, el lenguaje utilizado está incluido en el estándar de programación IEC 1131.3, por lo tanto, la aplicación desarrollada puede implementarse en cualquier otro PLC que sea compatible con IEC 1131.3 y que presente las características de memoria requeridas. La CPU mencionada posee 250 Kb de memoria de trabajo (para código) y 1 Mb de memoria de trabajo (para datos). El controlador fue implementado utilizando la plataforma de programación Simatic TIA Portal v15, la cual es totalmente compatible con IEC 1131.3.

La primera prueba consiste en variar el tamaño del horizonte de predicción N, fuera de línea, para que un nuevo controlador pueda ser generado desde el PLC Coder de Matlab, importado después en el TIA Portal y descargado en el controlador, con el fin de determinar la relación generada por los cambios en el parámetro mencionado con respecto al cambio de la memoria utilizada. El procedimiento es explicado paso a paso en el anexo B. Los resultados de las mediciones realizadas en el banco de pruebas con ayuda del entorno HILS se pueden observar en las siguientes tablas.

Las primeras pruebas con el horizonte de predicción N más bajo escogido, para el análisis de requerimientos de memoria y tiempo, fueron realizadas en un PLC S7-1200 de Siemens, sin embargo, como es posible observar en la tabla 6.1, y en la siguiente también, solo fue posible realizar la primera prueba, pues la capacidad para horizontes de predicción más amplios generaba un error como consecuencia de exceder la memoria permitida por la CPU del controlador; dicho mensaje "100016 bytes necesarios máximo 65534", permitió a la misma vez, descartar el PLC S7-1200 y sus características como un controlador industrial adecuado para la implementación de un esquema de control jerárquico y, a su vez, para lograr analizar la variación de los requerimientos de memoria con respecto al cambio de parámetros. Por lo tanto, las tablas 6.1 y 6.2, recopilan una información más útil y completa sobre los datos requeridos como se muestra a continuación.

Horizonte N	Memoria de trabajo (bytes)		Memoria de carga (bytes)		Memoria remanente (bytes)		Tiempo de ciclo (ms)		Archivo .scl KB
	Total	Ocupada	Total	Ocupada	Total	Ocupada	Más largo	Más corto	
10	102400	66954	4194304	700444	0	0	188	1	167
20	0	0	0	0	0	0	0	0	309

Tabla 6.1: Requerimiento para la planta FWKO orden 12 con S7-1200

Horizonte N	Memoria de trabajo (bytes)		Memoria de carga (bytes)		Memoria remanente (bytes)		Tiempo de ciclo(ms)		Archivo .scl KB
	Total	Ocupada	Total	Ocupada	Total	Ocupada	Más largo	Más corto	
10	102400	92788	4194304	921086	0	0	312	1	198
20	0	0	0	0	0	0	0	0	402

Tabla 6.2: Requerimiento para la columna de destilación orden 6 con S7-1200

En el caso de los controladores de la serie S7-1200 de Siemens, la CPU dispone tres zonas de memoria en donde se almacena el programa de usuario, los datos y la configuración. Se puede hablar de tres tipos de memoria: memoria de carga, memoria de trabajo y memoria remanente. La memoria de carga permite almacenar de forma no volátil el programa de usuario, los datos y la configuración; el programa de usuario se carga primero en esta área de la CPU. La memoria de trabajo cuenta con un almacenamiento volátil, es decir, que esta área de la memoria se pierde si se desconecta la alimentación del PLC; esta memoria generalmente almacena las partes del programa de usuario que son relevantes para la ejecución del programa, la CPU copia elementos del proyecto desde la memoria de carga a la memoria de trabajo. Finalmente, la memoria remanente puede almacenar datos (limitados) de forma no volátil de la memoria de

trabajo. Cuando se produce un corte de alimentación o una caída de tensión, la CPU al arrancar restaurará nuevamente esos valores; esta función debe estar previamente configurada para ello, por lo que en las pruebas que se hicieron en los PLC S7-1200 no era necesaria esta condición, el valor de esta memoria en las tablas es igual a 0.

Ahora, en el caso de los controladores S7- 1500 de Siemens, la memoria de trabajo de las CPU está dividida en dos áreas:

- Memoria de trabajo para código: la cual contiene partes del código del programa relevantes para la ejecución.
- Memoria de trabajo para datos: la cual contiene las partes de los bloques de datos y los objetos tecnológicos relevantes para la ejecución.

De la misma manera que en los controladores S7-1200, la memoria remanente es igual a 0 en las pruebas realizadas debido a que es una configuración de la cual se puede prescindir al realizar la pruebas con el aumento de N.

Horizonte N	Memoria de trabajo (bytes)				Memoria de carga (bytes)	Memoria remanente	Tiempo de ciclo (ms)		Archivo .scl KB
	Para datos		Para código						
	total	ocupada	total	ocupada	ocupada	ocupada	más largo	más corto	
10	20971520	18380	6291456	46629	0	0	74.915	0.270	167
20	20971520	35700	6291456	76945	0	0	75	0.237	309
30	20971520	59420	6291456	122941	0	0	110.526	0.279	517
40	20971520	89540	6291456	189891	0	0	155.759	0.352	800
50	20971520	126060	6291456	269692	0	0	93.055	0.351	1132
60	20971520	168980	6291456	366732	0	0	103.947	0.318	1532

Tabla 6.3: Requerimiento para la planta FWKO orden 12 con S7-1500

De la misma forma, se realizaron las pruebas correspondientes en la columna de destilación de orden 6, sin embargo, con un aumento del horizonte de predicción a pasos más cortos.

Una vez realizadas las pruebas y ordenados los datos, para determinar la contribución de memoria aportada por el aumento del horizonte de predicción, fueron graficados los datos de peso en KB del controlador generado y de los requerimientos de memoria con respecto al aumento de N.

En la figura 6.1 es posible observar que el tamaño del controlador crece cuadráticamente con el aumento del horizonte de predicción. Como se ha venido mencionando a lo largo del actual trabajo, el problema principal de la implementación de los algoritmos de control predictivo son los requerimientos de memoria necesarios; el tamaño del controlador generado en lenguaje SCL crece de esta forma debido a los cálculos

Horizonte N	Memoria de trabajo (bytes)				Memoria de carga (bytes)	Memoria remanente	Tiempo de ciclo (ms)		Archivo .scl KB
	Para datos		Para código						
	total	ocupada	total	ocupada	ocupada	ocupada	más largo	más corto	
10	20971520	34212	6291456	56662	0	0	156.740	0.274	198
15	20971520	45420	6291456	80460	0	0	75.451	0.324	292
20	20971520	59052	6291456	107210	0	0	75.138	0.237	402
25	20971520	75060	6291456	138969	0	0	207.351	0.284	533
30	20971520	93492	6291456	181341	0	0	446.484	0.974	695

Tabla 6.4: Requerimiento para la planta FWKO orden 12 con S7-1500

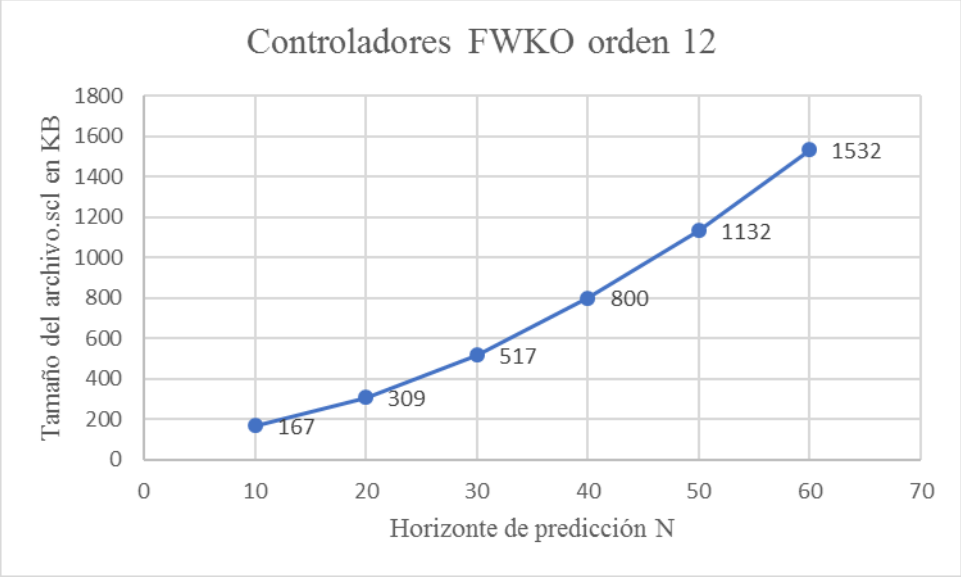

Figura 6.1: Tamaño del controlador en relación con el horizonte de predicción para la planta FWKO. Fuente: Propia

realizados para más instantes de tiempo futuros, ya que el almacenamiento de las matrices tanto del modelo como de la formulación del MPC y las operaciones necesarias de optimización y control son generadas automáticamente (por lo tanto sin restricciones de memoria) por el PLC Coder.

El mismo comportamiento de crecimiento cuadrático puede ser observado para el caso de los controladores generados de la columna de destilación (ver figura 6.2). Aunque, cabe resaltar que si se compara el tamaño de un controlador para un mismo horizonte de predicción para las dos plantas analizadas, el controlador de la columna de destilación requiere más capacidad de procesamiento por parte del PLC, lo anterior se debe a que, como se explicó en el capítulo 3, en la formulación del esquema de control jerárquico de esta planta se miden dos disturbios en lugar de uno, y que estos son parámetros necesarios en la solución del problema de optimización, por lo tanto, el número de operaciones aumenta al aumentar la dimensión de las matrices, aún cuando

el grado de la planta FWKO sea mayor y, a la vez, permite deducir, parciamente, que el orden de la planta tiene una incidencia menor en el consumo de recursos computacionales que los relacionados con el horizonte de predicción o los requerimientos de control.

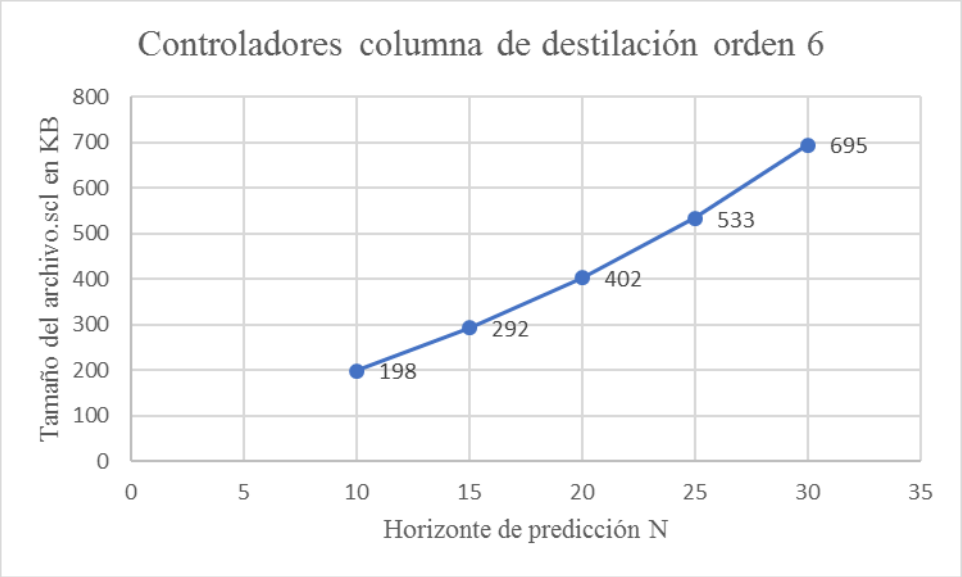

Figura 6.2: Tamaño del controlador en relación con el horizonte de predicción para la columna de destilación. Fuente: Propia

Como se había mencionado, el aumento de N es realizado a la mitad del valor de aumento para la planta FWKO, por lo que la parábola es menos pronunciada, sin embargo, es posible observar que para la columna de destilación este aumento tampoco presenta una relación lineal. Las gráficas anteriores permiten observar, en un primer plano, que el horizonte de predicción es un parámetro determinante a tener en cuenta para la implementación de estrategias de control jerárquicas como la del presente trabajo. Sin embargo, es posible determinar, en un plano de mayor profundidad, cuál es el área de memoria de la CPU más utilizada por el controlador. Las figuras 6.3 y 6.4, permiten observar el crecimiento de la memoria de trabajo para datos y la memoria de trabajo para código, empleadas por los controladores a medida que el horizonte de predicción N aumenta en la planta FWKO y la columna de destilación respectivamente.

En las figuras 6.3 y 6.4, es posible observar la distribución por áreas de memoria de trabajo ocupada en el PLC con la ejecución de los diferentes controladores generados. A la misma vez, las figuras demuestran que la memoria de trabajo para código ocupa un espacio mayor que la memoria de trabajo para datos, y que, al aumentar el horizonte de predicción, la diferencia entre sus respectivos requerimientos de memoria se hace cada vez más grande debido a su condición de crecimiento cuadrático.

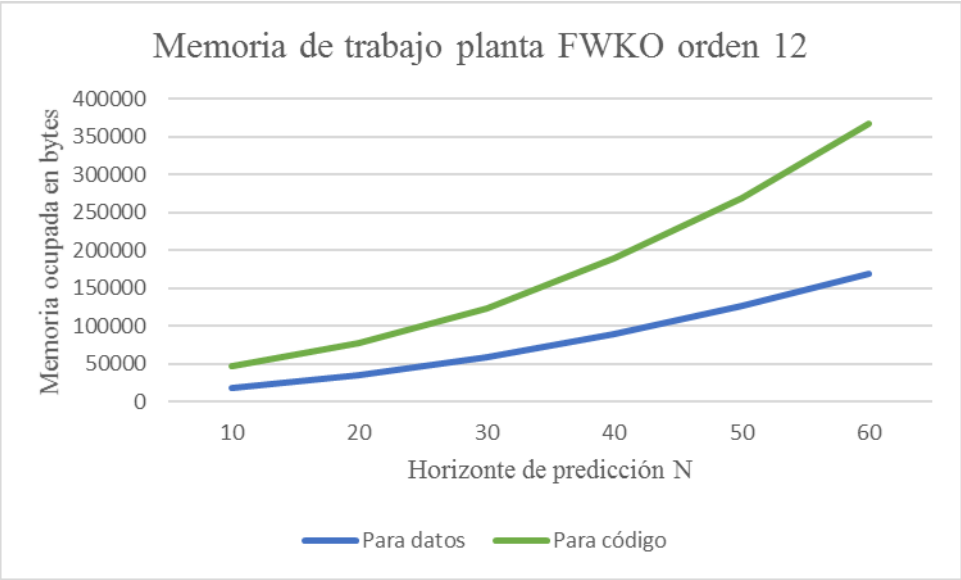

Figura 6.3: Memoria en relación con el horizonte de predicción para la planta FWKO. Fuente: Propia

En segundo lugar, la gráfica demuestra que, la metodología escogida en el actual trabajo para la generación del controlador MPC en lenguaje SCL, sobrecarga la memoria de trabajo para código en una proporción mayor que para los datos necesarios para correr el controlador. Es decir, que se están requiriendo desproporcionadamente muchas líneas de código para operar una cantidad de datos mucho menor, lo que se traduce en una eficiencia baja en el manejo de recursos computacionales.

El mismo análisis es soportado por la grafica de la figura 6.4, resaltando el hecho de que, debido a la condición de dos disturbios medidos para la columna de destilación, los recursos computacionales consumidos por los controladores generados para esta planta van a ser de un tamaño mayor y de menor eficiencia en su distribución; por consiguiente, el control realizado por el PLC: de menor calidad. Por lo tanto, es posible afirmar que el aumento de la complejidad de la planta a controlar, significa un porcentaje mayor de sobrecarga para la memoria del PLC y por lo tanto, una reducción en la viabilidad de implementación de la estructura multicapa por medio de generación de código SCL con el PLC Coder.

Después del análisis de distribución de memoria de trabajo y de la incidencia del horizonte de predicción en el tamaño del controlador, es importante mencionar que la memoria de carga a lo largo de las pruebas realizadas en el PLC S7-1500 mantuvo un valor de 0 en todas ellas (como puede corroborarse en las tablas 6.3 y 6.4); esto se debe a que los controladores generados por el método mencionado consumen toda la memoria de trabajo posible sin almacenar una porción de los datos y las operaciones en la memoria de carga.

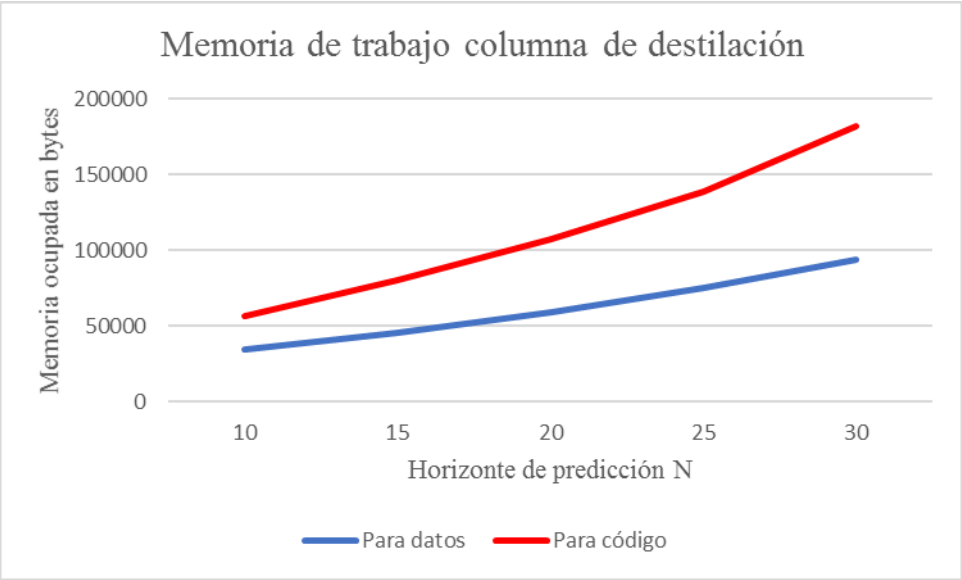

Figura 6.4: Memoria en relación con el horizonte de predicción para la columna de destilación. Fuente: Propia

Lo anterior confirma una desventaja más a considerar para la distribución de memoria con el método de generación de código semi automática, ya que, en el desarrollo y ejecución de un programa de forma manual, es posible asignar a la memoria de trabajo únicamente las operaciones más relevantes en la ejecución de un algoritmo de control, liberando de esta forma una posible sobrecarga en la ejecución del controlador sobre la memoria de trabajo (como se observó en la graficas anteriores).

6.1. Condiciones de tiempo real

Uno de los principales objetivos de la implementación del esquema de control en un entorno HILS, es el de posibilitar que las simulaciones puedan correr en sincronía con la evolución de los procesos a controlar, es decir, que cada paso de tiempo de simulación corresponda a la misma cantidad de tiempo de la planta real, por lo tanto, también requiere emular con la mayor precisión posible los tiempos de respuesta de esta.

La implementación propuesta en el actual trabajo integra tres elementos de simulación diferentes para el esquema multicapa: Simulink/Matlab, Aspen Hysys y un PLC S7-1500 de Siemens, en los que dicha sincronización desempeña un papel fundamental para que el control de la planta y la implementación del controlador alcancen resultados satisfactorios. Sin embargo, las herramientas mencionadas poseen características muy diferentes en cuanto a velocidad de procesamiento de datos y resolución o tamaño de los datos a calcular; por esta razón, la simulación en tiempo real se convierte en un factor determinante en la implementación de una estrategia de control jerárquica

en el entorno HILS planteado; debido a que el atraso o adelanto de los tiempos de computación, de cualquiera de los elementos mencionados, con respecto a los pasos de tiempo generales determinados para el entorno en su conjunto (un segundo para el presente caso), es capaz de generar que el PLC entre en modo STOP y la simulación no pueda seguir corriendo.

En una de las pruebas realizadas, por ejemplo, en la columna de destilación fue posible observar un comportamiento particular de la estrategia de control implementada. En esta, a pesar de seguir la metodología que se ha venido explicando, la señal de control no logró seguir la referencia óptima entregada por la capa de optimización LSSO; en su lugar, el error aumentaba con el paso del tiempo de simulación.

En primer lugar, en la figura 5.12 es posible observar que la respuesta del sistema es oscilatoria sin llegar a seguir la referencia y que esta oscilación crece a medida que la simulación avanza. La razón de este comportamiento se debe a dos factores; el primero es un error en el proceso de elaboración de la columna de destilación en el software de simulación Aspen Hysys, ya que no fue habilitada la opción de simulación en tiempo real cuando fue construida la planta, por lo tanto, los pasos de tiempo, en que se resuelve el problema de control para este elemento del entorno HILS creado, no coinciden durante la simulación con el de los demás elementos. Los pasos de tiempo para esta planta se resuelven en instantes de tiempo aleatorios, lo que significa que los tiempos computacionales de cálculo para este elemento pueden retrasarse o adelantarse en ciertos instantes de tiempo, generando esfuerzos de control y consignas que no corresponden a las requeridas por el problema en cada instante de tiempo. La no sincronización del entorno de simulación para el esquema de control, debido a la diferencia entre tiempos de cálculo computacional, se puede observar en la figura 6.5. Como se puede observar, la no sincronización de las tareas de control realizadas por cada capa puede generar acciones de control incorrectas y, por lo tanto, respuestas del sistema no deseadas, en particular para una estrategia de control que implemente un controlador MPC, debido a que en este se debe resolver un problema de optimización cuadrática en línea en cada paso del tiempo.

El segundo factor se debe al cambio de referencia, generado a partir de un cambio de valor en el disturbio de flujo de entrada, el cual ocasiona la parada de la simulación, cuando el PLC pasa de modo RUN al estado STOP, como consecuencia de la ausencia de parámetros necesarios para resolver el algoritmo de control en el instante de tiempo que estos son requeridos por el controlador.

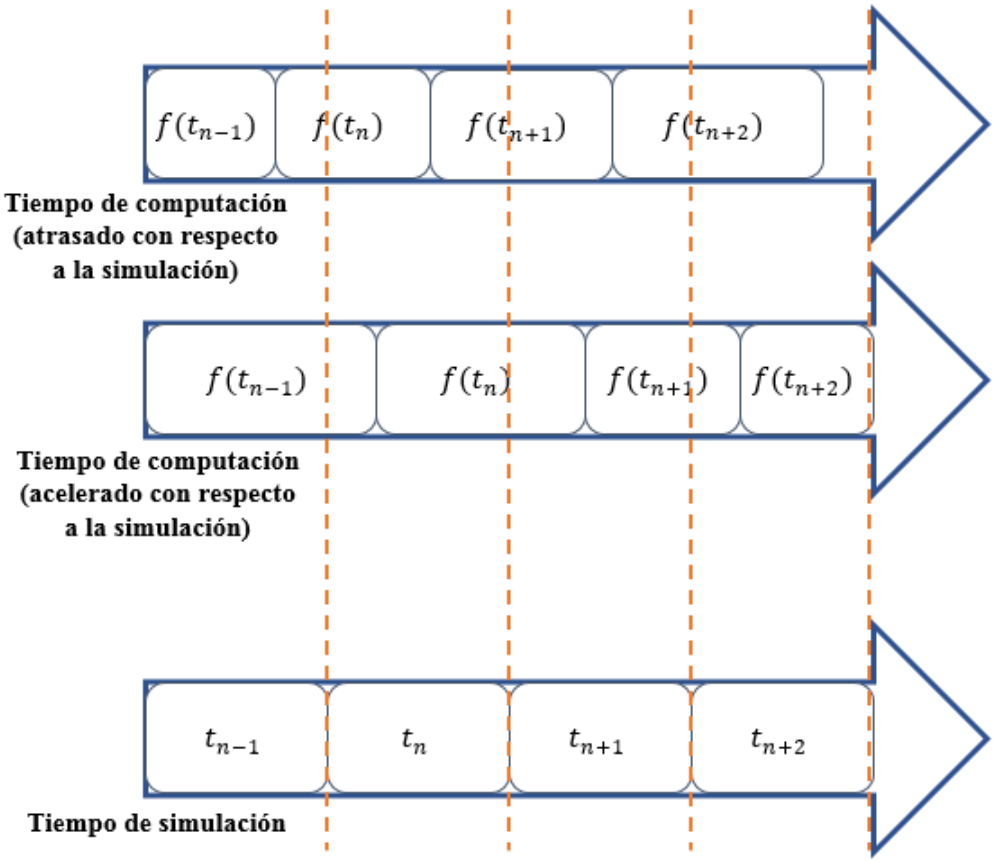

Figura 6.5: Simulación no sincronizada o fuera de línea. Fuente: Propia

Capítulo 7

Conclusiones y trabajos futuros

7.1. Conclusiones

La principal contribución del presente trabajo ha sido demostrar que se puede incorporar una estrategia de control multicapa sobre un controlador industrial estándar en un entorno HILS, con el planteamiento de una arquitectura apropiada que distribuya los objetivos de cada capa de forma eficiente para los recursos disponibles en el lazo de simulación.

Una de las características más limitantes para la implementación de estrategias de control avanzadas en un PLC es la memoria disponible. La metodología de generación de controladores en lenguaje SCL desde Matlab fue escogida con base en el aporte de agilidad que significaba en la realización de las pruebas necesarias que permitieron la evaluación del esquema propuesto y determinaron la viabilidad de implementación de este; sin embargo, de los resultados fue posible concluir que la asignación de memoria de este método puede sobrecargar la capacidad del PLC debido a la asignación ineficiente de recursos computacionales en la memoria de trabajo del controlador; por lo tanto, con el desarrollo del controlador MPC por medio de un algoritmo de reducción de la programación cuadrática, como los mencionados en el capítulo 4, se puede lograr un mejor uso de la estructura de las matrices y una mejor distribución de la ejecución del algoritmo, con lo que se lograrían menores requisitos de memoria para la implementación y, por consiguiente, menores tiempos en la ejecución del esquema de control.

También los diferentes parámetros del controlador predictivo embebido en la estructura jerárquica presentan una incidencia directa sobre los requerimientos de memoria. Debido a las diferentes pruebas realizadas con base en la variación de dichos parámetros, fue posible determinar que el tamaño del horizonte de predicción en la sintonización del MPC, aporta el mayor crecimiento en el tamaño del controlador. La segunda consideración se refleja en el número de disturbios medidos en el proceso, ya que estos

aumentan el número de cálculos a realizar por iteración en el problema de optimización. Por último, la reducción del orden del modelo de la planta tiene poco impacto en la reducción del tamaño del controlador, por lo que un procedimiento de reducción de orden de los sistemas no garantiza una reducción de memoria necesaria para que el algoritmo pueda ser ejecutado por el PLC como si lo es el horizonte de predicción en un grado mucho mayor.

Por otra parte, la ejecución en tiempo real desempeña un papel fundamental en la calidad del control realizado al proceso en el entorno HILS (como se demostró en la experimentación); desafortunadamente, múltiples razones pueden contribuir a la violación de este requerimiento y ocasionar, desde pérdida de optimalidad hasta ocasionar que el controlador entre en modo STOP. En primer lugar, en el procedimiento de elaboración del proceso a simular (Aspen Hysys) es determinante que se habilite la simulación en tiempo real para su posterior comunicación con el resto del entorno; de esta forma se garantiza la característica de sincronización en un escenario de simulación futuro. Sin embargo, es muy importante que tanto en la simulación del proceso como en la generación del controlador se haya configurado previamente el mismo tiempo de muestreo. En segundo lugar, dicho tiempo de muestreo está condicionado principalmente por el tipo de planta simulada y el proceso que esta ejecuta, es decir, que está determinado por la naturaleza de las variables que se desean optimizar y de los disturbios que intervienen, por lo tanto, para que la implementación de la estructura de control sea posible, se debe poner atención en la velocidad de sus dinámicas, las cuales, no deben forzar ni exceder la capacidad de procesamiento del controlador industrial escogido para la experimentación y de esta manera no ocasione la parada de este.

Aunque un entorno HILS permite la simulación en tiempo real del comportamiento de plantas industriales de complejidad considerable, la conexión y sincronización de sus elementos no es un procedimiento estandarizado en su configuración, lo que a menudo representa un consumo de tiempo que retrasa la etapa de experimentación para la que este fue creado. Por lo tanto, uno de los aportes destacables del actual trabajo es proveer, para investigaciones futuras, la selección, configuración y correcta puesta en marcha de una arquitectura de simulación para estrategias de control avanzadas, con lo cual, será posible invertir una menor cantidad de tiempo en la creación de un entorno de simulación en tiempo real y dirigir la investigación en un mayor porcentaje hacia el desempeño e implementación de las estrategias de control a la vanguardia.

Finalmente, aunque en un principio se pretendía evaluar la viabilidad de la propuesta de implementar un esquema de control jerárquico de dos capas, al culminar el presente trabajo se provee una base metodológica y secuencial de implementación de la estrategia de control mencionada, logrando superar la brecha de conectividad, configuración y no compatibilidad que antes se presentaba entre las posibles herramientas necesarias a manera de soluciones aisladas y por lo tanto insuficientes para dar respuesta a los requerimientos de esta línea de investigación tanto académica como industrial.

7.2. Trabajos futuros

Debido al numero de componentes que son requeridos en la implementación de un esquema de control jerárquico, existe un numero considerable de variantes que pueden significar un mejor desempeño del esquema propuesto en el actual trabajo, las cuales principalmente aportarán una mayor eficiencia en la distribución de memoria y una reducción en el esfuerzo computacional de los controladores industriales.

Una de estas variaciones es la implementación de un modelo no lineal para el controlador predictivo. Este enfoque permite la incorporación de valores iniciales en el algoritmo de control, lo cual resulta en tiempos de respuesta mínimos para cambios ocasionados por los disturbios del proceso y la aplicación inmediata de los resultados computacionales después de cada iteración, con lo cual, este enfoque puede adaptarse a los requisitos específicos de la optimización en tiempo real.

También, respecto al MPC, la adición de un algoritmo de identificación en línea del modelo utilizado por el controlador predictivo resultaría en un mejor desempeño en el control de procesos por medio de una estructura multicapa.

En cuanto a la capa de optimización, aunque su implementación se realizó por fuera del algoritmo embebido en el PLC, esto se debió principalmente a que las funciones utilizadas por Matlab para su solución no eran soportadas en el proceso de exportación por parte de la herramienta PLC Coder al lenguaje SCL. Sin embargo, es posible desarrollar una función de optimización propia desde Matlab que no contenga los comandos que imposibilitan su conversión por medio de esta herramienta.

Otra línea de investigación clave para trabajos futuros es el desarrollo de algoritmos que utilizan la mayor cantidad posible de información pre calculada (matrices Hessianas, gradientes y soluciones de programación cuadrática para trayectorias de referencia iteradas), con el objetivo de minimizar los tiempos de respuesta para utilizar eficientemente los recursos computacionales en línea.

Finalmente, cabe mencionar que los resultados de las investigaciones futuras reducirán de forma relevante la memoria ocupada de los controladores industriales, por lo tanto, el objetivo de implementar más capas del esquema de control dentro del mismo controlador (en lugar de solo una capa) se hace más cercano, simplificando, a su vez, la construcción del entorno HILS y mitigando los problemas de tiempo real que surgen debido a la comunicación entre un número mayor de elementos interconectados para su simulación.

Bibliografía

[1] P. Tatjewski, *Advanced control of industrial processes*. No. Mm, Springer, 2003.

[2] S. D. Fernando, "¿Qué es el Control Predictivo y Hacia Dónde se Proyecta?," 2012.

[3] B. Puchalski, T. A. Rutkowski, and K. Duzinkiewicz, "Implementation of the FO-PID Algorithm in the PLC Controller - PWR Thermal Power Control Case Study," *2018 23rd International Conference on Methods and Models in Automation and Robotics, MMAR 2018*, no. Cv, pp. 229–234, 2018.

[4] L. Gevorkov, V. Vodovozov, T. Lehtla, and Z. Raud, "PLC-based hardware-in-the-loop simulator of a centrifugal pump," *International Conference on Power Engineering, Energy and Electrical Drives*, vol. 2015-Septe, pp. 491–496, 2015.

[5] W. Dai, P. Zhou, D. Zhao, S. Lu, and T. Chai, "Hardware-in-the-loop simulation platform for supervisory control of mineral grinding process," *Powder Technology*, vol. 288, pp. 422–434, 2016.

[6] M. Iacob and G. D. Andreescu, "Real-time hardware-in-the-loop test platform for thermal power plant control systems," *SISY 2011 - 9th International Symposium on Intelligent Systems and Informatics, Proceedings*, pp. 495–500, 2011.

[7] H. T. Nguyen, G. Yang, A. H. Nielsen, and P. H. Jensen, "Hardware- and Software-in-the-loop Simulation for Parameterizing the Model and Control of Synchronous Condenser," *IEEE Transactions on Sustainable Energy*, vol. 3029, no. c, pp. 1–1, 2019.

[8] J. C. E. Duarte, A. Hernandez, and R. D. Keyser, "Evaluating the specification requirements for (N) EPSAC-MPC implementation on a programmable logic controller (PLC)," *Memorias del Congreso Internacional de Ciencias Básicas e Ingeniería - CICI 2016*, 2016.

[9] P. Pepe, M. P. Di Ciccio, and M. Bottini, "Digital control of a continuous stirred tank reactor," *Mathematical Problems in Engineering*, vol. 2011, 2011.

[10] A. M. Alzate Ibañez, "Modelado y control de una columna de destilación binaria," tech. rep., Universidad Nacional de Colombia sede Manizales, Manizales, 2010.

[11] M. Stewart and K. Arnold, “Crude Oil Treating Systems,” in *Emulsions and Oil Treating Equipment*, ch. I, pp. 1–80, 2009.

[12] K. A. Muñoz, *Diseño de controladores para la estrategia de control jerárquico en el problema de fijación de consignas óptimas en línea kevin andres muñoz*. 2019.

[13] K. Basak, K. S. Abhilash, S. Ganguly, and D. N. Saraf, “On-Line Optimization of a Crude Distillation Unit with Constraints on Product Properties,” pp. 1557–1568, 2002.

[14] V. Adetola and M. Guay, “Integration of real-time optimization & and model predictive control,” *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 7, no. PART 1, pp. 63–68, 2009.

[15] G. De Souza, D. Odloak, and A. C. Zanin, “Real time optimization (RTO) with model predictive control (MPC),” *Computers and Chemical Engineering*, vol. 34, no. 12, pp. 1999–2006, 2010.

[16] P. Tatjewski, “Advanced control and on-line process optimization in multilayer structures,” *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 11, no. PART 1, pp. 13–26, 2007.

[17] D. E. Kassmann, T. A. Badgwell, and R. B. Hawkins, “Robust steady-state target calculation for model predictive control,” *AIChE Journal*, vol. 46, no. 5, pp. 1007–1024, 2000.

[18] V. Adetola and M. Guay, “Integration of real-time optimization and model predictive control,” *Journal of Process Control*, vol. 20, no. 2, pp. 125–133, 2010.

[19] M. A. Brdys and P. Tatjewski, *Iterative Algorithms for Multilayer Optimizing Control*. 2012.

[20] Q. Pang, T. Zou, Y. Zhang, and Q. Cong, “A steady-state target calculation method based on "point"model for integrating processes,” *ISA Transactions*, vol. 56, pp. 196–205, 2015.

[21] S. J. Qin and T. A. Badgwell, “A survey of industrial model predictive control technology,” vol. 11, pp. 733–764, 2003.

[22] L. F. Pozas and L. V. R. de Arruda, “A New Approach to Integrate SSTO, MPC and RTO Using Online Identified Models: Use of Updated Models to Smooth Integration,” *Journal of Control, Automation and Electrical Systems*, vol. 29, no. 5, pp. 566–575, 2018.

[23] P. Tatjewski and M. Lawrynczuk, “Soft computing in model-based predictive control,” *International Journal of Applied Mathematics and Computer Science*, vol. 16, no. 1, pp. 7–26, 2006.

[24] T. A. Badgwell and S. J. Qin, *Model Predictive Control in Practice*. 2015.

[25] A. Rault, “Model Predictive Heuristic Control : Applications to Industrial Processes,” vol. 14, pp. 413–428, 1978.

[26] D. W. Clarke, “Application of Generalized Predictive Control to Industrial Processes,” no. April, pp. 49–55, 1988.

[27] M. Based and P. Control, “Industrial Applications of Model Based Predictive Control,” vol. 29, no. 5, pp. 1251–1274, 1993.

[28] E. F. J. Gómez Ortega and Camacho, “Mobile robot navigation in a partially structured static enviroment, using neural predictive control,” vol. 4, no. 12, pp. 1669–1679, 1996.

[29] M. L. Darby and M. Nikolaou, “MPC : current practice and challenges,” *Control Engineering Practice*, vol. 20, no. 4, pp. 328–342, 2012.

[30] Z. Cai, M. Yu, M. Steurer, H. Li, and Y. Dong, “A network model for the real-time communications of a smart grid prototype,” *Journal of Network and Computer Applications*, vol. 59, pp. 264–273, 2016.

[31] D. Thönnessen, S. Rakel, N. Reinker, and S. Kowalewski, “Matching Discrete Signals for Hardware-in-the-Loop-Testing of PLCs,” *IFAC-PapersOnLine*, vol. 51, no. 10, pp. 229–234, 2018.

[32] D. M. Lima, M. V. Americano da Costa, and J. E. Normey-Rico, “A flexible low cost embedded system for Model Predictive Control of industrial processes,” pp. 1571–1576, 2018.

[33] R. Hýl and R. Wagnerová, “Fast development of controllers with Simulink Coder,” *2017 18th International Carpathian Control Conference, ICCC 2017*, pp. 406–411, 2017.

[34] T. Puleva, G. Rouzhekov, T. Slavov, and B. Rakov, “Hardware in the Loop (Hil) Simulation of Wind Turbine Power Control,” pp. 1–8, 2017.

[35] S. J. Liu, D. Faille, M. Fouquet, B. El-hefni, Y. Wang, J. B. Zhang, and Z. F. Wang, “Dynamic simulation of a 1MWe CSP tower plant with two-level thermal storage implemented with control system,” *Energy Procedia*, vol. 69, pp. 1335–1343, 2015.

[36] T. Zang and A. Wang, “Design and application of the hardware-in-the-loop simulation system for distillation columns,” *Proceedings of 2009 International Conference on Image Analysis and Signal Processing, IASP 2009*, pp. 372–376, 2009.

[37] T. Roinila, T. Messo, R. Luhtala, R. Scharrenberg, E. C. De Jong, A. Fabian, and Y. Sun, “Hardware-in-the-Loop Methods for Real-Time Frequency-Response Measurements of on-Board Power Distribution Systems,” *IEEE Transactions on Industrial Electronics*, vol. 66, no. 7, pp. 5769–5777, 2019.

[38] K. Zhou and G. Rong, "Application of IDEF method in hardware-in-the-loop process simulation," *2011 International Conference on Management Science and Industrial Engineering, MSIE 2011*, pp. 1198–1200, 2011.

[39] S. Dominic, Y. Lohr, A. Schwung, and S. X. Ding, "PLC-Based Real-Time Realization of Flatness-Based Feedforward Control for Industrial Compression Systems," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 2, pp. 1323–1331, 2017.

[40] W. Chaaban, M. Schwarz, B. Batchuluun, H. Sheng, and J. Börcsök, "A partially automated HiL test environment for model-based development using Simulink® and OPC technology," *2011 23rd International Symposium on Information, Communication and Automation Technologies, ICAT 2011*, pp. 1–6, 2011.

[41] E. Joelianto, D. Chaerani, and A. Setiawan, "Hardware in the loop simulation of railway traffic re-scheduling by means of MILP algorithm," *Proceedings of 2011 2nd International Conference on Instrumentation Control and Automation, ICA 2011*, no. November, pp. 119–124, 2011.

[42] V. Philip, S. Jose, and S. Ashok, "Application of OPC protocol in islanded micro grid automation," no. Seiscon, pp. 560–563, 2012.

[43] E. Zaev, A. Tuneski, D. Babunski, L. Trajkovski, A. Nospal, and G. Rath, "Hydro power plant governor testing using hardware-in-the-loop simulation," *Proceddings of the 2012 Mediterranean Conference on Embedded Computin (MECO 2012)*, pp. 271–274, 2012.

[44] M. S. Mahmoud, M. Sabih, and M. Elshafei, "Using OPC technology to support the study of advanced process control," *ISA Transactions*, vol. 55, pp. 155–167, 2015.

[45] A. Felicísimo, "Conceptos básicos, modelos y simulación," in *Modelos Digitales de Terreno*, pp. 1–9, 1997.

[46] J. C. Oliden Semino, "Desarrollo de un controlador predictivo para plataforma industrial," 2016.

[47] B. Huyck, J. De Brabanter, B. De Moor, J. F. Van Impe, and F. Logist, "Online model predictive control of industrial processes using low level control hardware: A pilot-scale distillation column case study," *Control Engineering Practice*, vol. 28, no. 1, pp. 34–48, 2014.

[48] K. Sacha, "Automatic Code Generation for PLC Controllers," vol. 3688, no. II, pp. 303–316, 2005.

[49] J. Zambrano and W. Alegria, "Fijación de consignas óptimas en línea mediante identificación en línea del proceso," tech. rep., Universidad del Cauca, Popayán, 2019.

[50] M. ławryńczuk, P. Marusak, and P. Tatjewski, “Multilayer and Integrated Structures for Predictive Control and Economic Optimisation,” *IFAC Proceedings Volumes*, vol. 40, no. 9, pp. 198–205, 2007.

[51] Q. Pang, T. Zou, Q. Cong, and Y. Wang, “Constrained model predictive control with economic optimization for integrating process,” *Canadian Journal of Chemical Engineering*, vol. 93, no. 8, pp. 1462–1473, 2015.

[52] G. D. Souza, D. Odloak, and A. C. Zanin, *Real Time Optimization (RTO) with Model Predictive Control (MPC) Abstract :*, vol. 27. Elsevier Inc., 2009.

[53] D. E. Kassmann and T. A. Badgwell, “Robust Steady-State Targets for Model Predictive Control,” *IFAC Proceedings Volumes*, vol. 33, no. 10, pp. 413–418, 2000.

[54] T. Alamo, A. Ferramosca, A. H. Gonz, and D. Lim, “A gradient-based strategy for integrating Real Time Optimizer (RTO) with Model Predictive Control (MPC),” 2012.

[55] J. A. Paulson, “Modern control methods for chemical process systems,” 2017.

[56] C. O. Semino, “PARA PLATAFORMA INDUSTRIAL José Carlos Oliden Semino,” 2016.

[57] C. A. Gaviria and S. A. Sánchez, “Esquema de control jerárquico para fijación óptima de consignas en línea para un separador trifásico de un tren de tratamiento de crudo.” 2019.

Anexo A

Implementación entorno HILS

En el presente anexo se detallan paso a paso las configuraciones realizadas para la implementación del entorno HILS. Las siguientes configuraciones se hicieron utilizando los programas:

- SIEMENS TIA PORTAL V14 SP1
- Matlab/Simulink R2019a
- KepserverEx v6
- Aspen HYSYS V10

El computador utilizado en el presente proyecto cuenta con las siguientes características:

- Procesador Intel Core i5-4210U a 1.70GHz y 2.40GHz
- Memoria RAM 8 GB
- Windows 10 pro 64 bits
- HDD 1 TB

A.1. Conexión Matlab/Simulink y PLC SIEMENS

La siguiente configuración aplica tanto para el PLC S7-1200 como para el S7-1500.

1. Abra el programa SIEMENS TIA PORTAL y cargue el archivo MPC_CHAQUETA_3_1.ap14 desde el botón examinar si este no se encuentra en el espacio de “Últimos proyectos utilizados” (ver figura A.1).

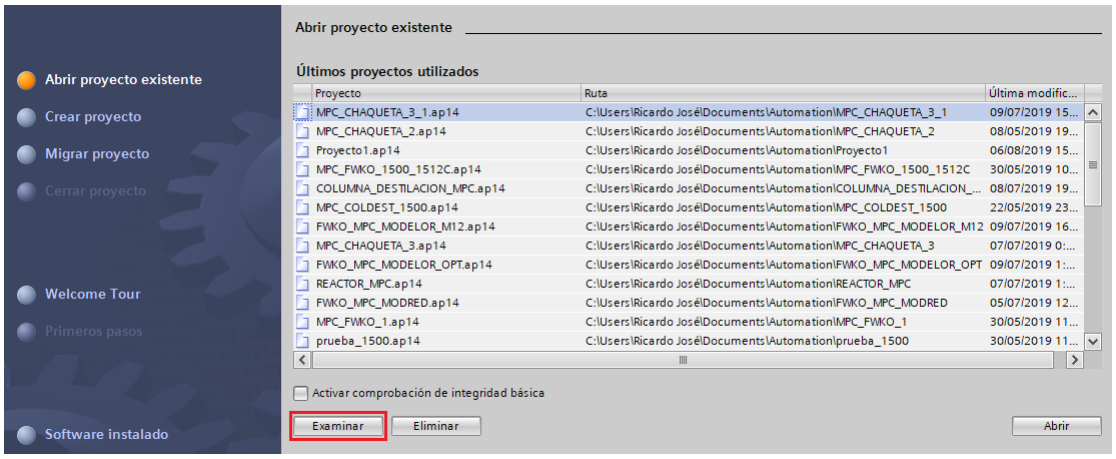

Figura A.1

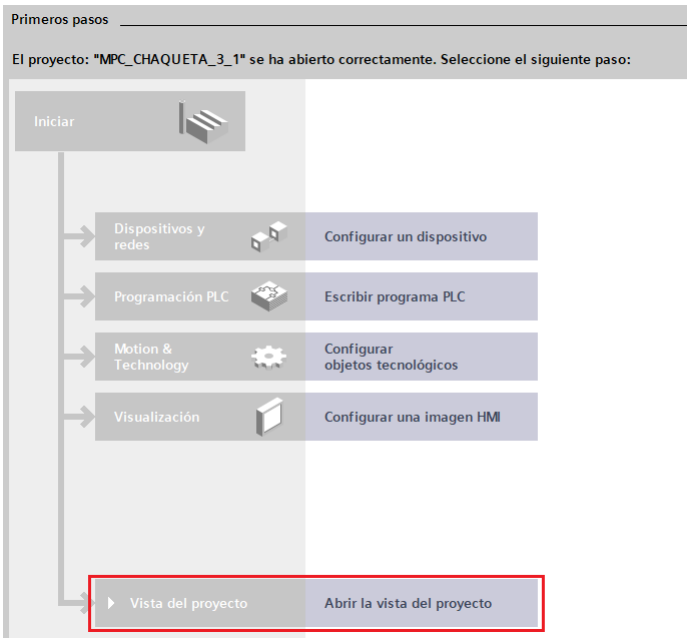

Figura A.2

2. Cuando el proyecto se haya cargado correctamente, de clic en “Abrir vista del proyecto” (ver figura A.2).

3. Se abrirá el entorno de programación TIA PORTAL V14. En la parte izquierda del entorno se encuentra el “árbol de proyecto”. De clic en “PLC_1 [CPU 1214C DC/DC/DC] (ver figura A.3).

4. Se desplegarán una serie de opciones de las cuales resaltan “Bloques de programa” y “Variables PLC” (ver figura A.4).

5. De clic en “Bloques de programa” en donde encontrará los bloques de organiza-

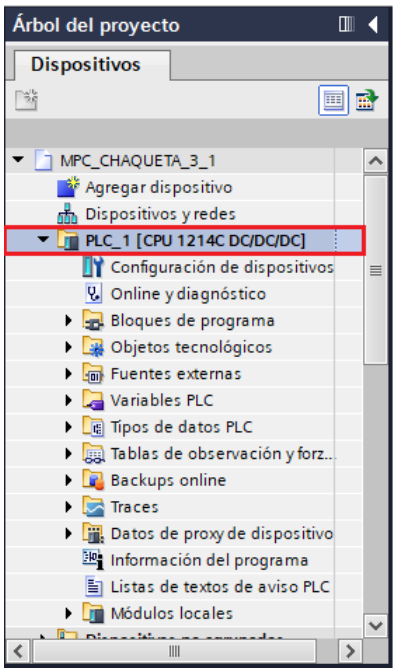

Figura A.3

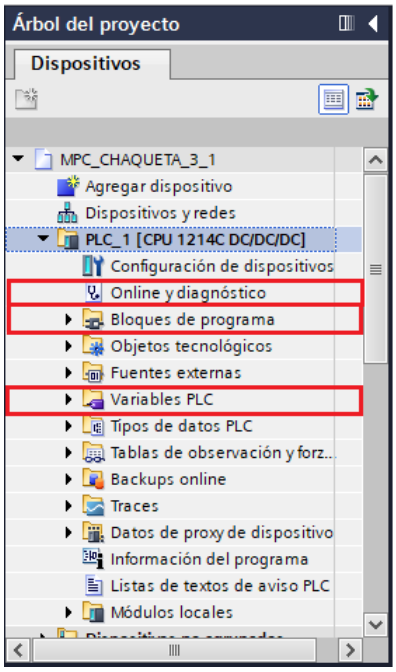

Figura A.4

ción [OB] en color morado; un bloque de función [FB] en color azul y un bloque de datos [DB] en color azul oscuro (ver figura A.5).

6. De doble clic en el bloque de organización “Cyclic interrupt” en donde encontrará

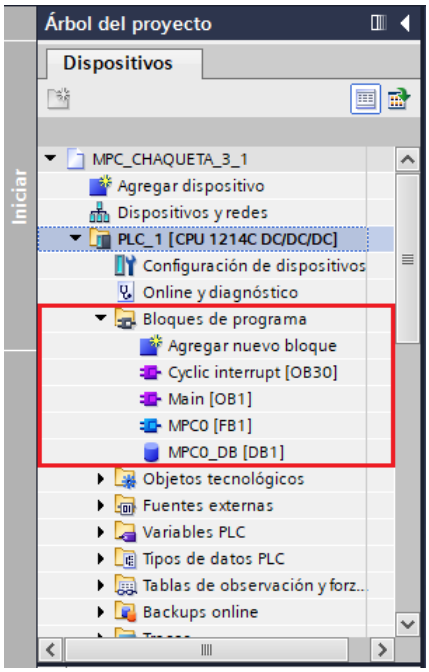

Figura A.5

el bloque del controlador MPC junto con las variables asignadas a cada entrada y salida del bloque (ver figura A.6).

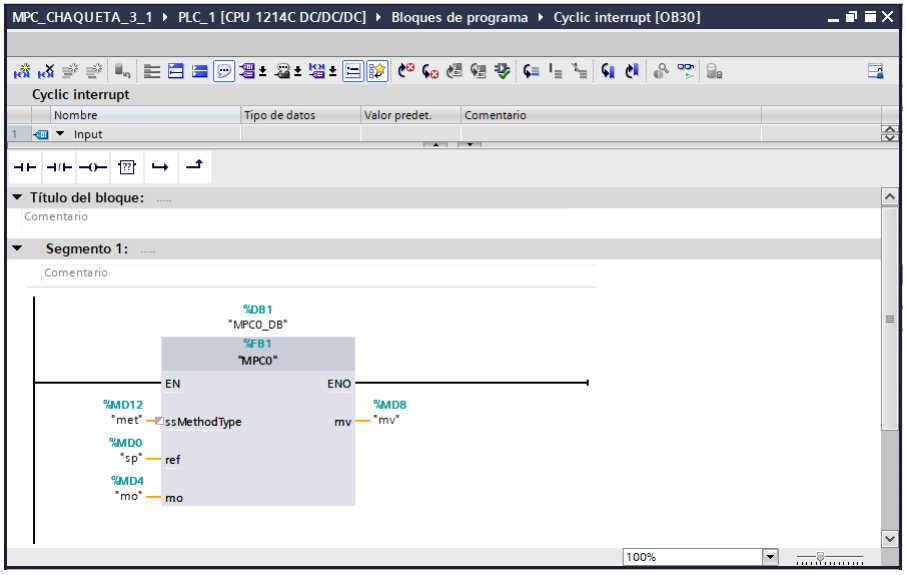

Figura A.6

7. Si en su caso es necesario agregar más variables, puede dirigirse a “Variables PLC” que se encuentra dentro del árbol de proyecto y luego “Mostrar todas las variables” (ver figura A.7).

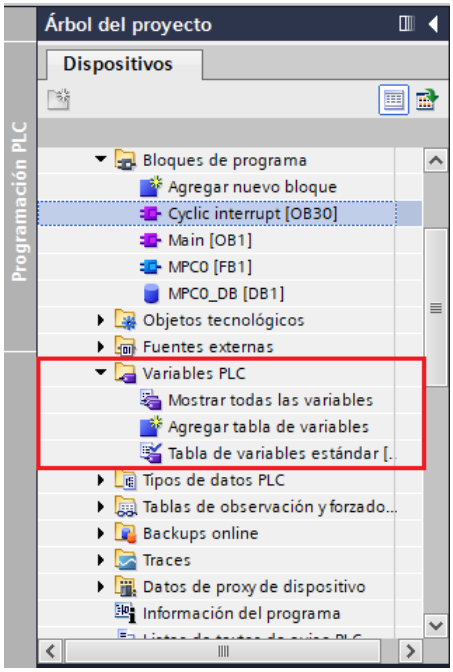

Figura A.7

8. En la ventana que se abre podrá ver las variables creadas con su respectivo nombre, tipo de dato y dirección. También podrá agregar más variables dando clic en la casilla “<agregar>” (ver figura A.8).

Figura A.8

9. Antes de cargar el programa al PLC, se debe asegurar que el mismo tenga deshabilitada la protección contra lectura y escritura. Para ello, de clic derecho en PLC_1 [CPU1214 DC/DC/DC] y luego en propiedades. Posteriormente se da clic en “Protección y Seguridad” y se marcan las opciones “Acceso completo (sin protección)” y “Permitir acceso vía comunicación PUT/GET del interlocutor remoto” (ver figura A.9).

10. Finalmente se procede a cargar el programa en el PLC, para lo cual se da clic

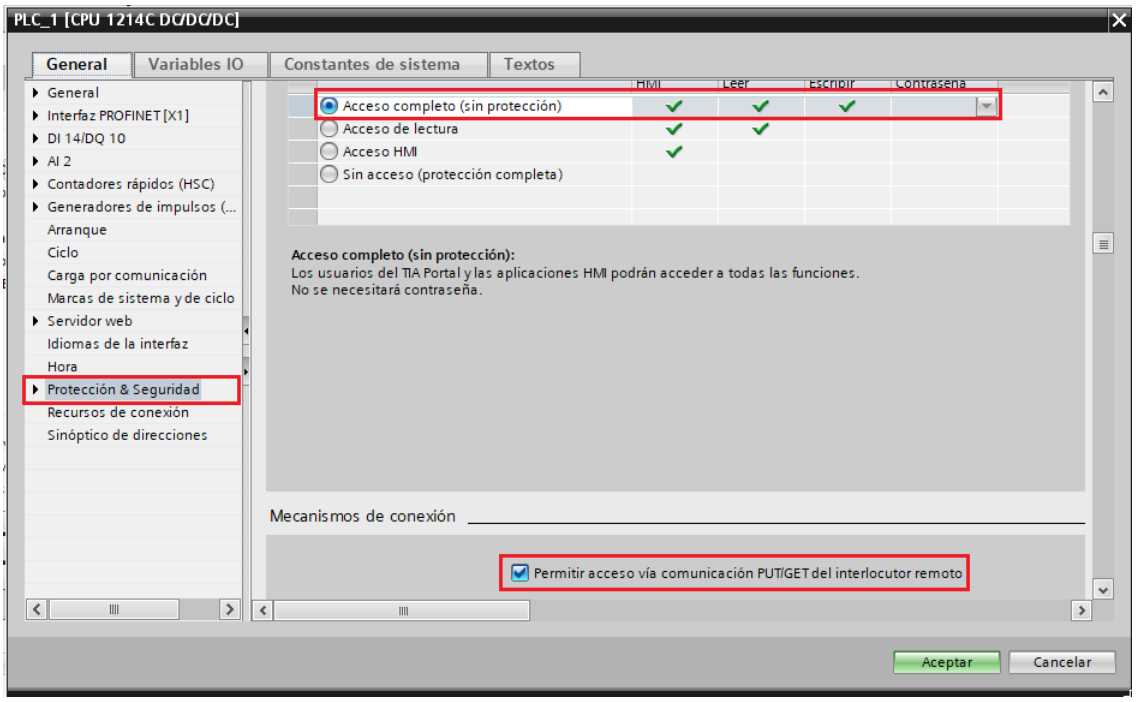

Figura A.9

en la pestaña “Online” y luego en la opción “Cargar en dispositivo”. Aparecerá una ventana en donde se aprecia la información del PLC como la referencia de la CPU y la ip del mismo, el tipo de interfaz y la tarjeta de red conectada al PLC por medio de un cable RJ-45. Luego se da clic en iniciar búsqueda y se espera hasta que aparezca el PLC en el cuadro “Seleccionar dispositivo de destino”. Se selecciona y se da clic en el botón cargar. Si hay más dispositivos conectados a la red, y no se tiene certeza a qué dispositivo cargar el programa, puede habilitar la opción “Parpadear led” y verificar qué PLC enciende y apaga las luces led (ver figura A.10).

11. Antes de cargar, el programa compilará el código. Cuando este proceso termine, de clic en cargar nuevamente.

12. Cuando la carga termine, aparecerá una ventana que da la opción de poner en modo RUN o dejar en modo STOP el PLC. Marque la opción “Arrancar todos” y de clic en el botón finalizar (ver figura A.11).

13. Ahora abra el servidor OPC KepserverEx v6 y desde la pestaña File/Open busque el archivo llamado “chaqueta”, de clic en abrir y luego en “yes, update” (ver figura A.12).

14. Cuando el archivo haya cargado correctamente, verifique que el “Channel” tenga el dispositivo PLC correcto y la IP asignada en el momento de carga del programa. Nota: Deberá asignar un IP fija a su tarjeta de red que corresponda a la

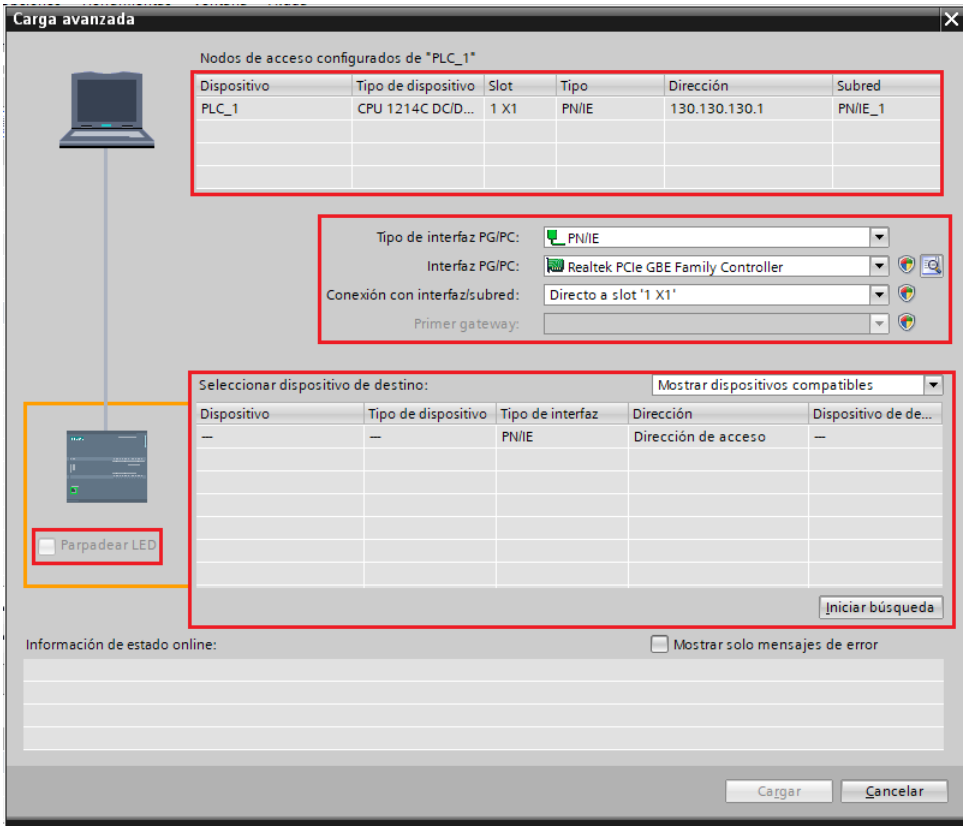

Figura A.10

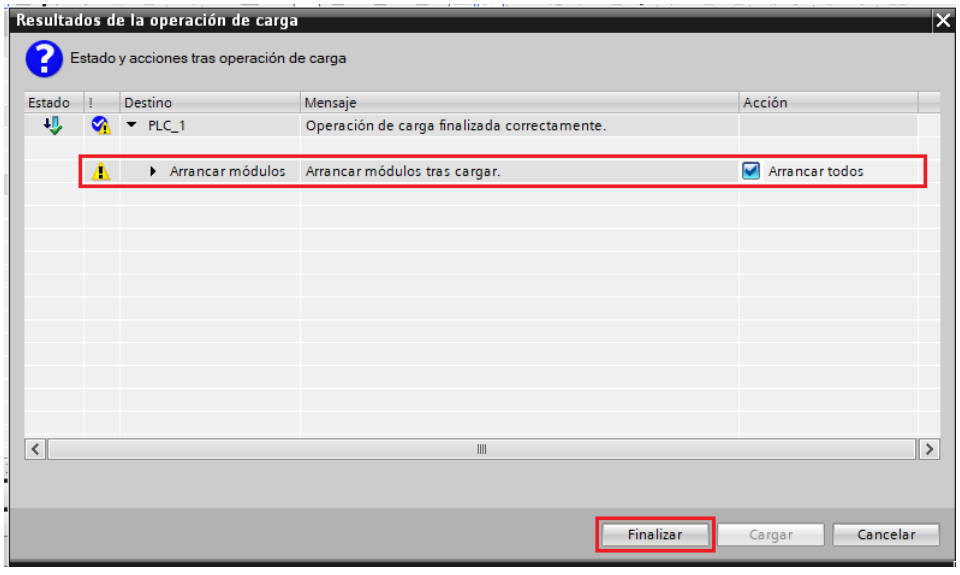

Figura A.11

misma red donde está conectado el PLC (ver figura A.13).

15. Luego, diríjase a “Device” en donde encontrará las tags creadas dentro del ser-

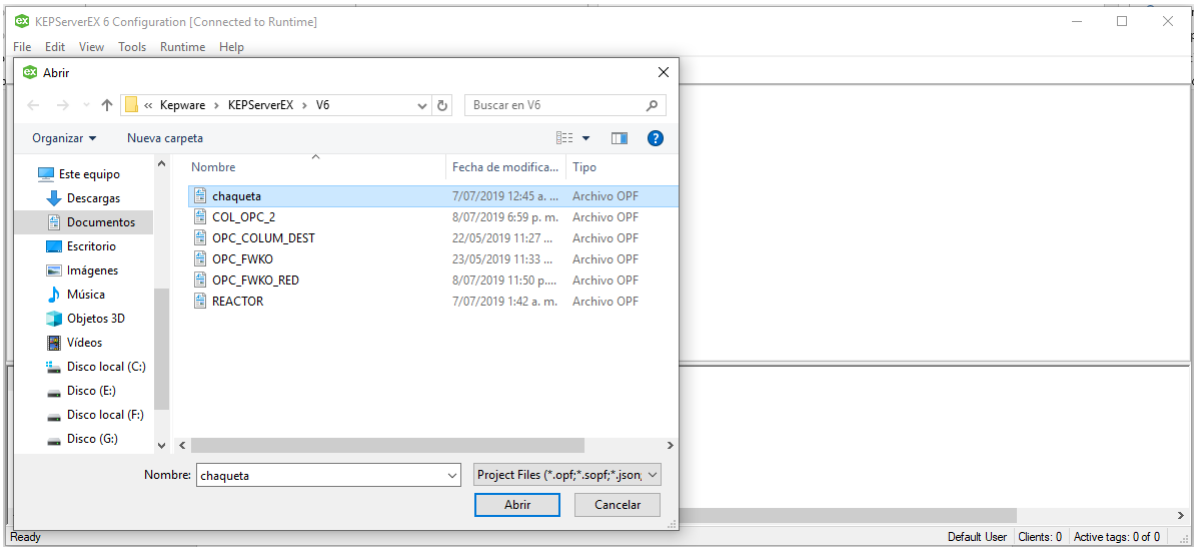

Figura A.12

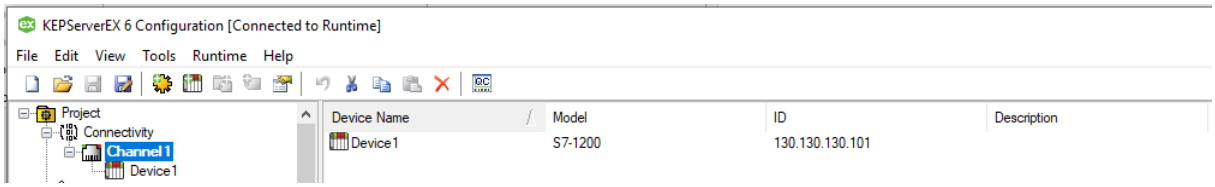

Figura A.13

vidor OPC las cuales corresponden a las variables creadas en TIA PORTAL y descargadas en el PLC (ver figura A.14).

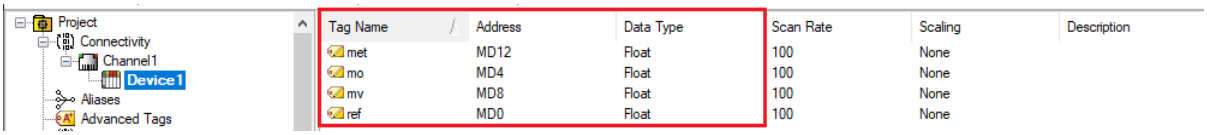

Figura A.14

16. Ahora abra Matlab/Simulink y busque los archivos llamados “CHAQUETA_MPC_OPC_1” y “MPC_CHAQUETA”. Ejecute el script “MPC_CHAQUETA”, luego configure la librería OPC de Matlab dando doble clic en el bloque “OPC Config Real-Time”, posteriormente en “Configure OPC Clients”, luego en “Add”, seguidamente en “Select” y finalmente escoja el servidor OPC que este caso corresponde a Kepserver (ver figura A.15).

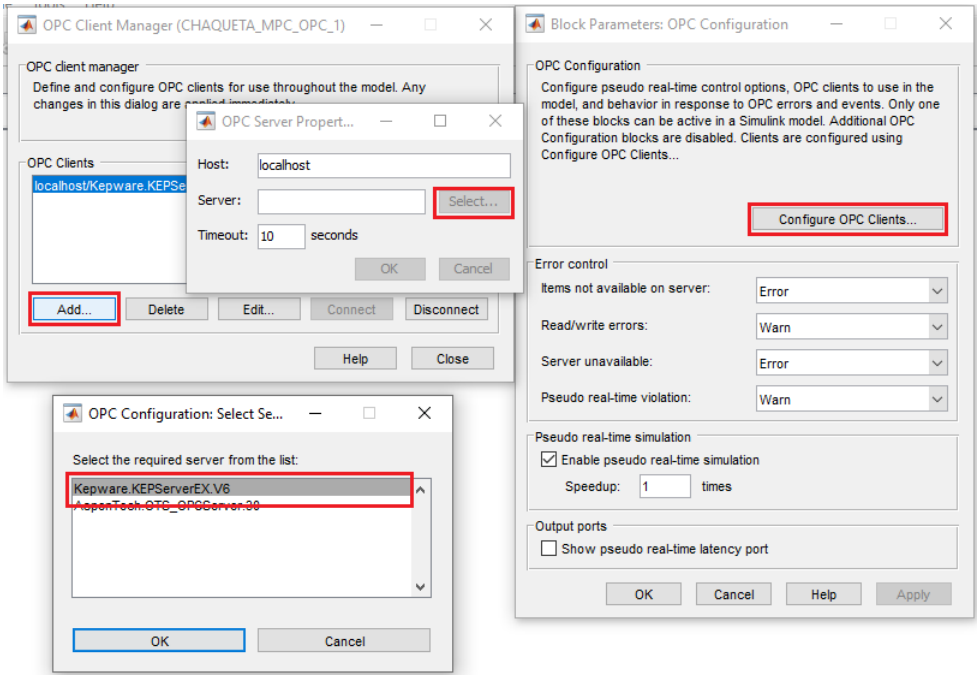
OPC Client Manager (CHAQUETA_MPC_OPC_1)
OPC client manager
Define and configure OPC clients for use throughout the model. Any
OPC Clients
localhost/Kepware.KEPSe
Add...
Delete
Edit...
Connect
Disconnect
Help
Close
OPC Server Propert...
Host:
localhost
Server:
Select...
Timeout:
10
seconds
OK
Cancel
OPC Configuration: Select Se...
Select the required server from the list:
Kepware.KEPServerEX.V6
OK
Cancel
Block Parameters: OPC Configuration
OPC Configuration
Configure pseudo real-time control options, OPC clients to use in the model, and behavior in response to OPC errors and events. Only one of these blocks can be active in a Simulink model. Additional OPC Configuration blocks are disabled. Clients are configured using Configure OPC Clients...
Configure OPC Clients...
Error control
Items not available on server:
Error
Read/write errors:
Warn
Server unavailable:
Error
Pseudo real-time violation:
Warn
Pseudo real-time simulation
Enable pseudo real-time simulation
Speedup:
1
times
Output ports
Show pseudo real-time latency port
OK
Cancel
Help
Apply

Figura A.15

Anexo B

Despliegue código generado sobre PLC

1. Al abrir el software de programación para PLCs de Siemens TIA Portal, lo primero que se debe hacer es crear un proyecto, en el cual, se realizará la importación del controlador generado desde Matlab y su posterior configuración para ser ejecutado en tiempo real (ver figura B.1).

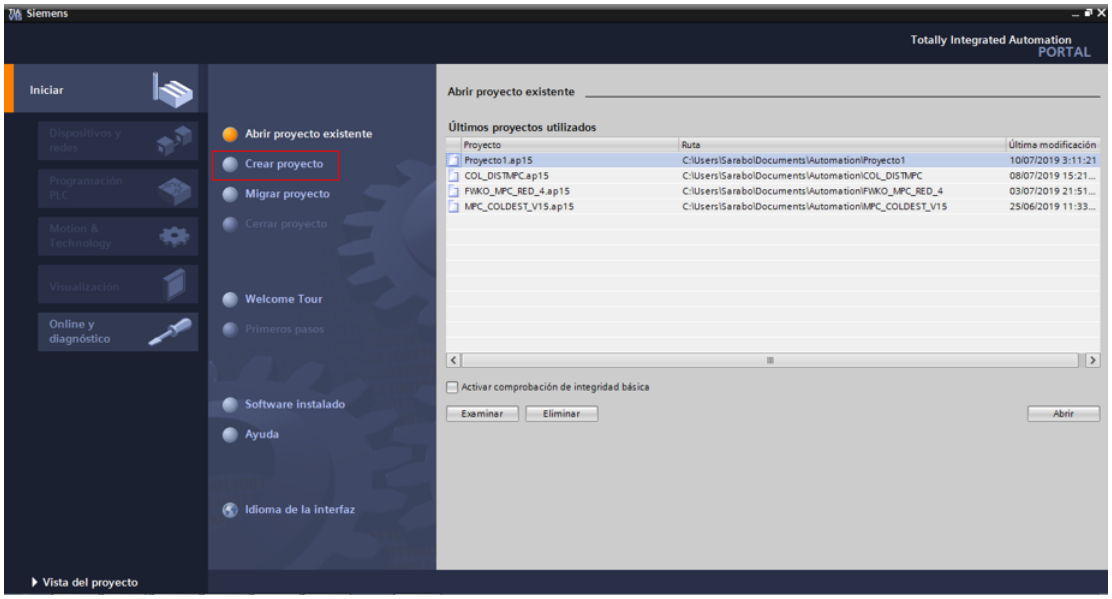

Figura B.1

2. Se deben rellenar los campos que se muestran en la figura B.2 para después crear el proyecto en TIA Portal.

3. Una vez creado el proyecto, el primer paso es configurar el tipo de dispositivo que va a ser usado (ver figura B.3).

4. Para poder buscar el dispositivo que se requiere, se debe agregar primero el mismo (ver figura B.4).

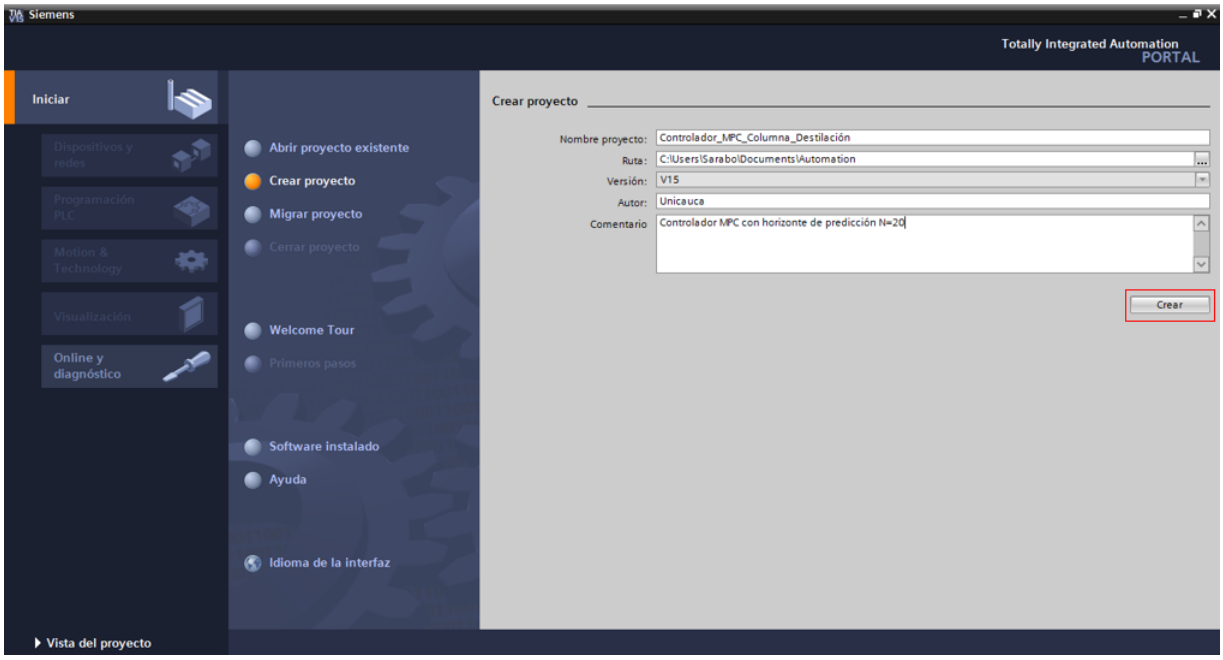

Figura B.2

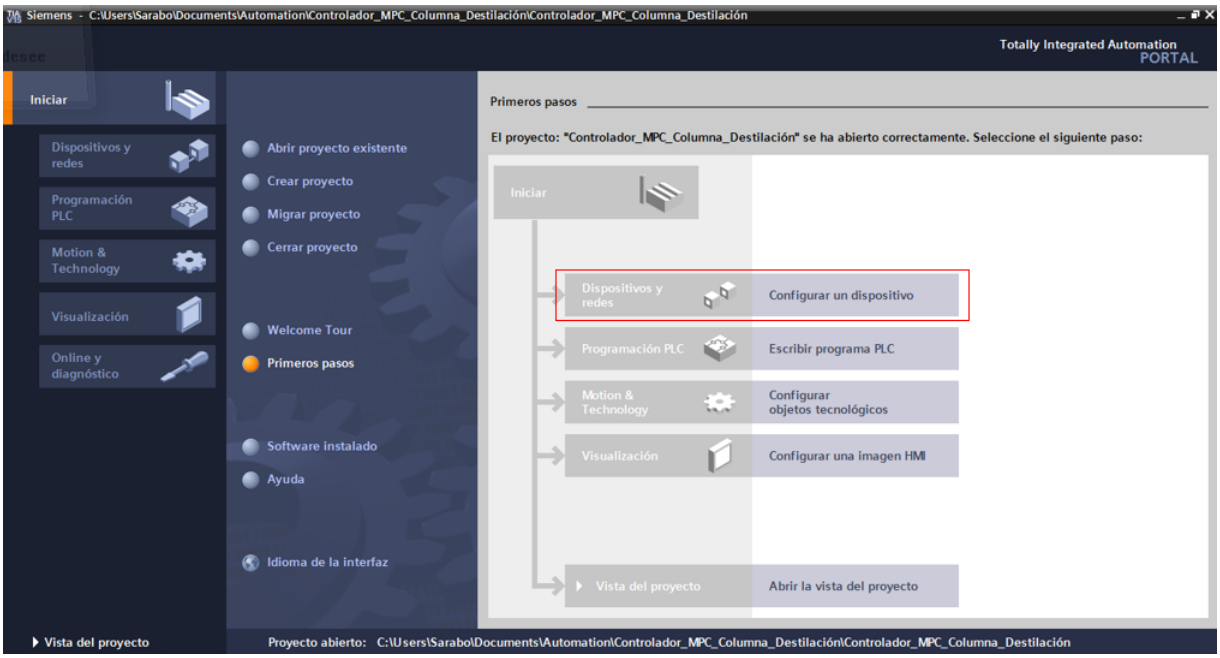

Figura B.3

5. En Siemens, los controladores tienen una CPU y una referencia de se puede encontrar en la parte frontal del PLC. Ambas referencias son importantes para poder configurar el dispositivo de la forma adecuada, a su vez, cada CPU cuenta con una capacidad de memoria y velocidad de operación diferentes, estas caracterís-

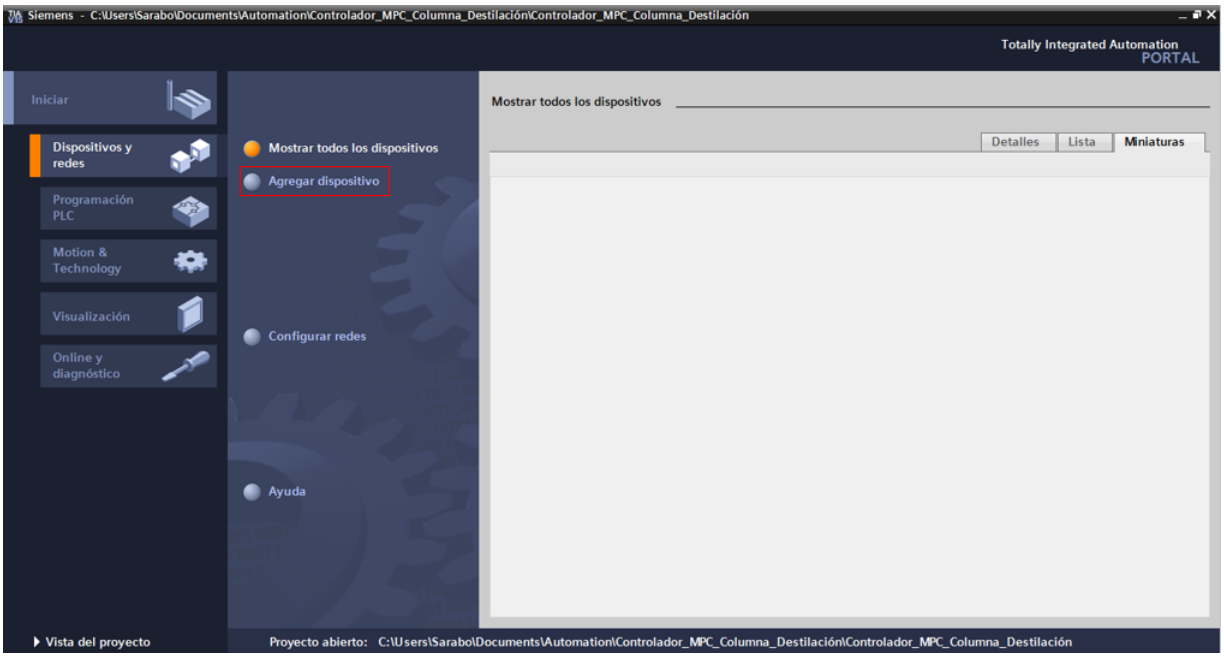

Figura B.4

ticas pueden observarse detalladamente en la parte derecha de la configuración del dispositivo, como se muestra en la figura B.5. Cabe resaltar que en este paso es importante fijarse también de la versión que va a ser utilizada, pues de esto también depende la correcta descarga del algoritmo en el PLC.

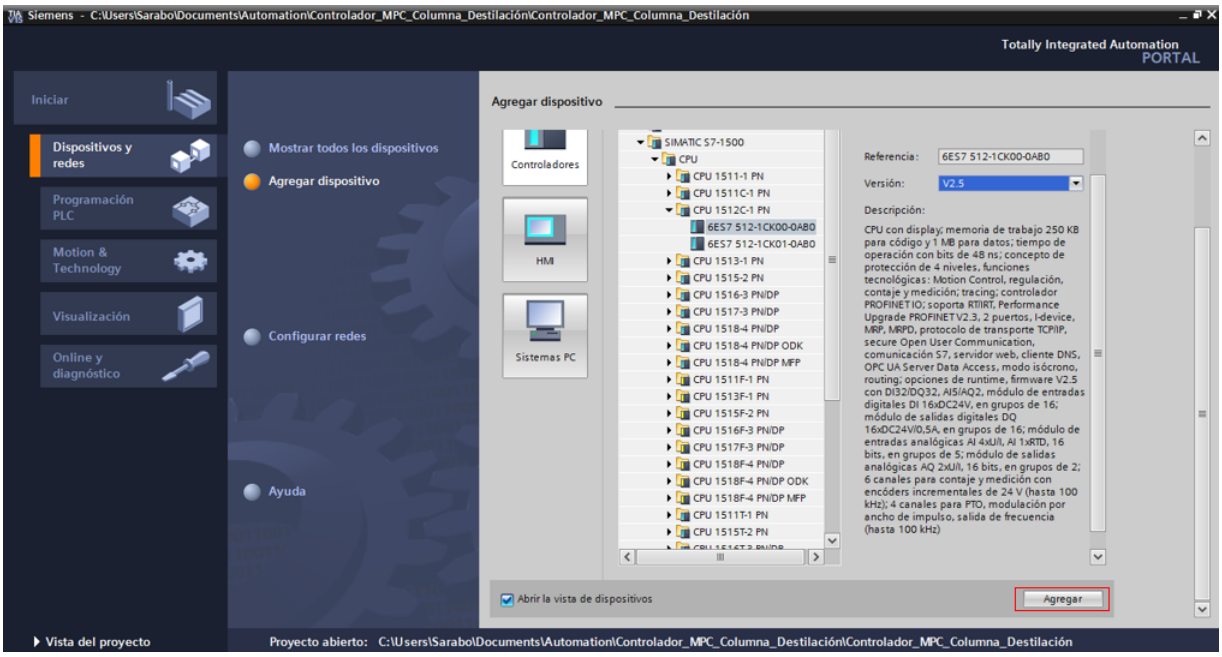

Figura B.5

6. Después de agregar el dispositivo y haber escogido correctamente la CPU de trabajo, se abre un árbol del proyecto y el entorno de trabajo de TIA Portal como se puede observar en la figura B.6. Para poder importar el controlador, en el árbol del proyecto se debe dirigir a “fuentes externas” y, desde ahí, buscar el archivo con extensión .scl, el cuál contiene el algoritmo MPC.

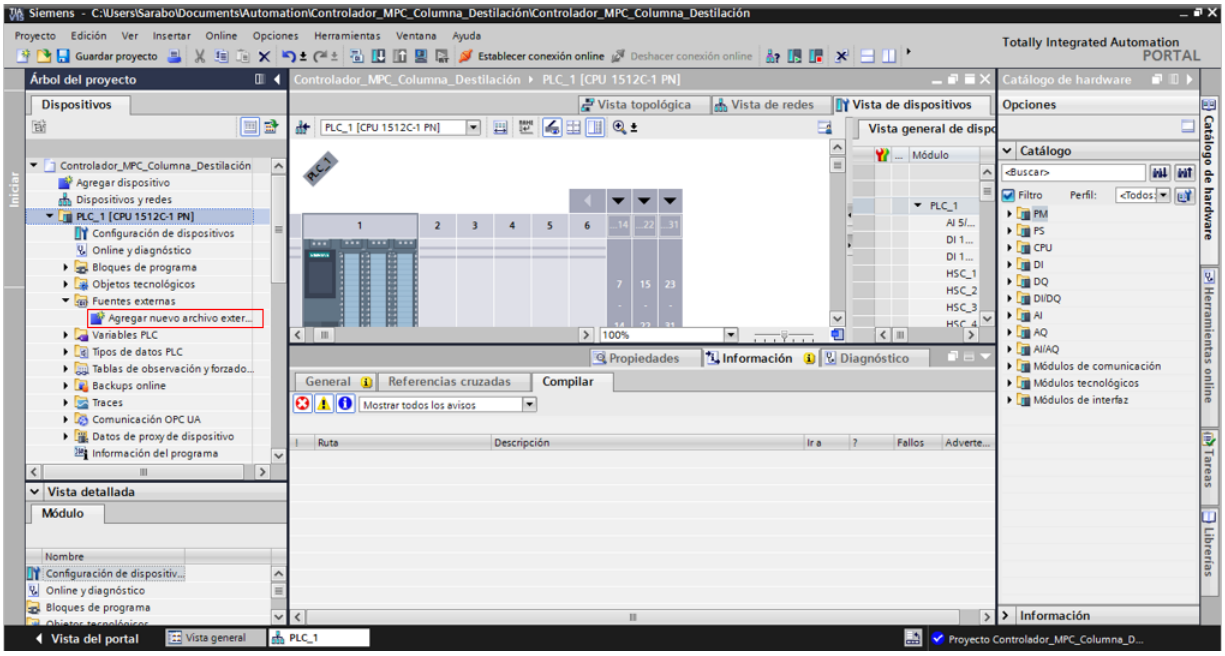

Figura B.6

7. Una vez importado el archivo que contiene el algoritmo de control en código SCL, se deben generar los bloques que permiten que el controlador realice sus operaciones (ver figura B.7).

8. Para poder generar los bloques de programación, se selecciona el archivo y con click derecho se busca la opción “generar bloques a partir de código” como se muestra en la figura B.8.

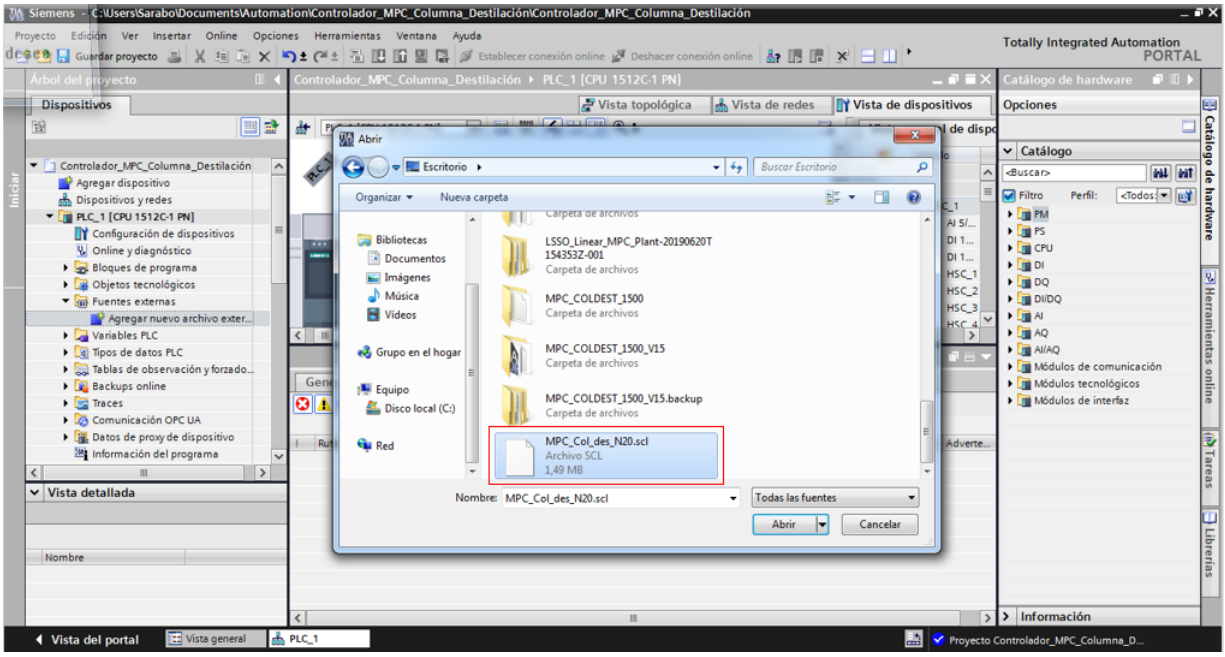

Figura B.7

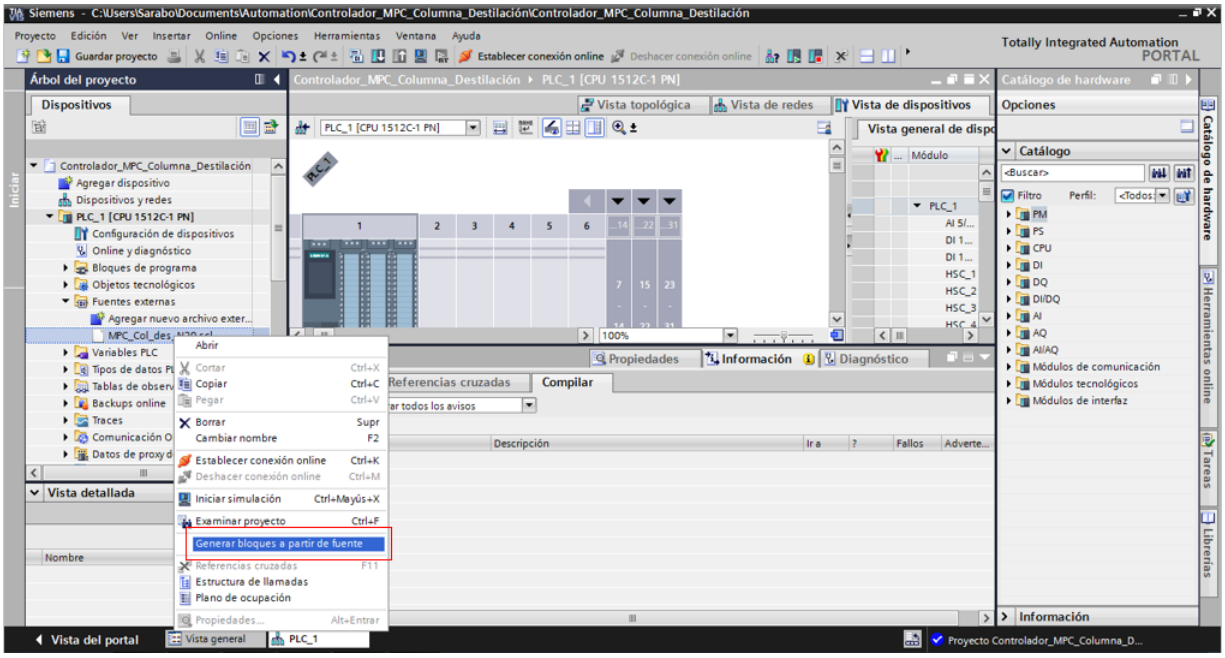

Figura B.8