UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO

FACULTAD DE CIENCIA FÍSICAS Y MATEMÁTICA

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA SCADA CON EL PLC MODICON M340 PARA EL CONTROL PID DE LA PLANTA DE PRESIÓN DE LA ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA DE LA UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO, UTILIZANDO SOFTWARE LIBRE"

TESIS

Para optar por el Título Profesional de INGENIERO ELECTRÓNICO

PRESENTADA POR:

Br. Jhon Alonso Piscoya Siadén

Br. Marlon Javier Vega Zuloeta

ASESOR:

Ing. Victor Olegario Jara Sandoval

LAMBAYEQUE - PERÚ

2015

UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO

FACULTAD DE CIENCIA FÍSICAS Y MATEMÁTICA

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA SCADA CON EL PLC MODICON M340 PARA EL CONTROL PID DE LA PLANTA DE PRESIÓN DE LA ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA DE LA UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO, UTILIZANDO SOFTWARE LIBRE"

TESIS

Para obtar por el Título Profesional de INGENIERO ELECTRÓNICO

PRESENTADA POR:

Br. Jhon Alonso Piscoya Siadén
Br. Marlon Javier Vega Zuloeta

ASESOR:

Ing. Victor Olegario Jara Sandoval

LAMBAYEQUE – PERÚ
2015

DISEÑO E IMPLEMTENTACION DE UN SISTEMA SCADA CON EL PLC MODICON M340 PARA EL CONTROL PID DE LA PLANTA DE PRESIÓN DE LA ESCUELA PROFESIONL DE INGENIERÍA ELECTRÓNICA DE LA UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO, UTILIZANDO SOFTWARE LIBRE.

Elaborado por los bachilleres...

Br. Jhon Alonso Piscoya Siadén
Tesista

Br. Marlón Javier Vega Zuloeta
Tesista

Aprobado por los miembros del jurado...

Ing. Manuel Javier Ramírez Castro
Presidente

Ing. Hugo Javier Chiclayo Padilla
Secretario

Ing. Carlos Leonardo Oblitas Vera
Vocal

Ing. Víctor Olegario Jara Sandoval
Asesor

A mi madre que siempre está presente
para apoyarme en cada paso que doy.

A mi padre y hermanos que son un ejemplo
de superación y de la clase de persona
que quiero llegar a ser.

ALONSO PISCOYA

A Dios por derramar muchas bendiciones
en mí y en mi hermosa familia.

A mis padres, Yolanda y Ricardo, por
todo el amor y comprensión que
me brindan cada día, por su esfuerzo,
sacrificio y sobre todo el apoyo
incondicional en cada momento de mi vida.

A mis hermanos Boris y Gianpier, por su
confianza y por qué son mi motivo de
superación en la vida.

MARLON VEGA

AGRADECIMIENTO

Queremos expresarles nuestros más sinceros agradecimientos a todas las personas que en algún momento del desarrollo del proyecto nos brindaron su apoyo.

Al Ing. Víctor Jara Sandoval, nuestro asesor, quien nos apoyó desde el planteamiento del proyecto, y estuvo presente en cada paso durante el desarrollo del mismo, dando sus consejos y ayudándonos a sobrellevar cada inconveniente que encontramos en el proceso.

Al Ing. David Aguirre Zapata, quien nos apoyó en el rediseño del software libre para el sistema SCADA, brindándonos sus conocimientos en programación, relacionado a códigos de lenguaje JAVA y PHP. A nuestro amigo José Zambrano Campos que nos apoyó durante gran parte de la realización de las pruebas, brindándonos herramientas, conocimientos y apoyo incondicional.

A nuestros familiares y amigos, por estar siempre en los buenos y malos momentos de esta investigación, porque gracias a ellos se logró culminar satisfactoriamente este proyecto de tesis.

LOS AUTORES

UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
Escuela profesional de Ingeniería Electrónica

Diseño e implementación de un sistema SCADA con el PLC MODICON M340 para el control PID de la planta de presión de la Escuela Profesional de Ingeniería Electrónica de la Universidad Nacional Pedro Ruiz Gallo, utilizando software libre.

TESIS

Para optar el Título Profesional de Ingeniero Electrónico

PRESENTADA POR:

Br. Jhon Alonso Piscoya Siadén
Br. Marlon Javier Vega Zuloeta

LAMBAYEQUE – PERÚ

RESUMEN

Este trabajo de tesis se divide en dos partes, la primera es desarrollar un sistema de control para la planta de presión de la escuela profesional de ingeniería electrónica, y la segunda es implementar en la planta de presión un software de monitoreo y control basado en el uso de software libre, con la consigna de que sea compatible con los recursos de la escuela en cuanto a tecnología se refiere.

La planta de presión está constituida por un tanque compresor, un tanque de almacenamiento de agua, una bomba centrífuga controlada por un variador de frecuencia, transmisor de presión, Presóstato calibrado en un rango de 0 a 6 bar, una válvula proporcional que permite el desfogue del agua desde el tanque compresor al tanque de almacenamiento y un manómetro como indicador de presión. Para el proceso se ha elegido usar como controlador a la válvula, dejando el bombeo de agua a una frecuencia constante, y se usa el sensor para cerrar el lazo de control.

Para el diseño del sistema de control primero se requirió determinar el modelo dinámico de la planta de presión a controlar, y mediante simulaciones en el ambiente de MATLAB se logró determinar los parámetros iniciales para 3 tipos de controladores; P, PI, PID, que fueron implementados en las pruebas en tiempo real del sistema.

Para la implementación del software SCADA se plantearon ciertos requerimientos, el principal debe estar basado en el uso de Software libre, es decir, sin licencia alguna y presentar compatibilidad con cualquier tipo de PLC, para que pueda ser usado en distintas aplicaciones, que sea fácil de manipular ya que su finalidad principal es ayudar al aprendizaje de los estudiantes.

Como fin del proyecto, se logró el objetivo, se implementó el software SCADA libre y el OPC libre usados, logrando así contrastar el correcto funcionamiento, en sus distintos controles de la planta de presión de la escuela profesional de Ingeniería Electrónica de la Universidad Nacional Pedro Ruiz Gallo.

UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
Escuela profesional de Ingeniería Electrónica

Design and implementation of a SCADA system with the Modicon M340 for PID control of the plant pressure of the Professional School of Electronic Engineering, National University Pedro Ruiz Gallo, using free software.

THESIS

To obtain the Electronic Engineering Professional Title

BY:

Br. Jhon Alonso Piscoya Siadén
Br. Marlon Javier Vega Zuloeta

LAMBAYEQUE – PERÚ

ABSTRACT

This thesis is divided into two parts, the first is to develop a control system for the pressure plant of the professional school of electronic engineering, and the second is to implement at the pressure plant a monitoring and control software based use of free software, with the motto that is compatible with the resources of the school as technology is concerned.

The pressure plant is comprised of a compressor tank, a storage tank of water, a centrifugal pump controlled by a frecuency inverter, a pressure transmitter, pressure switch is calibrated in a range 0 to 6 bar, a proportional valve which allows the venting water from the compressor to the storage tank and pressure gauge as indicator of pressure. In the process it has chosen to use as a controller to the valve, letting water pumping at a constant frequency, and the sensor is used to close the control loop.

For the design of the control system is first required to determine the dynamic model of the plant pressure control, and through simulations in MATLAB environment was possible to determine the initial parameters for 3 types of controllers; P, PI, PID, which were implemented in the real-time test system.

To implement the software SCADA certain requirements were raised, the principal must be based on the use of free software, that is, without a license and feature compatibility with any type of PLC, so it can be used in different applications, which is easy handling since its main purpose is to help student learning.

As the end of the project, the objective was achieved, free software and free SCADA OPC used was implemented, thus achieving the proper functioning contrast, in its various plant controls pressure professional school of Electronic Engineering, National University Pedro Ruiz Gallo.

TABLA DE CONTENIDOS

LISTA DE FIGURAS

LISTA DE TABLAS

CAPÍTULO 1
INTRODUCCIÓN

1.1. Situación Problemática

En la actualidad el laboratorio de la Escuela Profesional de Ingeniería Electrónica dispone de distintos módulos y equipos para el buen desarrollo de los estudiantes en lo que al área de control e instrumentación se refiere.

Sin embargo, estos módulos son usados de tal manera que no son aprovechados al cien por ciento para el beneficio de los alumnos. Esto debido a que el software y la aplicación entregada en el proyecto realizado por la Escuela Profesional de Ingeniería electrónica con National Instrument, resultó dañado durante su uso, y el hecho de reponerlo tendría un costo muy elevado por la adquisición de nuevas licencias del software. Costo que la escuela profesional de Ingeniería Electrónica no tiene en sus planes solventarlo. Es así que actualmente estos módulos como son la planta de presión, planta de nivel, y otros módulos de entrenamiento en la cual se usa un PLC, son usados para aplicaciones sencillas en la rama de control y para mostrar algunos instrumentos y equipos usados en plantas industriales.

Por lo descrito anteriormente es que resulta la idea de desarrollar una aplicación, usando software libre, con funciones de un sistema SCADA que realice el monitoreo de los procesos que se pueden realizar para fines académicos y que todos los alumnos podrán manejar, explotar y conocer más acerca de la planta de presión, con esto evitaremos la dependencia que ahora tenemos con el software Labview, porque al realizar un sistema SCADA con Labview es necesario tener licencias de funcionamiento, que por lo expuesto anteriormente sería muy costoso adquirirlos.

Es así que surgió la idea de implementar esta aplicación con el PLC MODICON M340, este PLC actualmente trabaja con la planta de presión y así poder realizar con este

software la supervisión y monitoreo de un controlador PID de la planta de presión de dicho laboratorio.

1.2. Justificación del trabajo de Tesis

Esta investigación se desarrollará con la finalidad de implementar un sistema de fácil uso, con una interface amigable para el estudiante, y que esté diseñada para admitir cambios en la programación de acuerdo a la utilización que se le esté pensando dar para fines didácticos. Así también sirve como inicio de trabajos futuros como la exportación de la aplicación SCADA a un servidor web, mediante el cual se pudiera visualizar el comportamiento de la planta en tiempo real, así como modificar algunos parámetros del proceso, desde cualquier lugar con acceso a internet.

Con este software libre, evitaremos la dependencia de licencias que son muy costosas, además el manejo y la interfaz será muy sencilla para todo el alumnado de las EPIE, también se podrá realizar trabajos, practicas sin ningún tiempo límite.

1.3. Objetivos del Trabajo de Tesis

Los objetivos logrados se enmarcan dentro de un objetivo principal y varios secundarios, los cuales se detallan a continuación.

1.3.1. Objetivo Principal

Implementar un sistema SCADA para el control y monitoreo de la planta de presión, mediante el desarrollo de una aplicación utilizando Software libre.

1.3.2. Objetivos Secundarios

Los objetivos secundarios planteados son:

1. Evaluar el funcionamiento del sistema actual del módulo de presión.
2. Implementar un sistema SCADA mediante el uso de software libre.
3. Diseñar controladores P, PI y PID para la planta de presión de la EPIE para que pueda ser usado con fines didácticos.
4. Comparar los diferentes comportamientos de cada controlador, P, PI y PID.
5. Utilizar software libre que emplee el estándar de comunicación OPC.

CAPÍTULO 2
MARCO TEÓRICO

2.1. Sistema de Control Automático

Un sistema de control automático es un conjunto de componentes físicos conectados o relacionados entre sí, de manera que regulen o dirijan su actuación por sí mismos, es decir sin intervención de agentes exteriores (incluido el factor humano), corrigiendo además los posibles errores que se presenten en su funcionamiento.

Actualmente, cualquier mecanismo, sistema o planta industrial presenta una parte actuadora, que corresponde al sistema físico que realiza la acción, y otra parte de mando o control, que genera las órdenes necesarias para que esa acción se lleve o no a cabo.

En Automática se sustituye la presencia del ser humano por un mecanismo, circuito eléctrico, circuito electrónico o, más modernamente por un ordenador. El sistema de control será, en este caso automático.

Un ejemplo sencillo de sistema automático lo constituye el control de temperatura de una habitación por medio de un termostato, en el que se programa una temperatura de referencia que se considera idónea. Si en un instante determinado la temperatura del recinto es inferior a la deseada, se producirá calor, lo que incrementará la temperatura hasta el valor programado, momento en que la calefacción se desconecta de manera automática.

2.1.1. Conceptos Básicos de Control Automático

- *Variables del sistema* : son todas las magnitudes, sometidas a vigilancia y control, que definen el comportamiento de un sistema (Presión, velocidad, temperatura, posición, etc).
- *Entrada* : es la excitación que se aplica a un sistema de control desde una fuente de energía externa, con el fin de provocar una respuesta.
- *Salida* : es la respuesta que proporciona el sistema de control.
- *Perturbación* : son las señales no deseadas que influyen de forma adversa en el funcionamiento del sistema.
- *Planta* : sistema sobre el que pretendemos actuar.
- *Sistema* : es un conjunto de elementos interrelacionados capaces de realizar una operación dada o de satisfacer una función deseada.
- *Señal de referencia* : (set point) es una señal de entrada conocida que nos sirve para calibrar al sistema.
- *Señal de error* : representa la diferencia entre la señal de entrada y la realimentada.
- *Unidad de realimentación* : está formada por uno o varios elementos que captan la variable de salida, la acondicionan y trasladan a la unidad de comparación.
- *Actuador* : es un elemento que recibe una orden desde el regulador o controlador y la adapta a un nivel adecuado según la variable de salida necesaria para accionar el elemento final de control, planta o proceso.
- *Transductor* : transforma una magnitud física en otra que es capaz de interpretar el sistema.
- *Amplificador* : nos proporciona un nivel de señal procedente de la realimentación, entrada, comparador, etc, adecuada al elemento sobre el que actúa.

2.1.2. Representación de un Sistema de Control

Un proceso o sistema de control es un conjunto de elementos interrelacionados capaces de realizar una operación dada o de satisfacer una función deseada. Los sistemas de control se pueden representar en forma de diagramas de bloques, en los que se ofrece una expresión visual y simplificada de las relaciones entre la entrada y la salida de un sistema físico. A cada componente del sistema de control se le denomina elemento, y se representa por medio de un rectángulo. El diagrama de bloques más sencillo es el bloque simple, que consta de una sola entrada y de una sola salida, la interacción entre los bloques se representa por medio de flechas que indican el sentido de flujo de la información, como se muestra en la Figura 2.1.

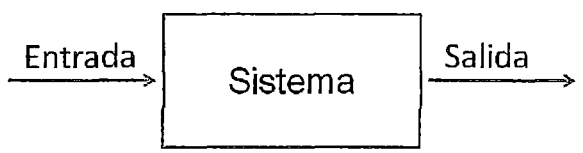


Figura 2.1: Representación de un Sitema de Control Simple.

En estos diagramas es posible realizar operaciones de adición y de sustracción, que se representan por un pequeño círculo en el que la salida es la suma algebraica de las entradas con sus signos. También se pueden representar las operaciones matemáticas de multiplicación y división como muestra la Figura 2.2.

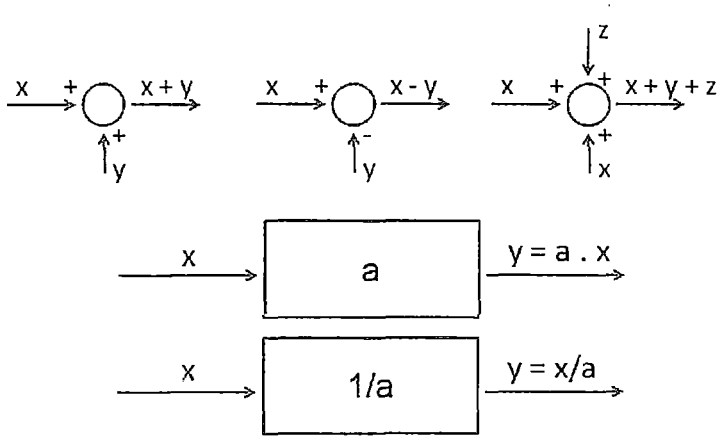


Figura 2.2: Diagramas Operacionales en un Sitema de Control.

2.1.3. Tipos de Control Automático

Los sistemas de regulación se pueden clasificar en:

Sistemas de Lazo Abierto

Un sistema de control en lazo o bucle abierto es aquel en el que la señal de salida no influye sobre la señal de entrada. La exactitud de estos sistemas depende de su calibración, de manera que al calibrar se establece una relación entre la entrada y la salida con el fin de obtener del sistema la exactitud deseada, el diagrama de bloques de un sistema en lazo abierto se muestra en la Figura 2.3.

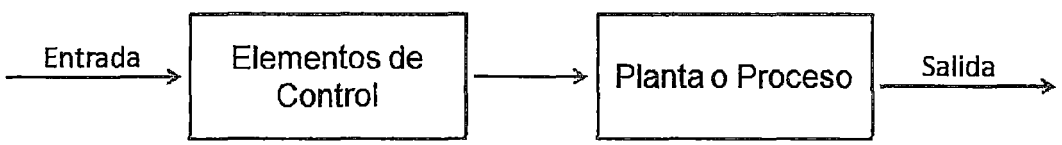


Figura 2.3: Diagrama de bloque de un sistema en lazo abierto.

El sistema se controla mediante un transductor y un actuador. El esquema típico del sistema se muestra en la Figura 2.4, en este caso:

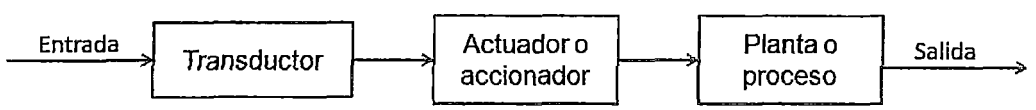


Figura 2.4: Diagrama de bloques de un sistema controlado por el actuador.

El transductor modifica o adapta la naturaleza de la señal de entrada al sistema de control. El actuador o accionador modifica la entrada del sistema entregada por el transductor (normalmente amplifica la señal). Los sistemas de lazo abierto dependen de la variable tiempo y la salida no depende de la entrada.

El principal inconveniente que presentan los sistemas de lazo abierto es que son extremadamente sensibles a las perturbaciones. Por ejemplo si en una habitación se ha

conseguido una temperatura idónea y se abre una puerta o ventana (perturbación) entraría aire frío, de manera que el tiempo necesario para obtener dicha temperatura sería diferente.

Si en un sistema en lazo abierto existen perturbaciones, no se obtiene siempre la variable de salida deseada. Conviene, por tanto, utilizar un sistema en el que haya una relación entre la salida y la entrada.

Sistemas de Lazo Cerrado

Un sistema de control de lazo cerrado es aquél en el que la acción de control es, en cierto modo, dependiente de la salida. La señal de salida influye en la entrada. Para esto es necesaria que la entrada sea modificada en cada instante en función de la salida. Esto se consigue por medio de lo que llamamos realimentación o retroalimentación (feedback).

La realimentación es la propiedad de un sistema en lazo cerrado por la cual la salida (o cualquier otra variable del sistema que esté controlada) se compara con la entrada del sistema o una de sus entradas, de manera que la acción de control se establezca como una función de ambas. A veces también se le llama a la realimentación transductor de la señal de salida, ya que mide en cada instante el valor de la señal de salida y proporciona un valor proporcional a dicha señal. Por lo tanto podemos definir también los sistemas de control en lazo cerrado como aquellos sistemas en los que existe una realimentación de la señal de salida, de manera que ésta ejerce un efecto sobre la acción de control. El diagrama de bloques correspondiente a un sistema de control en lazo cerrado se muestra en la Figura 2.5.

El controlador está formado por todos los elementos de control y a la planta también se le llama proceso.

En la Figura 2.5, se observa cómo la salida es realimentada hacia la entrada. Ambas se comparan, y la diferencia que existe entre la entrada, que es la señal de referencia o consigna (señal de mando), y el valor de la salida (señal realimentada) se conoce como error o señal de error. La señal que entrega el controlador se llama señal de control o manipulada y la entregada por la salida, señal controlada.

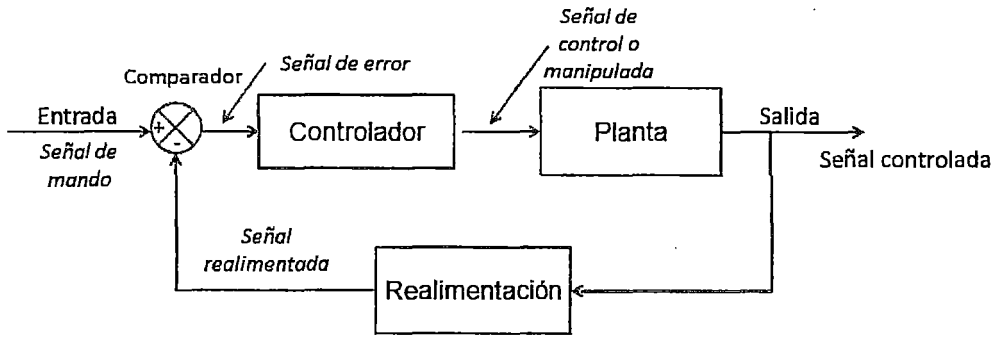


Figura 2.5: Diagrama de bloques de un sistema en lazo cerrado.

El error, o diferencia entre los valores de la entrada y de la salida, actúa sobre los elementos de control en el sentido de reducirse a cero y llevar la salida a su valor correcto. Se intenta que el sistema siga siempre a la señal de consigna. El diagrama de bloques anterior se puede sustituir por el esquema de la Figura 2.6.

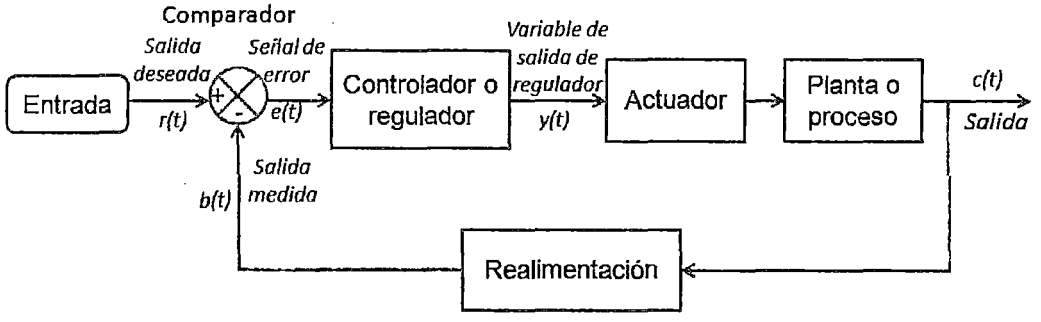


Figura 2.6: Diagrama de un Sistema en lazo cerrado completo.

La salida del sistema de regulación se realimenta mediante un captador o sensor. En el comparador o detector de error, la señal de referencia (salida del transductor) se compara con la señal de salida medida por el sensor, con lo que se genera la siguiente señal de error:

$$e(t) = r(t) - b(t) \tag{2.1}$$

Donde:

$e(t)$ es la señal de error

$r(t)$ es la señal de referencia

$b(t)$ es la variable realimentada

Pueden suceder dos casos:

1. Que la señal de error sea nula. En este caso la salida tendrá exactamente el valor previsto.
2. Que la señal de error no sea nula. Esta señal de error actúa sobre el elemento regulador que a su salida proporciona una señal que, a través del elemento accionador, influye en la planta o proceso para que la salida alcance el valor previsto y de esta manera el valor se anule.

El regulador o controlador es el elemento que determina el comportamiento del bucle, por lo que debe ser un componente diseñado con gran precisión. Es el cerebro del bucle de control.

Mientras que la variable controlada se mantenga en el valor previsto, el regulador no actuará sobre el elemento accionador. Pero si el valor de la variable se aleja del prefijado, el regulador modifica su señal, ordenando al accionador que actúe sobre la planta o proceso, en el sentido de corregir dicho alejamiento.

Los sistemas en lazo cerrado son mucho menos sensibles a las perturbaciones que los de lazo abierto, ya que cualquier modificación de las condiciones del sistema afectará a la salida, pero este cambio será registrado por medio de la realimentación como un error que es en definitiva la variable que actúa sobre el sistema de control. De este modo, las perturbaciones se compensan, y la salida se independiza de las mismas.

2.2. Función de Transferencia

Se define función de transferencia $G(s)$ de un sistema como el cociente entre las transformadas de Laplace de las señales de salida y entrada, como se muestra en la Figura 2.7.

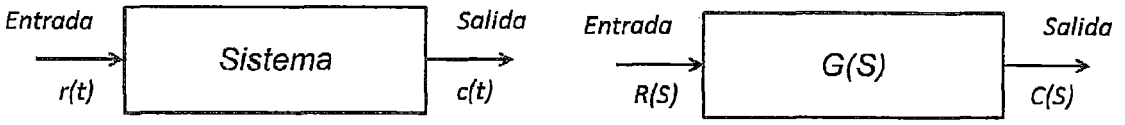


Figura 2.7: Función de Transferencia, dominio del Tiempo (Izquierda), dominio Complejo (Derecha).

La *función de transferencia* $G(s)$ del sistema será:

$$G(s) = \frac{C(s)}{R(s)} \tag{2.2}$$

Las características de la función de transferencia dependen únicamente de las propiedades físicas de los componentes del sistema, no de la señal de entrada aplicada.

A través del concepto de función de transferencia, se puede conocer de forma sencilla:

- Cómo va a comportarse el sistema en cada situación: según la entrada que se produzca en el sistema, sabremos cuál será la respuesta de salida.
- La estabilidad del mismo: saber si la respuesta del sistema se mantendrá siempre dentro de unos límites determinados, o llegará en algún momento a ser inestable.
- Qué valores se podrán aplicar a ciertos parámetros del sistema de manera que éste sea estable.

La función de transferencia viene dada como el cociente de dos polinomios en la variable compleja s de Laplace, uno, $N(s)$ (numerador) y otro $D(s)$ (denominador).

$$G(s) = \frac{N(s)}{D(s)} = \frac{b_0 s^m + b_1 s^{m-1} + ... + b_{m-1} s + b_m}{a_0 s^n + a_1 s^{n-1} + ... + a_{n-1} s + a_n} \tag{2.3}$$

La función de transferencia es muy útil para, una vez calculada la transformada de Laplace de la entrada, conocer de forma inmediata la transformada de Laplace de la salida.

Calculando la trasformada inversa se obtiene la respuesta en el tiempo del sistema ante esa entrada determinada.

2.2.1. Polos y Ceros de la Función de Transferencia

El denominador de la función de transferencia, $D(s)$, se conoce como función característica, pues determina, a través de los valores de sus coeficientes, las características físicas de los elementos que componen el sistema.

La función característica igualada a cero se conoce como ecuación característica del sistema:

$$D(s) = a_0 s^n + a_1 s^{n-1} + ... + a_{n-1} s + a_n = 0 \tag{2.4}$$

Las raíces de la ecuación característica se denominan polos del sistema. Las raíces del numerador $N(s)$ reciben el nombre de ceros del sistema.

Se puede demostrar que para que un sistema sea físicamente realizable, el número de polos debe ser mayor, o al menos igual, que el número de ceros. Si fuese al contrario, esto implicaría que el sistema responde antes de que se produzca el estímulo, lo cual es físicamente imposible.

2.2.2. Estabilidad de un Sitema

Un sistema estable es aquél que permanece en reposo a no ser que se excite por una fuente externa, en cuyo caso alcanzará de nuevo el reposo una vez que desaparezcan todas las excitaciones.

Para que un sistema sea estable, las raíces de la ecuación característica o polos deben estar situadas en el lado izquierdo del semiplano complejo de Laplace, como muestra la Figura 2.8.

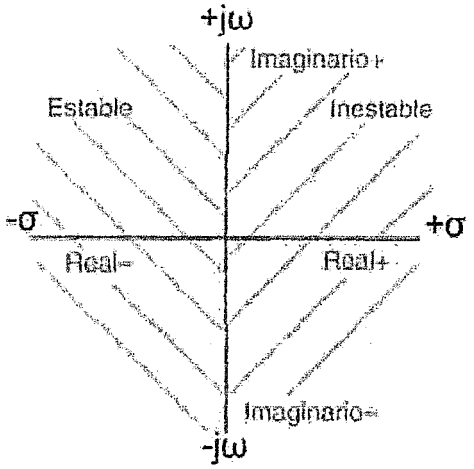


Figura 2.8: Estabilidad de un sistema, mediante la ubicación de sus raíces.

Los polos situados en el origen o sobre el eje imaginario dan lugar a respuestas continuas o constantes que se consideran inestables. Los polos en la parte derecha del plano complejo dan lugar a respuestas que crecen con el tiempo y por lo tanto son inestables. Se dice que un sistema de control es estable cuando aplicando a su entrada una señal Delta de Dirac $\delta(t)$, en la salida aparece una senal decreciente en el tiempo que se hace cero cuando el tiempo tiende a infinito, como se muestra en la Figura 2.9.
Donde:

1. Amortiguamiento exponencial.
2. Sinusoide amortiguada exponencialmente.

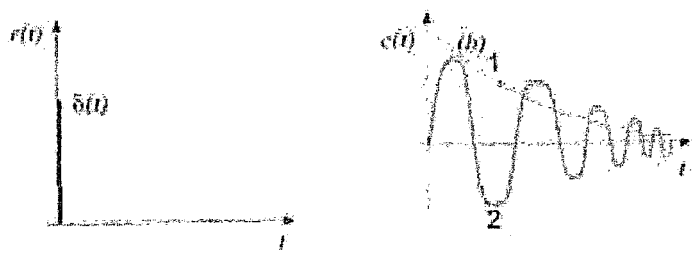


Figura 2.9: Estabilidad de un sistema, señal Delta de Dirac (Izquierda), señal decreciente en el Tiempo como respuesta a la señal Delta de Dirac (Derecha).

2.2.3. Tipos de Sistemas Según sus Polos

Se denomina orden de un sistema al correspondiente a su función característica. Según esto nos podemos encontrar:

1. *Sistemas de orden cero* : su función de transferencia no tiene ningún polo.
2. *Sistemas de primer orden* : su función de transferencia tiene un polo.
3. *Sistemas de segundo orden* : su función de transferencia tiene dos polos.
4. *Sistemas de orden superior* : su función de transferencia tiene más de dos polos.

2.3. Regulador o Controlador

Antiguamente el control de los procesos industriales se llevaba a cabo de manera manual, el propio operario realizaba los cambios adecuados en el sistema para obtener los resultados finales deseados. Hoy en día, muchas aplicaciones automáticas utilizan el computador como elemento de control.

El controlador o regulador constituye el elemento fundamental en un sistema de control, pues determina el comportamiento del bucle, ya que condiciona la acción del elemento actuador en función del error obtenido.

La forma en que el regulador genera la señal de control se denomina acción de control. Algunas de estas acciones se conocen como acciones básicas de control, mientras que otras se pueden presentar como combinaciones de las acciones básicas, en la Tabla 2.1 se muestra las acciones básicas usadas en la investigación.

ACCIONES BÁSICAS	COMBINACIÓN DE ACCIONES BÁSICAS
Proporcional(P)	Proporcional - Integral(PI)
Derivador(D)	Proporcional - Derivador(PD)
Integrador(I)	Proporcional - Integral - Derivador(PID)

Tabla 2.1: Combinaciones de acciones básicas.

Controlador de acción Proporcional (P)

En este regulador la señal de accionamiento es proporcional a la señal de error del sistema. Si la señal de error es grande, el valor de la variable regulada es grande y si la señal de error del sistema es pequeña, el valor de la variable regulada es pequeño.

Es el más simple de todos los tipos de control y consiste simplemente en amplificar la señal de error antes de aplicarla a la planta o proceso. La función de transferencia de este tipo de control se reduce a una variable real, denominada K_p que determinará el nivel de amplificación del elemento de control.

$$y(t) = K_p * e(t) \tag{2.5}$$

La función de transferencia entre la salida del controlador $y(t)$ y la señal de error $e(t)$, pasando al dominio de Laplace es:

$$\frac{Y(s)}{E(s)} = K_p \tag{2.6}$$

Donde:
K_p se le denomina ganancia proporcional.

Otro parámetro importante en la acción de este controlador, es la denominada banda proporcional que expresa que tan grande será la acción de control ante una señal de error en la entrada, y es igual a:

$$\frac{1}{K_p} = BP \tag{2.7}$$

En la Figura 2.10 se muestra la representación gráfica de un controlador proporcional.

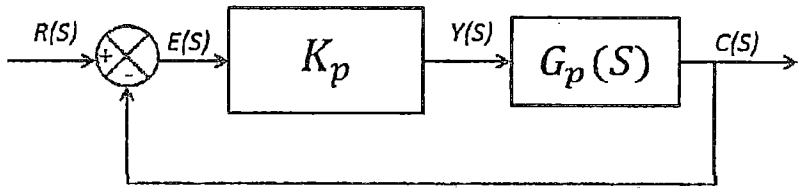


Figura 2.10: Representación gráfica de un controlador Proporcional.

Controlador de acción Proporcional - Integral (PI)

En la práctica no existen controladores que tengan sólo acción integral sino que llevan combinada una acción proporcional. Estas dos acciones se complementan. La primera en actuar es la acción proporcional (instantáneamente) mientras que la integral actúa durante un intervalo de tiempo. Así y por medio de la acción integral se elimina la desviación remanente (proporcional).

La salida del bloque de control PI responde a la ecuación:

$$y(t) = K_p * e(t) + \frac{K_p}{T_i} \int_0^t e(t)dt \tag{2.8}$$

Donde K_p y T_i son parámetros ajustables del sistema. A T_i se le denomina tiempo integral y controla la acción integral del sistema, mientras K_p controla ambas. Si T_i es muy grande la pendiente de la rampa, correspondiente a la acción integral será pequeña, y por tanto, el efecto de esta acción suave, y viceversa. Analizando el sistema en el dominio de Laplace:

$$\frac{Y(S)}{E(S)} = K_p(1 + \frac{1}{T_i S}) \tag{2.9}$$

Donde:

K_p se le denimona Ganancica proporcional.

T_i se le denomina Tiempo integral.

En la Figura 2.11 se muestra la representación gráfica de un controlador proporcional integral.

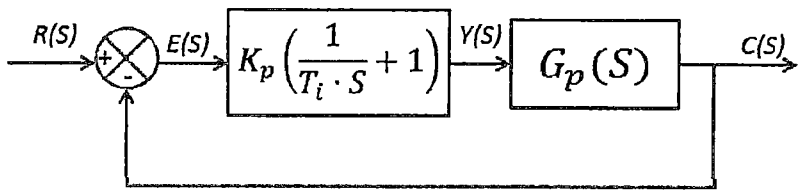


Figura 2.11: Representación gráfica de un controlador Proporcional - Integral.

Controlador de acción Proporcional - Integral - Derivativo (PID)

Aprovecha las características de los reguladores anteriores, de forma, que si la señal de error varía lentamente en el tiempo, predomina la acción proporcional e integral y, si la señal de error varía rápidamente, predomina la acción derivativa. Tiene la ventaja de tener una respuesta más rápida y una inmediata compensación de la señal de error en el caso de cambios o perturbaciones. Tiene como desventaja que el bucle de regulación es más propenso a oscilar y los ajustes son más difíciles de realizar.

La salida del regulador viene dada por la siguiente ecuación:

$$y(t) = K_p * e(t) + \frac{K_p}{T_i}\int_0^t e(t)dt + K_p * T_d\frac{de(t)}{dt} \tag{2.10}$$

En el dominio de Laplace:

$$\frac{Y(S)}{E(S)} = K_p(1 + T_dS + \frac{1}{T_iS}) \tag{2.11}$$

Donde:

K_p se le denimona Ganancica proporcional.

T_i se le denomina Tiempo integral.

T_d se le denomina Tiempo derivativo.

Donde también se puede concluir:

$$K_i = K_p(\frac{1}{T_i}) \tag{2.12}$$

$$K_d = K_pT_d \tag{2.13}$$

Donde:

K_i se le denomina Ganancia integral.

K_d se le denomina Ganancia derivativo.

Al utilizar los tres términos simultáneamente logramos que el derivativo suavice los sobre picos introducidos principalmente por el término integral, sin renunciar a la eliminación del error estacionario.

En la Figura 2.12 se muestra la representación gráfica del controlador proporcional integral derivativo.

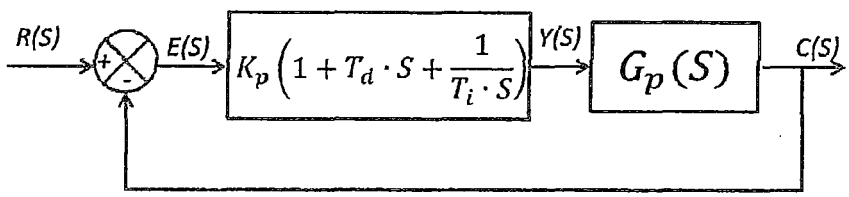


Figura 2.12: Representación gráfica de un controlador Proporcional - Integral - Derivativo.

Las funciones que realizan las acciones de control en un PID, son las que se muestran en la Tabla 2.2:

Acción de Control	Tiempo de subida	Sobrepico	Tiempo de establecimiento	Error estacionario
$Si\ K_p\ disminuye$	Disminuye	Aumenta	Cambia Poco	Disminuye
$Si\ K_i\ disminuye$	Disminuye	Aumenta	Aumenta	Eliminado
$Si\ K_d\ disminuye$	Cambia poco	Disminuye	Disminuye	Cambia poco

Tabla 2.2: Funciones de las Constantes de las acciones de Control.

2.3.1. Métodos de Sintonización de Controladores

El primer paso para aplicar un controlador consiste en elegir el tipo adecuado (P, PI, PID), para lo cual se necesita comprender el efecto de las tres acciones y, a ser posible, tener experiencia sobre el proceso a controlar. El segundo paso es ajustar los parámetros para que la respuesta del sistema se ajuste a unas determinadas especificaciones.

El ajuste de parámetros es frecuente en procesos industriales, no sólo en los trabajos de puesta en marcha, sino también cuando se detectan cambios sustanciales del comportamiento del proceso. Si se puede obtener un modelo matemático de una planta, es posible aplicar diversas técnicas de diseño con el fin de determinar los parámetros del

controlador que cumpla las especificaciones en estado transitorio y en estado estable del sistema en lazo cerrado. Sin embargo, si la planta es tan complicada que no es fácil obtener su modelo matemático, tampoco es posible un enfoque analítico para el diseño de un controlador PID. Las técnicas experimentales están especialmente orientadas al mundo industrial, donde existen grandes dificultades para obtener una descripción matemática, las técnicas o métodos de ajustes son:

Método Ziegler - Nichols

Ziegler y Nichols propusieron unas reglas para determinar los valores de la ganancia proporcional K_p, del tiempo integral T_i y del tiempo derivativo T_d, con base en las características de respuesta transitoria de una planta especifica.

Existen dos métodos denominados reglas de sintonización de Ziegler - Nichols. En ambos se pretende obtener un 25 % de sobrepico máximo en la respuesta escalón, como muestra la Figura 2.13:

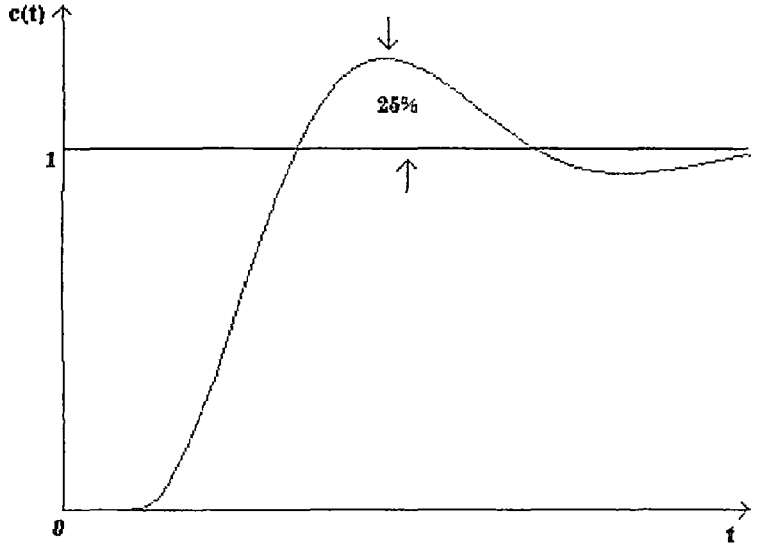


Figura 2.13: Sobre pico ideal por el método de Ziegler - Nichols.

1. **Lazo Abierto:** En el primer método, la respuesta de la planta a una entrada escalón unitario se obtiene de manera experimental. Si la planta no contiene integradores

ni polos dominantes complejos conjugados, la curva de respuesta escalón unitario puede tener forma de S, como se observa en la Figura 2.14. Si la respuesta no exhibe una curva con forma de S, este método no es pertinente. Tales curvas de respuesta escalón se generan experimentalmente o a partir de una simulación dinámica de la planta.

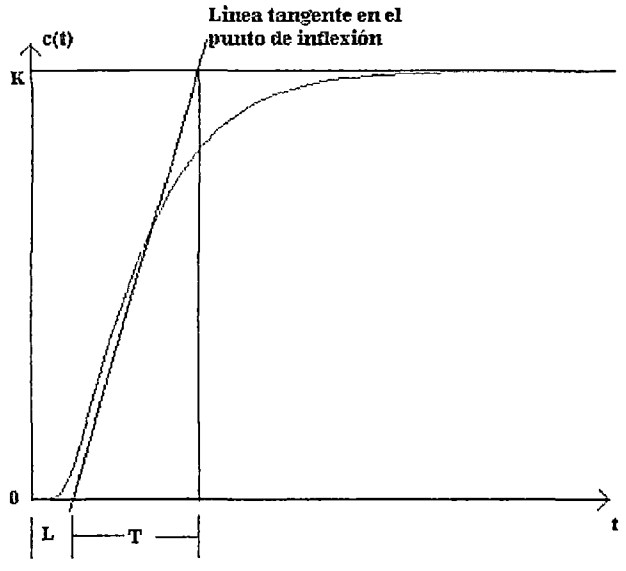


Figura 2.14: Respuesta al escalón unitario en lazo abierto.

La curva con forma de S se caracteriza por dos parámetros: el tiempo de retardo L y la constante de tiempo T. El tiempo de retardo y la constante de tiempo se determinan dibujando una recta tangente en el punto de inflexión de la curva con forma de S y determinando las intersecciones de esta tangente con el eje del tiempo y la línea $c(t) = K$, como se aprecia en la Figura 2.14. En este caso, la función de transferencia $C(s)/U(s)$ se aproxima mediante un sistema de primer orden con un retardo de transporte del modo siguiente:

$$\frac{C(s)}{U(s)} = \frac{Ke^{-Ls}}{Ts+1} \tag{2.14}$$

Las técnicas en lazo abierto se suelen usar sólo en procesos lentos tales como control de temperatura, mientras que en procesos rápidos (caudal, presión) el enfoque en lazo cerrado es mucho más rápido y seguro.

En la Tabla 2.3 se muestran los parámetros que Ziegler - Nichols propuso para los distintos controladores.

Tipo de Controlador	K_p	T_i	T_d
P	T/L	∞	0
PI	$0{,}9\,T/L$	$L/0{,}3$	0
PID	$1{,}2\,T/L$	$2L$	$0{,}5L$

Tabla 2.3: Valores de los distintos controladores para Lazo Abierto.

2. **Lazo Cerrado:** En el segundo método, primero establecemos $T_i = \infty$ y $T_d = 0$. Usando sólo la acción de control proporcional, se incrementa K_p de 0 a un valor crítico K_{cr} en donde la salida exhiba primero oscilaciones sostenidas. Por tanto, la ganancia crítica K_{cr} y el periodo P_{cr} correspondiente se determinan experimentalmente. Si la salida no presenta oscilaciones sostenidas para cualquier valor que pueda tomar K_p, no se aplica este método.

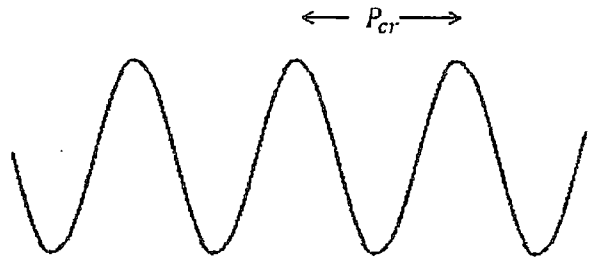


Figura 2.15: Oscilaciones sostenidas del método de Ziegler - Nichols.

Donde:

K_{cr} se le denomina ganancia crítica.

$P_{cr} = \frac{2\pi}{W_{cr}}$

W_{cr} se le denomina frecuencia de oscilaciones sostenidas.

P_{cr} se le denomina Peiíodo Crítico.

Ziegler y Nichols sugirieron que se establecieran los valores de los parámetros K_p, T_i y T_d de acuerdo con la fórmula que aparece en la Tabla 2.4.

Tipo de Controlador	K_p	T_i	T_d
P	$0{,}5K_{cr}$	∞	0
PI	$0{,}45\ K_{cr}$	$P_{cr}/1{,}2$	0
PID	$0{,}6\ K_{cr}$	$0{,}5\ P_{cr}$	$0{,}125\ P_{cr}$

Tabla 2.4: Valores de los distintos controladores para Lazo Cerrado.

Método de Aström - Hägglund

Este método incorpora un relé (Relay) que provoca oscilaciones controladas en el proceso que permiten la identificación de características dinámicas del mismo, ver Figura 2.16.

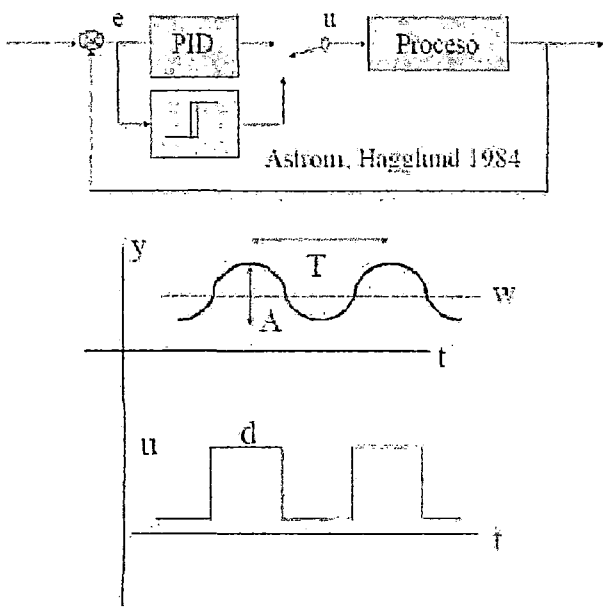


Figura 2.16: Método de sintonización de Aström - Hägglund.

Para el desarrollo de este método se utiliza Matlab como herramienta de simulación para establecer los parámetros A y T del sistema $G(s)$ al introducir el relé con amplitud d dentro del lazo cerrado.

Para calcular las constantes K_p, T_i y T_d del controlador PID, se mide la amplitud de la oscilación de salida A y el periodo del mismo P_{cr}, Se determina el valor de K_{cr} dado por la siguiente ecuación:

$$K_{cr} = \frac{4d}{\pi a} \tag{2.15}$$

Después se utilizan las fórmulas usadas para Ziegler - Nichols en lazo cerrado, ver Tabla 2.4.

Método de Kayser - Rajka

Este método trabaja similar al método de Aström - Hägglund, solo que incorpora un retardo (Transport Delay) después del relé (Relay). Esto también provoca oscilaciones controladas en el proceso que permiten la identificación de características dinámicas del proceso.

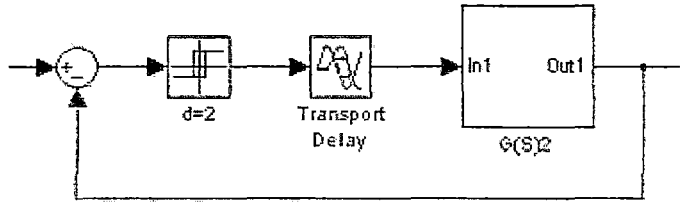


Figura 2.17: Método de sintonización de Kaysar - Rajka.

Para este método se busca obtener una oscilación controlada de salida, proporcional a los valores de amplitud del relé d y el valor del retardo Td, para que se cumpla el siguiente criterio:

$$\Phi = \frac{360 T_d}{T_c} \tag{2.16}$$

Donde T_c es el periodo de oscilación de salida del sistema. Para un valor de Φ = 10, se debe encontrar el valor de T_d (transport delay), para que cumpla el criterio:

$$T_c = 36 T_d \tag{2.17}$$

Luego calcular K_{cr}, ver Ecuación 2.15, y después aplicamos las fórmulas de la Tabla 2.4 para encontrar los parámetros.

Método del Lugar Geométrico de las Raíces

El método del lugar geométrico de las raíces es un enfoque gráfico que permite determinar las ubicaciones de todos los polos en lazo cerrado a partir de las ubicaciones de los polos y ceros en lazo abierto conforme algún parámetro (por lo general la ganancia) varía de cero a infinito. El método produce un indicio claro de los efectos del ajuste del parámetro.

El diseño por el método del lugar de las raíces se basa en redibujar el lugar de las raíces del sistema añadiendo polos y ceros a la función de transferencia en lazo abierto del sistema y hacer que el lugar de las raíces pase por los polos en lazo cerrado deseados en el plano S.

El enfoque del lugar geométrico de las raíces es muy poderoso en el diseño cuando se incorporan las especificaciones en términos de las cantidades en el dominio del tiempo, tales como el factor de amortiguamiento relativo y la frecuencia natural no amortiguada de los polos dominantes en lazo cerrado, el sobrepico máximo, el tiempo de levantamiento y el tiempo de asentamiento.

Para el desarrollo de este método debemos elegir el controlador a usar, una vez *elegido el controlador, la siguiente tarea es determinar sus parámetros. A continuación se* debe realizar la verificación del desempeño del sistema y en caso de ser necesario reajustar los parámetros del compensador.

Se emplea la simulación mediante un software específico, en este caso Matlab. Una vez obtenido un modelo matemático satisfactorio, se debe construir un prototipo y probar el sistema en lazo abierto. Si se asegura la estabilidad absoluta en lazo abierto, se cierra el lazo y prueba el desempeño del sistema en lazo cerrado. Mediante el enfoque de prueba y error, se debe cambiar el prototipo hasta que el sistema cumpla las especificaciones. Se analiza cada prueba e incorpora los resultados de este análisis en la prueba siguiente. Se tiene que verificar que el sistema final cumpla las especificaciones de desempeño y, al mismo tiempo, sea confiable.

Cuando se diseña un sistema de control, si se requiere de un ajuste diferente al

de la ganancia, debemos modificar los lugares geométricos de las raíces originales insertando un compensador conveniente. Una vez comprendidos los efectos de la adición de los polos y/o ceros sobre el lugar geométrico de las raíces, podemos determinar con facilidad las ubicaciones de los polos y los ceros del compensador que volverán a dar una forma conveniente al lugar geométrico de las raíces. En esencia, en el diseño realizado mediante el método del lugar geométrico de las raíces, los lugares geométricos de las raíces del sistema se vuelven a construir mediante el uso de un compensador, a fin de poder colocar un par de polos dominantes en lazo cerrado en la posición deseada.

- **Efectos de la adición de polos:** La adición de un polo a la función de transferencia en lazo abierto tiene el efecto de jalar el lugar geométrico de las raíces a la derecha, lo cual tiende a disminuir la estabilidad relativa del sistema y alentar el asentamiento de la respuesta. La Figura 2.18 muestra ejemplos de los lugares geométricos de las raíces, que presentan el efecto de la adición de uno o dos polos a un sistema de un solo polo.

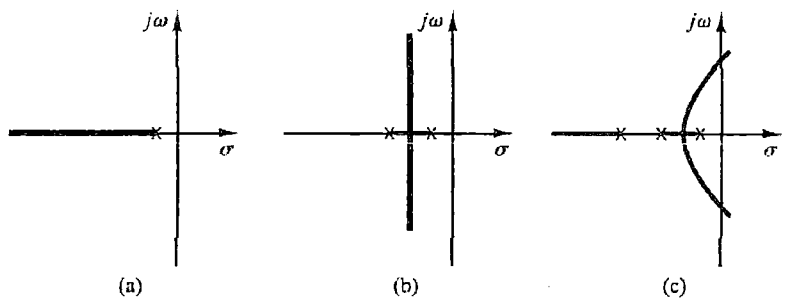


Figura 2.18: Efectos de la adición de polos al Sistema.

- **Efectos de la adición de ceros:** La adición de un cero a la función de transferencia en lazo abierto tiene el efecto de jalar el lugar geométrico de las raíces hacia la izquierda, con lo cual el sistema tiende a ser más estable, y se acelera el asentamiento de la respuesta. (Físicamente, la adición de un cero a la función de transferencia de la trayectoria directa significa agregar al sistema un control derivativo. El efecto de tal control es introducir un grado de previsión al sistema y acelerar la respuesta transitoria.) La Figura 2.19 (a) muestra los lugares geométricos de las raíces para un sistema estable con una ganancia pequeña, pero inestable con una ganancia grande. Las Figuras 2.19 (b), (c) y (d) muestran las gráficas del lugar geométrico de las raíces para

el sistema cuando se añade un cero a la función de transferencia en lazo abierto. Observar que, cuando se agrega un cero al sistema, éste se vuelve estable para todos los valores de ganancia.

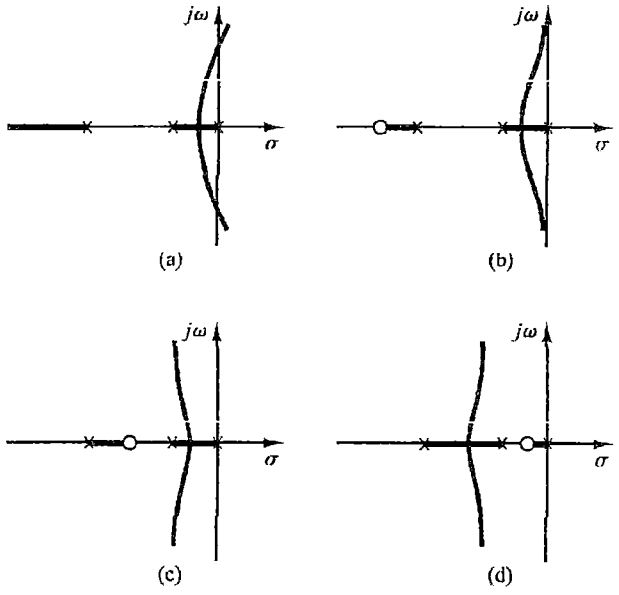


Figura 2.19: Efectos de la adición de ceros al Sistema.

2.4. Protocolos de Comunicación

Los protocolos de comunicación que se usan en los procesos industriales para un sistemas SCADA son: Ethernet, Serial, OPC.

2.4.1. Protocolo Ethernet

Es un estándar de redes de área local para computadores, Ethernet define las características de cableado y señalización de nivel físico y los formatos de trama de datos del nivel de enlace de datos del modelo OSI.

Ethernet se tomó como base para la redacción del estándar internacional IEEE 802.3, siendo usualmente tomados como sinónimos. Se diferencian en uno de los campos de la trama de datos. Sin embargo, las tramas Ethernet e IEEE 802.3 pueden coexistir en la misma red.

Las tecnologías Ethernet que existen se diferencian en estos conceptos:

- *Velocidad de transmisión*, Velocidad a la que transmite la tecnología.
- *Tipo de cable*, Tecnología del nivel físico que usa la tecnología.
- *Longitud máxima*, Distancia máxima que puede haber entre dos nodos adyacentes (sin estaciones repetidoras).
- *Topología*, Determina la forma física de la red. Bus; si se usan conectores T (hoy sólo usados con las tecnologías más antiguas) y estrella si se usan hubs (estrella de difusión) o switches (estrella conmutada).

Los elementos de una red Ethernet son: tarjetas de red, repetidoras, concentradoras, puentes, los conmutadores, los nodos de red y el medio de interconexión.

2.4.2. Protocolo Serial

La comunicación serial es un protocolo muy común para la comunicación entre dispositivos que se incluye de manera estándar en prácticamente cualquier computadora. La mayoría de las computadoras incluyen dos puertos seriales $RS-232$. La comunicación serial es también un protocolo común utilizado por varios dispositivos para instrumentación.

El concepto de comunicación serial es sencillo. El puerto serial envía y recibe bytes de información un bit a la vez. Aun y cuando esto es más lento que la comunicación en paralelo, que permite la transmisión de un byte completo por vez, este método de comunicación es más sencillo y puede alcanzar mayores distancias de hasta $1200\ m$.

Típicamente, la comunicación serial se utiliza para transmitir datos en formato ASCII. Para realizar la comunicación se utilizan 3 líneas de transmisión: (1) Tierra (o referencia), (2) Transmitir, (3) Recibir. Debido a que la transmisión es asincrónica, es posible enviar datos por una línea mientras se reciben datos por otra.

Las características más importantes de la comunicación serial son la velocidad de transmisión, los bits de datos, los bits de parada, y la paridad. Para que dos puertos se puedan comunicar, es necesario que las características sean iguales.

2.4.3. Protocolo OPC

Actualmente los sistemas SCADA disponen de un tipo de comunicación que se ha convertido en un estándar a nivel internacional para transferir datos independientemente de la aplicación y del lenguaje de comunicación. Dicho estándar es el denominado OPC.

El OPC (OLE for Process Control: Object Linking and Embedding for Process Control) es un estandar de comunicacion en el campo del control y supervisión de procesos industriales, basado en una tecnologia Microsoft, que ofrece una interfaz comun para comunicación, que permite que componentes de software individuales interaccionen y compartan datos.

La comunicación OPC se realiza a través de una arquitectura Cliente - servidor. El servidor OPC es la fuente de datos (como un dispositivo hardware a nivel de planta) y cualquier aplicación basada en OPC puede acceder a dicho servidor para leer/escribir cualquier variable que ofrezca el servidor.

Es una solución abierta y flexible al clásico problema de los drivers propietarios. Prácticamente todos los mayores fabricantes de sistemas de control, instrumentación y de procesos han incluido OPC en sus productos. Tiene como propósito cubrir las necesidades de acceso en forma estándar de las distintas aplicaciones hacia los dispositivos o base de datos. Es decir una aplicación X y una Y se podrían comunicar con distintos servidores A, B, C de diferentes protocolos de comunicación, siempre y cuando estos tengan interfaces OPC las cuales se les puede aprovechar para conectarlas con las aplicaciones, como muestra la Figura 2.20.

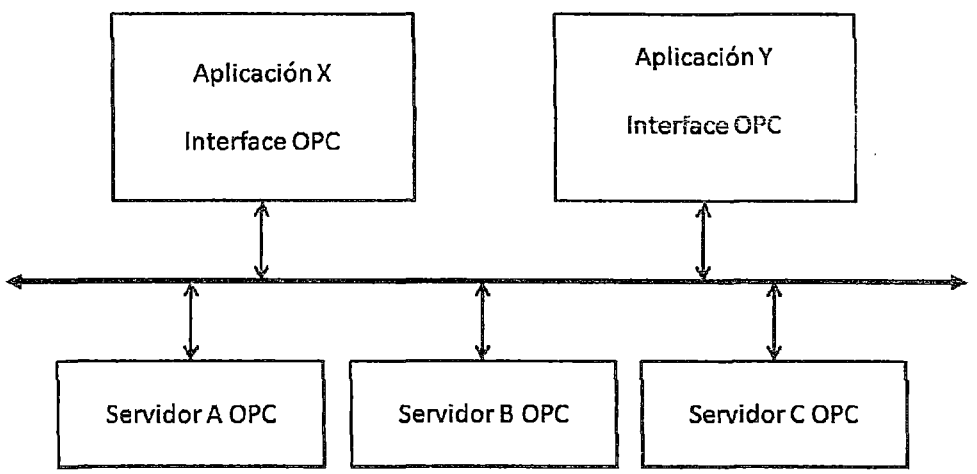


Figura 2.20: Arquitectura OPC, Explicación gráfica.

En la actualidad la mayoría de dispositivos controladores contienen drivers OPC, por tanto no es necesario adaptar los drivers ante nuevos dispositivos de otras marcas. La arquitectura OPC es de entorno heterogéneo, es decir integra equipos de distintos fabricantes y simplifica las comunicaciones. Un cliente OPC se puede conectar a múltiples servidores OPC, tan solo direccionándolos. El ejemplo que se presenta en la figura 2.21 establece el sistema SCADA como cliente OPC y a los dispositivos como los Servidores OPC, ambos se pueden conectar aun teniendo diferentes protocolos de comunicación gracias a la comunicación OPC.

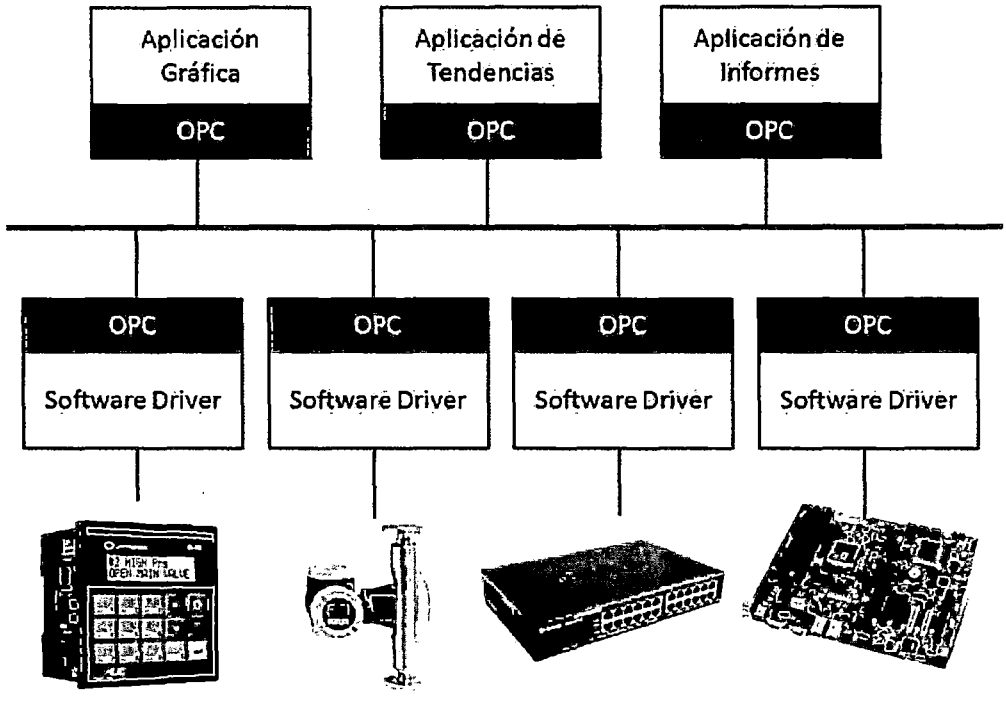


Figura 2.21: Arquitectura OPC, conexión de diferentes protocolos por medio del OPC.

Arquitectura OPC, Cliente - Servidor

La arquitectura cliente - servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras, como muestra la Figura 2.22.

Elementos de la comunicación OPC

1. El servidor
    - Mantiene información sobre el servidor.
    - Sirve como contenedor para objetos del grupo OPC.

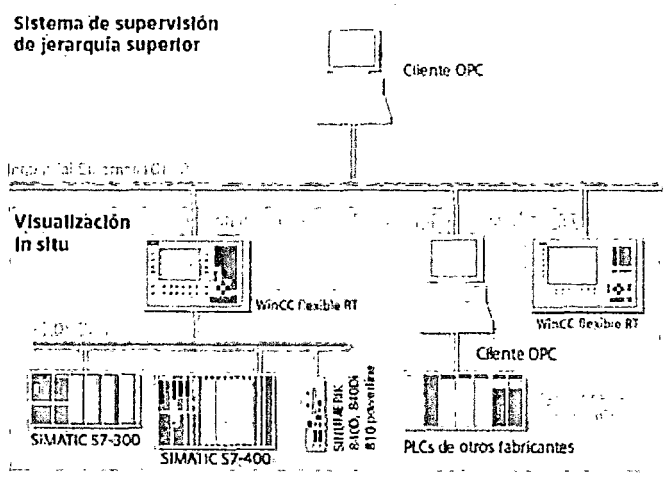


Figura 2.22: Arquitectura OPC, Cliente - Servidor.

2. El grupo
    - Mantiene información sobre sí mismo.
    - Provee mecanismos para contener/organizar lógicamente ítems.
3. El elemento
    - Representan conexiones a fuentes de datos dentro de un servidor

Estos elementos se detallan en la Figura 2.23

2.5. Sistema SCADA

La palabra SCADA es el acrónimo de Supervisory Control And Data Adquisition, que traducido al español significa Supervisión, Control y Adquisición de Datos; es una aplicación de software diseñado con la finalidad de controlar y supervisar datos a distancia, los cuales se basan en la adquisición de datos de los procesos remotos.

Los sistemas SCADA utilizan la computadora y tecnologías de comunicación para automatizar el monitoreo y el control de procesos industriales, proporcionando la información oportuna para poder tomar decisiones operacionales apropiadas.

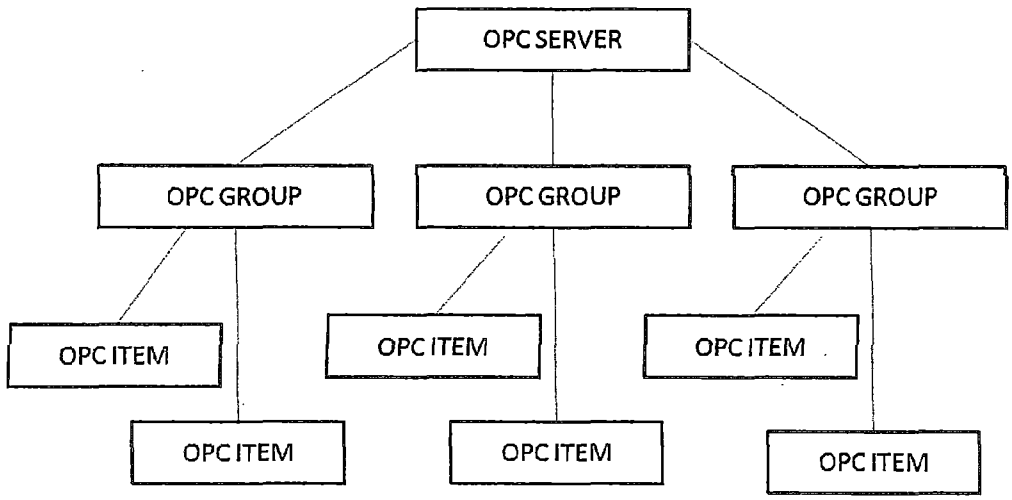


Figura 2.23: Elementos de Comunicación OPC.

Los sistemas SCADA involucran muchos subsistemas, por ejemplo, la adquisición de los datos puede estar a cargo de un PLC (Controlador Lógico Programable) o de dispositivos los cuales toman las señales y las envía a las estaciones remotas usando un protocolo determinado.

Las tareas de Supervisión y Control generalmente están más relacionadas con el software SCADA, en él, el operador puede visualizar en la pantalla del computador de cada una de las estaciones remotas que conforman el sistema, los estados de ésta, las situaciones de alarma y tomar acciones físicas sobre algún equipo lejano, la comunicación se realiza mediante buses especiales o redes LAN. Todo esto se ejecuta normalmente en tiempo real, y están diseñados para dar al operador de planta la posibilidad de supervisar y controlar dichos procesos.

El sistema SCADA actúa sobre los dispositivos instalados en la planta, como son los controladores autómatas, sensores, actuadores, registradores, etc. Además permite controlar el proceso desde una estación remota, para ello el software brinda una interfaz gráfica que muestra el comportamiento del proceso en tiempo real. Generalmente se vincula el software al uso de una computadora o de un PLC.

2.5.1. Funciones de los sistemas SCADA

Dentro de las funciones básicas de los sistemas SCADA, destacan:

- **Supervisión remota de instalaciones y equipos:** Permite al operador conocer el estado de desempeño de las instalaciones y los equipos alojados en la planta, lo que permite dirigir las tareas de mantenimiento y estadística de fallas.
- **Control remoto de instalaciones y equipos:** Mediante el sistema se puede activar o desactivar los equipos remotamente (por ejemplo abrir válvulas, activar interruptores, prender motores, etc.), de manera automática y también manual. Además es posible ajustar parámetros, valores de referencia, algoritmos de control, etc.
- **Procesamiento de datos:** El conjunto de datos adquiridos conforman la información que alimenta el sistema, esta información es procesada, analizada, y comparada con datos anteriores, y con datos de otros puntos de referencia, dando como resultado una información confiable y veraz.
- **Visualización gráfica - dinámica:** El sistema es capaz de brindar imágenes en movimiento que representen el comportamiento del proceso, dándole al operador la impresión de estar presente dentro de una planta real. Estos gráficos también pueden corresponder a curvas de las señales analizadas en el tiempo.
- **Generación de reportes:** El sistema permite generar informes con datos estadísticos del proceso en un tiempo determinado por el operador.
- **Representación de señales de alarma:** A través de las señales de alarma se logra alertar al operador frente a una falla o la presencia de una condición perjudicial o fuera de lo aceptable. Estas señales pueden ser tanto visuales como sonoras.
- **Almacenamiento de información histórica:** Se cuenta con la opción de almacenar los datos adquiridos, esta información puede analizarse posteriormente, el tiempo de almacenamiento dependerá del operador o del autor del programa.
- **Programación de eventos:** Referido a la posibilidad de programar subprogramas que brinden automáticamente reportes, estadísticas, gráfica de curvas, activación de tareas automáticas, etc.

2.5.2. Tipos de Sistemas SCADA

Los sistemas SCADA pueden dividirse en dos categorías y según estas definirlas:

Sistemas SCADA Abierto y Propietario

1. Los sistemas abiertos son aquellos desarrollados para poder ser aplicados a cualquier *tipo de tecnología o* dispositivo de control, es decir si se necesita enlazar un equipo de distintos fabricantes, es necesario solo contar con los drivers que interpreten los distintos códigos de comunicación utilizados. La principal ventaja de este tipo de sistema es su capacidad de crecimiento conjunto con la planta, es decir nuevos equipos pueden ser implementados así sean de distintos fabricantes.
2. Los sistemas propietarios son aquellos desarrollados por los propios fabricantes de equipos o dispositivos de control, los cuales se comunican entre sí con sus propios drivers; la principal desventaja de este tipo de software SCADA es la gran dependencia que se tiene del proveedor del sistema.

Sistemas SCADA Comerciales y Libres

1. Un sistema SCADA comercial es aquel en el que por lo general su desarrollo está a *cargo de una compañía*, la cual se encarga de crear todas las *interfaces necesarias* para comunicar los distintos dispositivos, y una vez finalizado esto, entregar al usuario un producto de fácil uso. Mientras más confiable y amigable sea el software, este es más costoso.
2. Un sistema SCADA libre por lo general fue creado como un SCADA comercial, con el transcurso del tiempo se vio que había mayores ventajas en poner estos sistemas con su código de programación en forma abierta a disposición de distintos desarrolladores alrededor del mundo, los cuales cooperan con su desarrollo, por lo general la única condición para poder adquirir estos software es comprometerse a que una vez logrado el objetivo buscado, este conocimiento sea compartido. En las Tablas 2.5 y 2.6 mostramos algunos sistemas comerciales y libres.

SOFTWARES SCADA COMERCIALES		
NOMBRE	COMPAÑÍA	PRECIO
Invesysy Scada System	Foxboro	$ 1900.00
Simantic WinCC	Siemens	$ 2600.00
DAQ Factory	Azeo Tech	$ 2500.00
LookOut	National Instrument	$ 1045.00
Intouch HMI	Wondeware	$ 3100.00
JAKO SCADA	JANUS	
CitecSCADA	Schneider Electric	
Monitor Pro	Schneider Electric	
Aimax	Desin Instruments S.A	
FIX	Intellution	
SCADA INTOUCH	LOGITEK	
SYSMAC SCS	OMRON	
Scatt Graph 5000	ABB	
VB _ ScadaLader	Microsoft	
AddVANTAGE Pro	ADCON Telemetry	
LabView	National Instruments	

Tabla 2.5: Lista de Softwares SCADA Comerciales.

SOFTWARES SCADA LIBRE, OPEN SOURCES				
NOMBRE	DESARROLLADOR	PLATAFORMA	BASE DE DATOS	INTERFACES
EPICS	National Laboratory	Linux	Propio	Modbus, Ethernet
LINKDOY	Andalucía España	Linux	MySQL	Modbus, Ethernet, Web
PVBROWSER	Web	Linux	MySQL	Modbus, Ethernet, Web
SOFT CONTROL	UDEP	Windows	MySQL	Ethernet, Serial, OPC

Tabla 2.6: Lista de Softwares SCADA libres.

2.5.3. Partes de un Sistema SCADA

Unidades Terminales Maestras (MTU)

Es el centro del sistema, es el componente del cual el personal de operaciones se valdrá para visualizar las distintas variables de los procesos en la planta; generalmente una MTU es un computador PC, de regular capacidad, el cual cumple funciones no solo de monitoreo, sino a la vez de almacenamiento y procesado ordenado de datos, los cuales servirán para las distintas aplicaciones que el operario o usuario requiera.

Unidades Terminales Remotas (RTU)

El RTU es una unidad de control y adquisición de datos independiente, por lo general ha estado basada en microprocesadores, los cuales monitorean y controlan equipo en algunas ubicaciones remotas desde la estación central. Esta tarea primaria es controlar y adquirir datos desde los equipos de los procesos en la ubicación remota y transferir esta información hacia la estación central.

Infraestructura y Métodos de Comunicación

Los sistemas SCADA tienen tradicionalmente muchos requerimiento de comunicación, como por ejemplo lo diferentes protocolos, Ethernet, serial, y OPC, wireless, etc. Las partes de un sistemas SCADA en un dibujo, quedaría expresado como muestra la Figura 2.24.

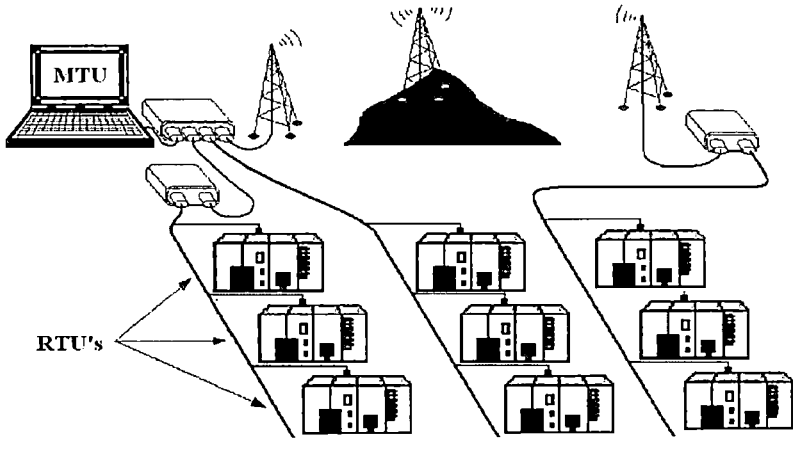


Figura 2.24: Partes de un Sistema SCADA.

2.6. Controlador Lógico Programable

El PLC es un sistema electrónico programable diseñado para ser usado en un entorno industrial, que utiliza una memoria programable para el almacenamiento interno de instrucciones orientadas al usuario, para implantar soluciones específicas tales como, funciones lógicas, secuencia, temporización, recuentos y funciones aritméticas con el fin de controlar mediante entradas y salidas digitales y análogas diversos tipos de máquinas o procesos.

2.6.1. Estructura de un PLC

La estructura básica de un autómata programable se muestra en la Figura 2.25.

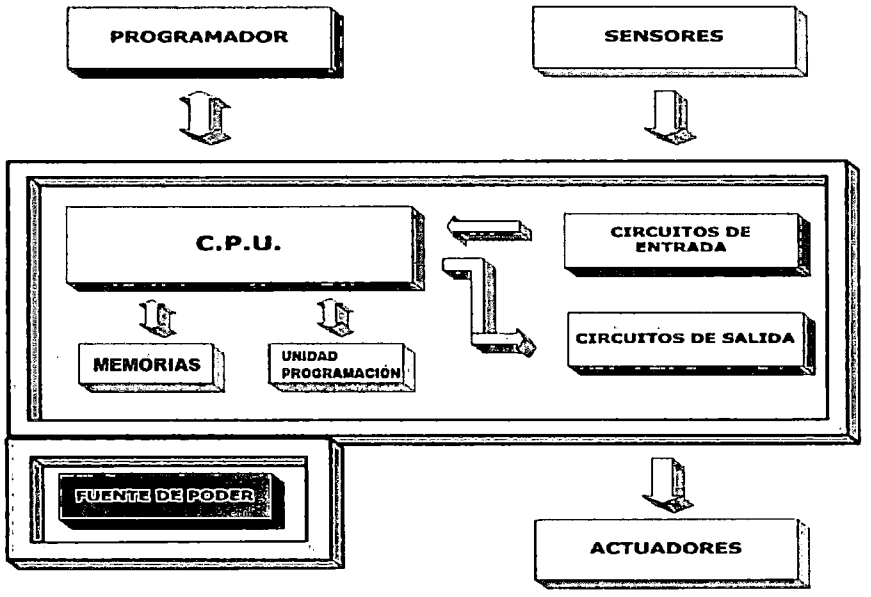


Figura 2.25: Estructura de un PLC.

- **Unidad Central de Procesos (CPU):** El CPU es un elemento inteligente que está en capacidad de leer e interpretar las instrucciones cargadas en la memoria y sobre la base de los estados de las entradas, toma de decisiones sobre las salidas.
  Generalmente, todas las unidades de procesamiento de los PLC están basadas en

microprocesadores de 8, 16 ó 32 bits, los cuales tienen capacidad de manejar los comandos e instrucciones de entradas, los estados de las señales, también proveen la capacidad de procesamiento lógico, la cual se encarga de resolver lógica booleana, temporización, secuencia, suma, resta, multiplicación, división y conteo.

- **Memorias:** La memoria es el almacén donde el Controlador Lógico Programable guarda todo cuanto necesita para ejecutar la tarea de control, como por ejemplo los datos del proceso, señales de planta, entradas y salidas, variables internas, de bit y de palabra, datos alfanuméricos y constantes, datos de control, instrucciones de usuario (programa), etc. Existen varios tipos de memorias:
  - RAM, memoria de lectura y escritura.
  - ROM, memoria de solo lectura, no reprogramable.
  - EPROM, memoria de solo lectura, reprogramables con borrado por ultravioletas.
  - EEPROM, memoria de solo lectura, alterables por medios eléctricos.
- **Fuente de Poder:** Usualmente los suministros de voltaje de los PLC, requieren fuentes de poder AC; sin embargo, algunos PLC aceptan entradas de fuentes DC, estos son muy solicitados para aplicaciones en las operaciones de las plataformas de operación que están mar adentro donde comúnmente se usan las fuentes DC.
  Los requerimientos más comunes son las fuentes de 120 VAC o 220 VAC, mientras algunos pocos controladores aceptan 24 VDC.
- **Sistemas de Entradas y Salidas:** La característica principal que hace extremadamente atractivo a un PLC y que lo diferencia de un computador es su sistema de entradas y salidas (E/S) compuesto en la mayoría de los casos por módulos diseñados especialmente para proveer la conexión física entre el mundo exterior (Equipos de Campo) y la unidad de procesamiento. Esta es la conexión real entre el CPU y del PLC y los dispositivos de campo.
  A través de varios circuitos de interfaz y el uso de los dispositivos de campo (Sensores de Limites, Transductores, etc), el controlador puede censar y medir cantidades físicas requeridas a máquinas de procesos tales como: proximidad, posición, movimiento, nivel, temperatura, presión, voltaje.
  Basados en el estado de los dispositivos de campo censados o los valores medidos en el proceso, el CPU emite comandos que controlan variados dispositivos así como son válvulas, motores, bombas y alarmas. Resumiendo la interfaz de Entradas / Salidas es el sentido y hábil motor requerido por el CPU para ejercer el control sobre máquinas y procesos.

- **Unidad de Programación:** Es el conjunto de medios, hardware y software mediante los cuales el programador introduce y depura las secuencias de instrucciones (en uno u otro lenguaje) que constituyen el programa a ejecutar.

2.6.2. Clasificación de los PLC

Por su contrucción

- **Compacto**

  Estos PLC tienen incorporada la fuente de alimentación, su CPU y los módulos de entrada y salida en un solo módulo principal y permiten manejar desde unas pocas entradas y salidas hasta varios cientos (alrededor de 500 entradas y salidas), su tamaño es superior a los PLC tipo Nano y soportan una gran variedad de módulos especiales, tales como entradas y salidas analógicas, módulos contadores rápidos, módulos de comunicaciones, interfaces de operador, expansiones de E/S, como muestra la Figura 2.26.

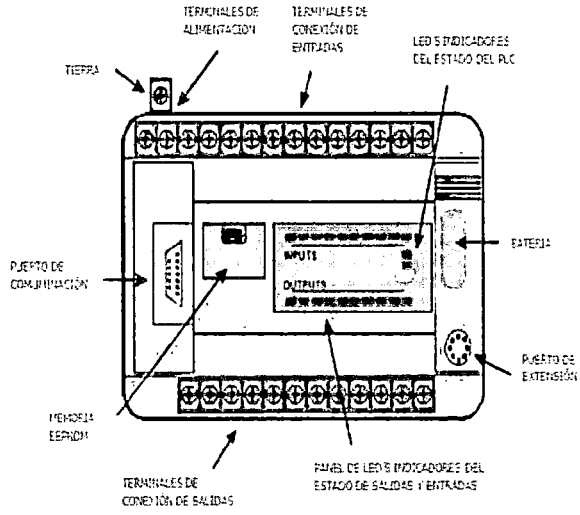


Figura 2.26: Representación de un PLC Compacto.

- **Modular**

  Estos PLC se componen de un conjunto de elementos, representados en la Figura 2.27 que conforman el controlador final. Estos son:

1. Rack
2. Barra de Compensación de potencial
3. Módulos de Entradas y Salidas
4. Módulos de Comunicaciones
5. CPU
6. Tarjeta de Memoria
7. Fuente de Modulación

De estos tipos de PLC existen desde los denominados Micro PLC que soportan gran cantidad de entradas y salida, hasta los PLC de grandes prestaciones que permiten manejar miles de entradas y salidas.

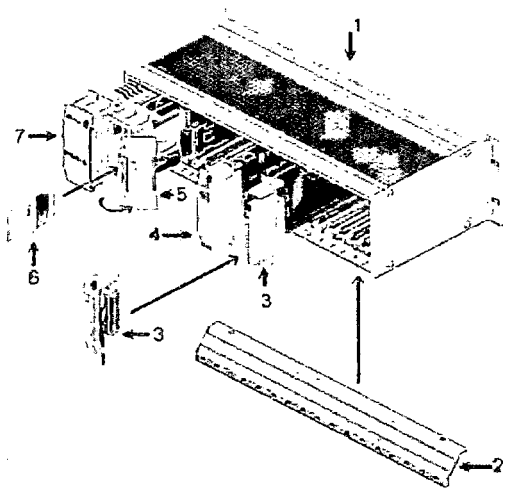


Figura 2.27: Representación de un PLC Modular.

Por su capacidad

- **Nivel 1:** Control de variables discretas y pocas analógicas, operaciones aritméticas y capacidad de comunicación elementales.
- **Nivel 2:** Control de variables discretas y analógicas. Matemáticas de puntos flotantes. E/S inteligentes. Conexión de red. Gran capacidad de manejo de datos analógicos y discretos.

Por su cantidad de Entradas y Salidas

- Nano PLC: menor a 64 E/S.
- Micro PLC: 64 E/S.
- PLC Pequeño: 65 a 255 E/S.
- PLC Mediano: 256 a 1023 E/S.
- PLC Grande: mayor a 1024 E/S.

2.6.3. *Principales Marcas de PLC*

- Siemens
- ABB
- Allen Bradley
- Schneider Electric
- Festo
- General Electric
- Mitsubishi
- Hitachi
- Omron

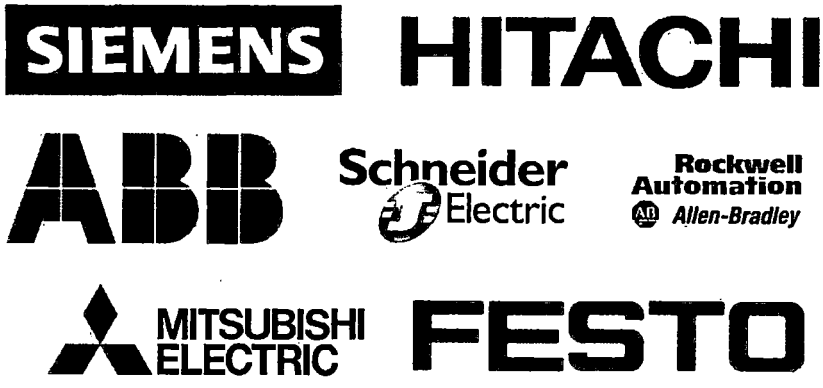


Figura 2.28: Principales Marcas de PLC.

CAPÍTULO 3
GENERALIDADES

3.1. Planta de Presión

La planta de presión, del que haremos el tema de investigación para la realización de este proyecto de tesis, está ubicado en las instalaciones del laboratorio de ingeniería electrónica de la Universidad Nacional Pedro Ruiz Gallo, la planta está diseñada como muestra la Figura 3.1.

El diseño está compuesto por una parte Externa y una parte Interna:

3.1.1. Parte Externa

- **Válvula Proporcional**
  Marca $Danfoss$, de la gama $EV260B$. Se trata de una válvula solenoide servoaccionada de 2 vías con función de modulación proporcional. Mediante la regulación continua de la corriente de la bobina, es posible situar la armadura en cualquier posición del tubo de la armadura y, de este modo, ajustar la bobina entre las posiciones de cierre completo y apertura completa. La válvula se abre totalmente cuando la corriente de la bobina alcanza su máximo valor. Tiene como bobina de excitación a la bobina tipo $BG024D$, con convertidor de señal piloto de 4-20 mA. Se puede observar la válvula en la Figura 3.2.

- **Indicador de Presión**
  Cuenta con 2 manómetros. Indica la presión en un rango de 0 a 100 PSI y de 0 a 700 KPa. Este dispositivo se muestra en la Figura 3.3.

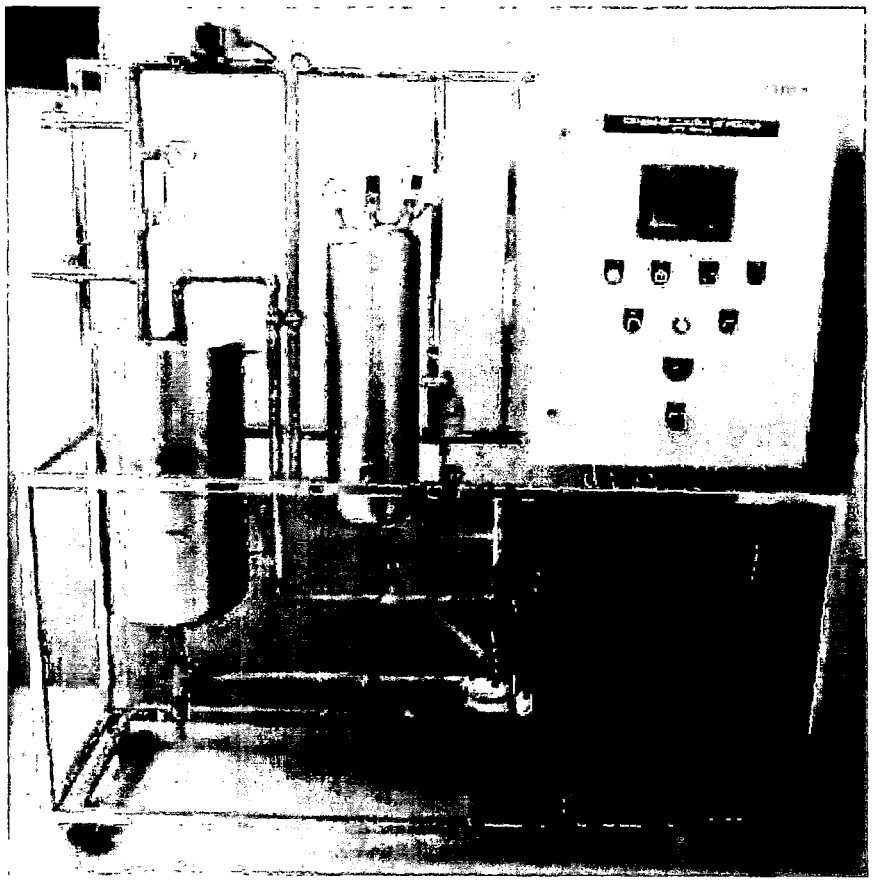


Figura 3.1: Planta de Presión EPIE.

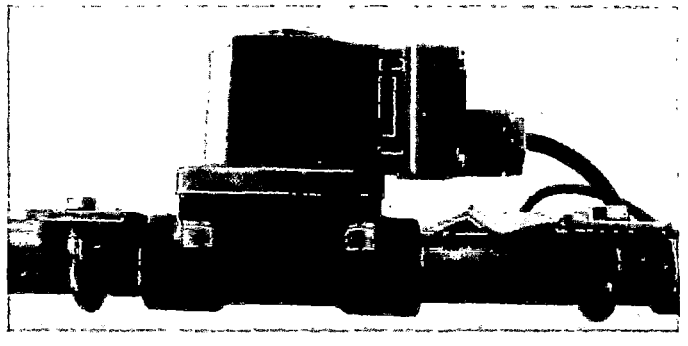


Figura 3.2: Imagen de la Válvula Proporcional.

- **Transmisor de Presión**

  Marca $Danfoss$, tipo MBS 3000, se trata de un transmisor de presión compacto, con una señal de salida de 4 a $20\ mA$, y un rango de medición de 0 a 6 bar. Este dispositivo se muestra en la Figura 3.4.

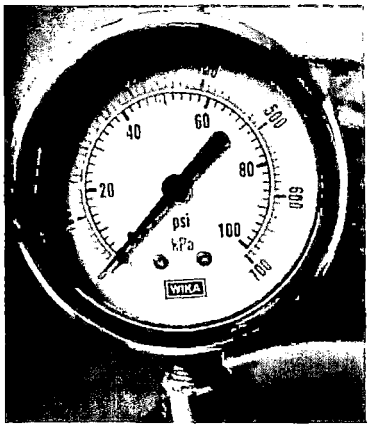


Figura 3.3: Imagen del indicador de Presión.

Figura 3.4: Imagen del Transmisor de Presión.

- **Presóstato**

  De la marca $Danfoss$, tipo KPI 135. Se trata de un limitador electromecánico que verifica que la presión de los fluidos se mantenga dentro de un intervalo de presión específico. trabaja en un rango de $-0{,}2$ a 8 bar. Este dispositivo se muestra en la Figura 3.5.

- **Electrobomba**

  Cuenta con una electrobomba autocebante de Jaula de ardilla marca $Pentax$, puede funcionar con 220 VAC o 380 VAC. Este dispositivo se muestra en la Figura 3.6.

- **Llaves manuales y tanque**

  Las 7 llaves de paso están ubicada en lugares estratégicos de la planta, de tal manera que faciliten el mantenimiento de ésta, así como para simular perturbaciones en la

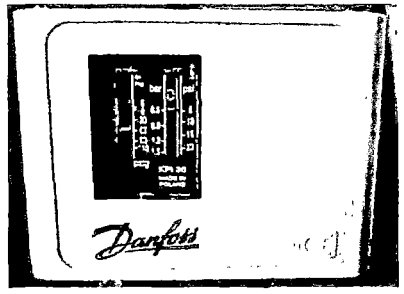


Figura 3.5: Imagen del Presóstato.

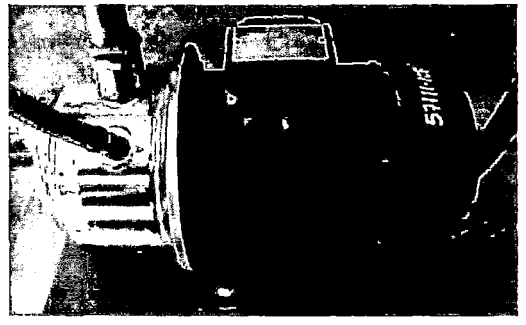


Figura 3.6: Imagen de la electrobomba.

planta. Estas llaves se presentan en la Figura 3.7, y son todas del mismo tipo. Cuenta con un tanque de almacenamiento de agua, y un tanque de presión. como se muestra en la Figura 3.8.

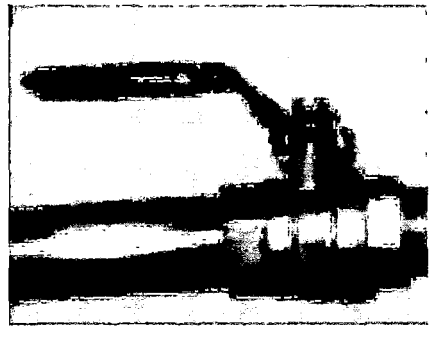


Figura 3.7: Imagen de una de las 7 llaves manuales.

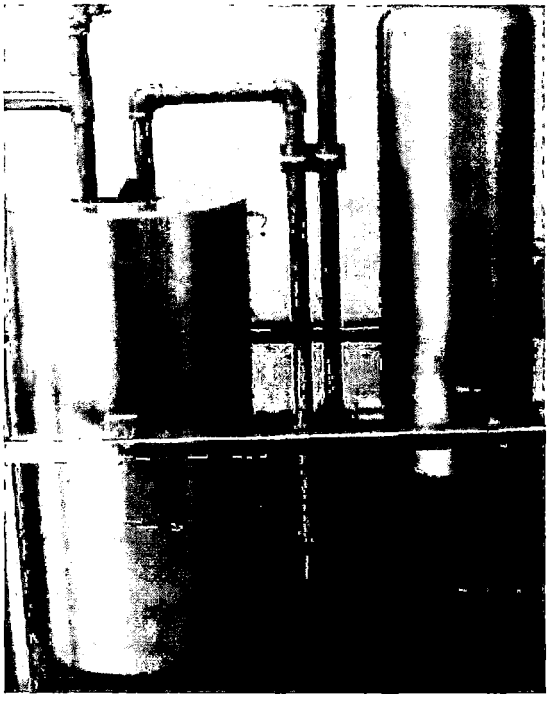


Figura 3.8: Imagen de los tanques, tanque de presión (derecha), tanque de almacenamiento (izquierda).

3.1.2. Parte Interna (Tablero)

El tablero de control que se muestra en la Figura 3.9 en su exterior tiene 4 Lámparas indicadoras de colores amarillo, verde y 02 rojos, 02 botones, uno verde con la etiqueta $START$ y uno rojo con la etiqueta $STOP$, un potenciómetro, y una botón de emergencia.

En el interior encontramos:

- **Elementos de Seguridad**
  El sistema de protección está conformado por una llave térmica monofásica, una llave términca trifásica, un interruptor diferencial y un guardamotor. Estos dispositivos se muestran en las Figuras 3.10 y 3.11

- **Variador de Frecuencia**
  En la Figura 3.12 se muestra el variador Altivar 31 de Telemecanique, variador de velocidad para motores asíncronos.

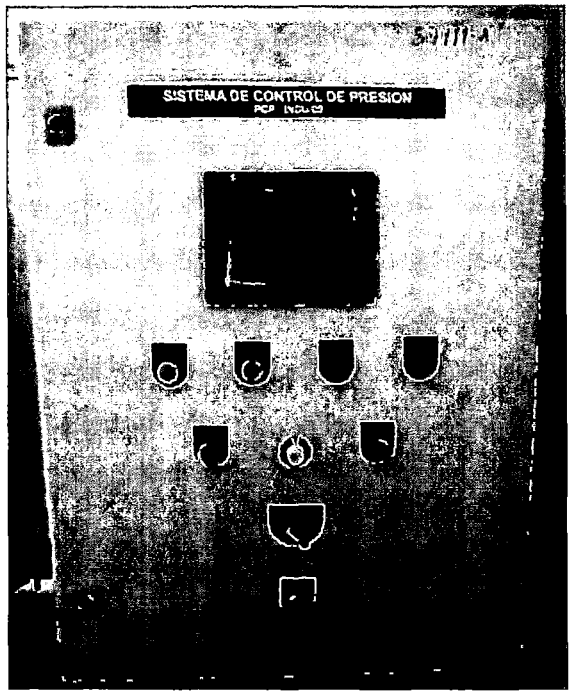


Figura 3.9: Imagen del Tablero de Control.

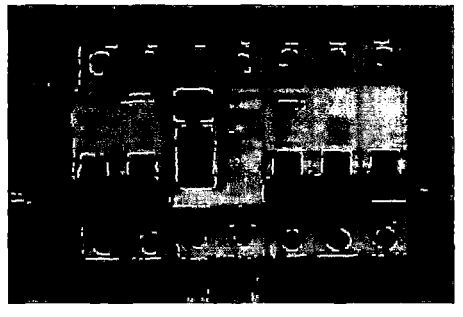


Figura 3.10: Imagen de los dispositivos de seguridad: LLave térmica monofásica (izquierda), interruptor diferencial (centro), llave térmica trifásica (derecha).

- **Controlador Lógico programable**

  En la Figura 3.13 se muestra el PLC Modicon M340 de Telemecanique compuesto de Módulo de alimentación, el módulo CPU, módulo de entradas y salidas analógicas, y módulo de entradas y salidad digitales.

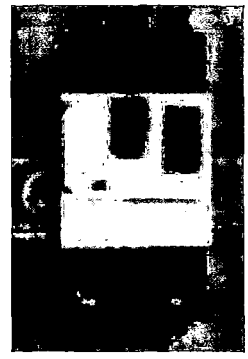


Figura 3.11: Imagen del dispositivo de seguridad: Guardamotor.

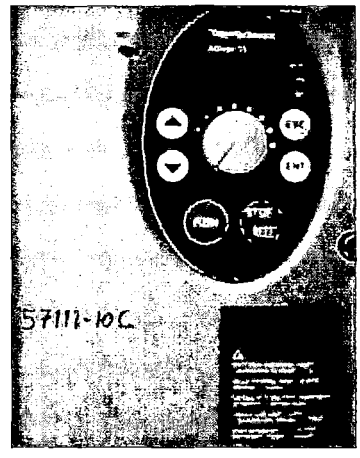


Figura 3.12: Imagen del Variador Altivar 31.

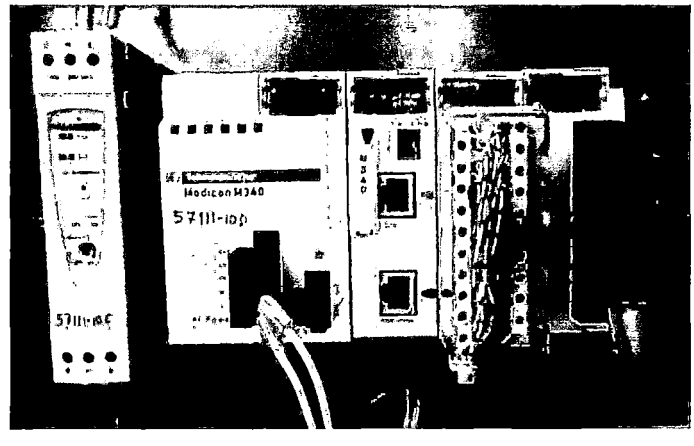


Figura 3.13: Imagen del Modicon M340 de Telemecanique.

3.2. PLC Modicon M340

La planta de presión es controlada por el PLC MODICON M340 de la marca Schneider Electric. Incorpora un Módulo de alimentación, un CPU BMX P34 2020 que contiene el puerto Ethernet y Modbus, el módulo BMX AMM 0600 y el módulo BMX DDM 3202K, como se muestra en la Figura 3.13.

Modulo Alimentación CPS 2000

Los módulos de alimentación BMX CPS 2000 de corriente alterna cuentan con una fuente de alimentación integrada que proporciona una tensión de 24 V_{cc}, que se emplea para la alimentación de los sensores. La Figura 3.14 identifica los distintos componentes de un módulo de alimentación CPS 2000 que se describen a continuación:

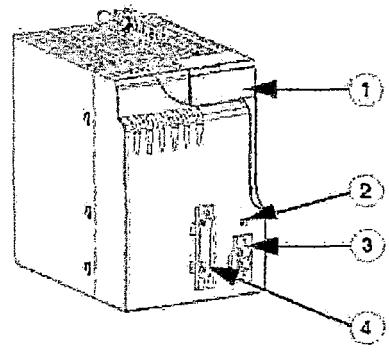


Figura 3.14: Imagen del Módulo de alimentación CPS 2000.

1. El panel de visualización, un indicador LED OK (verde) encendido si el módulo de la fuente de alimentación del bastidor está operativa y funciona correctamente y un indicador LED de 24 V_{cc} (verde) encendido si la fuente de alimentación del sensor está en funcionamiento.
2. Botón RESET.
3. Conector del relé de alarma.

4. Conector para la red de entrada (y fuente de alimentación del sensor de 24 V_{cc} para los módulos de alimentación BMX CPS 2000 de corriente alterna).

Módulo CPU BMX P34 2020

La figura 3.15 muestra al CPU con cada una de sus partes y detallaremos las funciones que desempeñan a continuación:

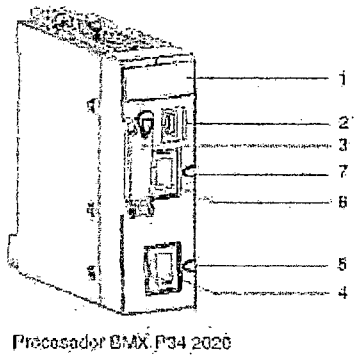


Figura 3.15: Imagen del Módulo CPU BMX P34 2020.

1. Panel de visualización.
2. Puerto USB.
3. Puerto de protección de la tarjeta de memoria.
4. Puerto serie.
5. Anillo de identificación del puerto serie (negro).
6. Puerto Ethernet.
7. Anillo de identificación del puerto Ethernet (verde).

Módulo analógico BMX AMM 0600

El modulo BMX AMM 0600 combina 4 entradas analógicas y 2 salidas analógicas. El modulo ofrece la siguiente gama, de acuerdo a la selección realizada en su configuración:

a) Rango de voltaje en la entrada: -10 a +10 V, 0 a 10 V, 0 a 5 V.

b) Rango de corriente en la entrada: 0 a 20 mA, 4 a 20 mA.

c) Rango de voltaje en la salida:-10 a +10 V.

d) Rango de corriente en la salida: 0 a 20 mA, 4 a 20 mA.

Módulo digital BMX DDM 3202K

El módulo BMX DDM 3202 K es un módulo binario de 24 VCC conectado a través de un conector de 40 pines. Se trata de un módulo de lógica positiva: sus 16 canales de entrada reciben corriente desde el sensor y sus 16 canales de salida proporcionan corriente a los pre actuadores.
Caracterśticas:

a) Voltaje 24 V.

b) Corriente 2.5 mA.

c) Impedancia 9.6 Kohms.

d) Todas las salidas están equipadas con circuitos de desmagnetización rápida.

3.3. Software Unity Pro XL

Para la programación del PLC se usó el software Unity PRO XL v4. Es una plataforma de software Licenciado, diseñado para programar el PLC Modicón M340, y cualquier

PLC de la marca Schneider Electric.

Para el control de la planta se utiliza un PLC M340 de Modicom. El autómata está formado por módulos o tarjetas con diversas funciones: entradas, salidas, comunicación, CPU. Estos módulos se distribuyen a lo largo del bastidor, que cuenta con su fuente de alimentación.

3.3.1. Configuración en Unity Pro para el PLC Modicon

1. Se debe crear un nuevo proyecto para lo cual se debe hacer clic en el menú Fichero - Nuevo. Aparecerá una ventana como muestra la Figura 3.16, en la que se pide escoger el PLC que se programará. En este caso, el PLC es el Modicom M340, con CPU BMX P34 2020.

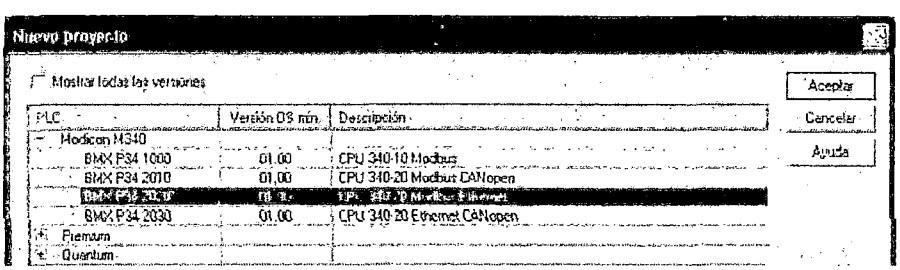


Figura 3.16: Selección del PLC a utilizar.

2. Una vez seleccionado aparecerá en la parte izquierda de la pantalla el explorador de proyectos, como se muestra en la Figura 3.17.

   Como se puede observar, el proyecto de Unity se estructura en:

   - Configuración
   - Variables y tipos de datos
   - Movimiento
   - Comunicación
   - Programas
   - Tablas de animación y pantallas de operación
   - Documentación

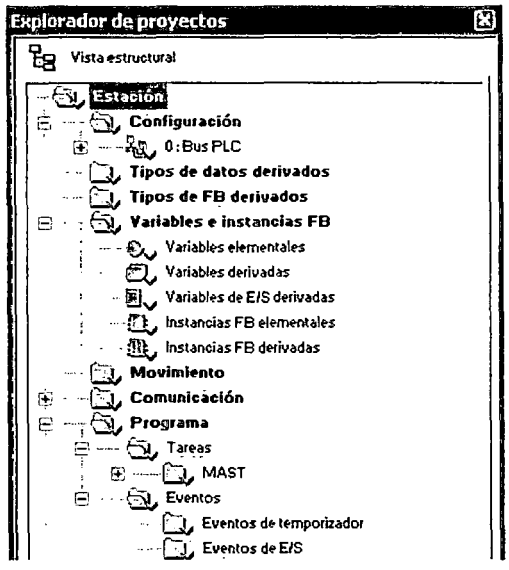


Figura 3.17: Navegador Explorador de Proyectos.

3. El siguiente paso es indicarle al programa qué módulos de entrada y salida tiene instalado el PLC. Para ello se debe desplegar toda la carpeta de configuración, y hacer doble clic en "0: Bus PLC". La ventana que aparece, presentada en la Figura 3.18, muestra un esquema del PLC y de sus ranuras. Por defecto se considera que el PLC sólo dispone de 8 ranuras.

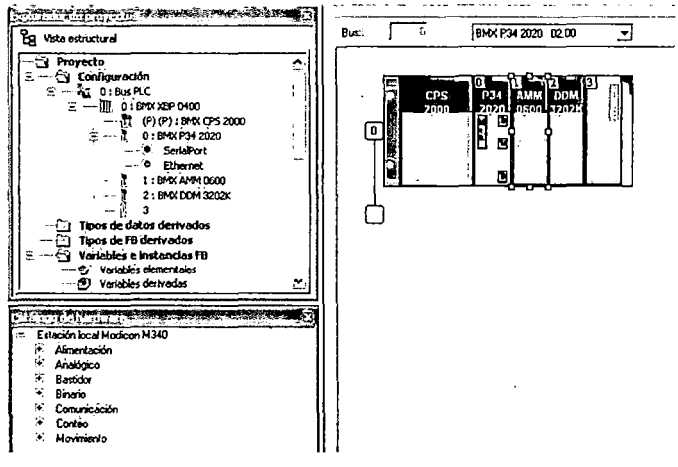


Figura 3.18: Ventana Configuración de Módulos.

4. Una vez configurado los módulos del PLC, se debe configurar la red. Para esto se despliega la carpeta de comunicaciones (en el explorador de proyectos), se hace clic

con el botón derecho sobre redes y se selecciona la opción "nueva red".

En la lista de redes disponibles se escoge "Ethernet", y se elige un nombre para la red. Luego, haciendo doble clic sobre el icono de la nueva red, se hacen los cambios de Configuración de la comunicación, en este caso Ethernet y después validamos, este paso es mostrado en la Figura 3.19.

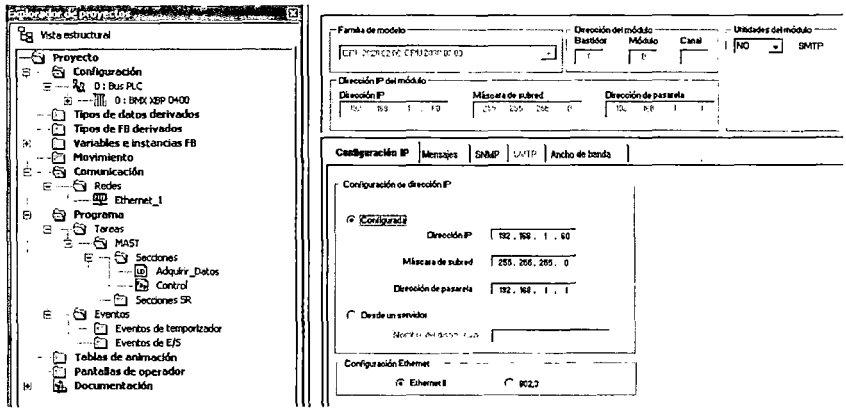


Figura 3.19: Ventana Configuración de Red.

5. Para terminar con la configuración del autómata, aún deben de asociarse las direcciones de lectura y escritura con las variables correspondientes a sensores y actuadores, este paso es mostrado en la Figura 3.20.

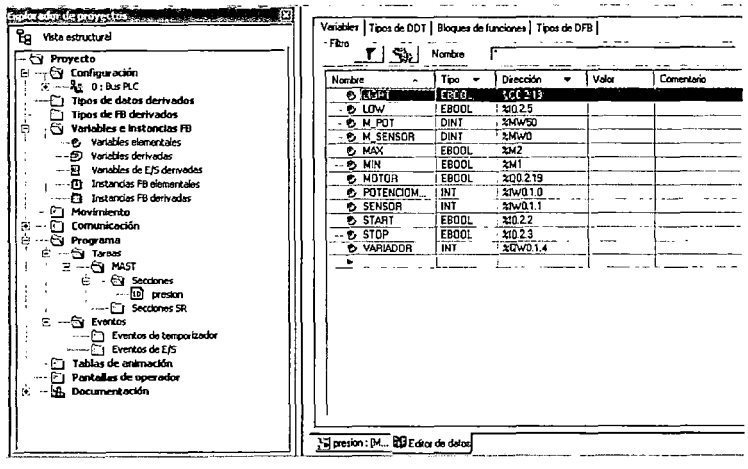


Figura 3.20: Ventana Configuración de Variables.

3.4. Software para el cálculo de la función de transferencia

Para el cálculo de la función de transferencia se usaron dos software licenciados, LabView y Matlab.

3.4.1. NI LabVIEW

Software de National Instruments, LabView (acrónimo de Laboratory Virtual Instrumentation Engineering Workbench) es una plataforma y entorno de desarrollo para diseñar sistemas, con un lenguaje de programación visual gráfico. Recomendado para sistemas de hardware y software de pruebas, control y diseño, simulado o real y embebido, pues acelera la productividad. El lenguaje que usa se llama lenguaje G, donde la G simboliza que es lenguaje Gráfico.

Para la investigación de este proyecto de tesis, se utilizó la versión 2013, así como los siguientes complementos:

- Control Design and Simulation
- Reporter Generation

3.4.2. *Matlab*

Software licenciado de MATHWORKS, es un lenguaje de computación técnica que integra la computación, visualización y programación en un entorno fácil de utilizar. Puede ser utilizado en, matemáticas y computación, desarrollo de algoritmos, adquisición de datos, modelado y simulación. Se utiliza para el análisis de datos y visualización, trazado de gráficos, en el desarrollo científico y la aplicación de gráficos de ingeniería, incluyendo la construcción de la interfaz gráfica de usuario. Aquí, las variables son generalmente representadas como matrices. Se puede utilizar para hacer cualquier manipulación de matrices,

la diferenciación, integración, e incluso laplace, transformadas inversas de Laplace y series Taylor.

Para el desarrollo de este proyecto se utilizaron 2 toolbox de Matlab:

- Sisotool
- System Identification

3.5. Software Matrikon OPC

Matrikon OPC de Conexión Universal Server (UCS) es un único servidor OPC que proporciona conectividad a dispositivos múltiples y protocolos. Este OPC permite a los proveedores proporcionar una conectividad segura a todos los sistemas de control. El servidor Matrikon OPC de Conexión Universal, no requiere licencia para su instalación, reúne a toda la gama de funcionalidad de conectividad en una plataforma de servidor de gran alcance. Estas funcionalidades incluyen:

- OPC Seguridad
- Dispositivos de redundancia de comunicación
- Alarmas y Eventos
- Interoperabilidad máxima
- Modo de simulación
- Nombres de etiquetas definidas por el usuario

3.5.1. Configuración de proyecto en Matrikon

Para la configuración del Matrikon OPC, y de acuerdo a la comunicación que se establece con el PLC MODICOM M340, se configura de la siguiente manera.

- De el grupo de dispositivos seleccionables, escogemos el dispositivo Modbus Ethernet PLC, este paso se observa en la Figura 3.21.

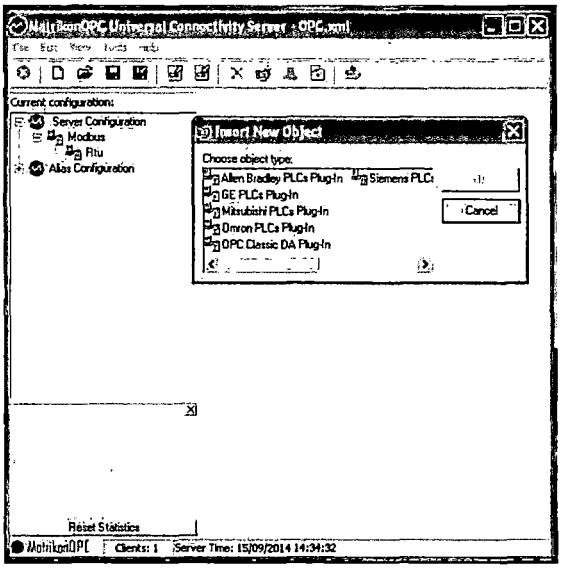


Figura 3.21: Ventana de Matrikon para seleccionar Comunicación Modbus Ethernet.

- Se configura la dirección IP del PLC, como se muestra en la Figura 3.22

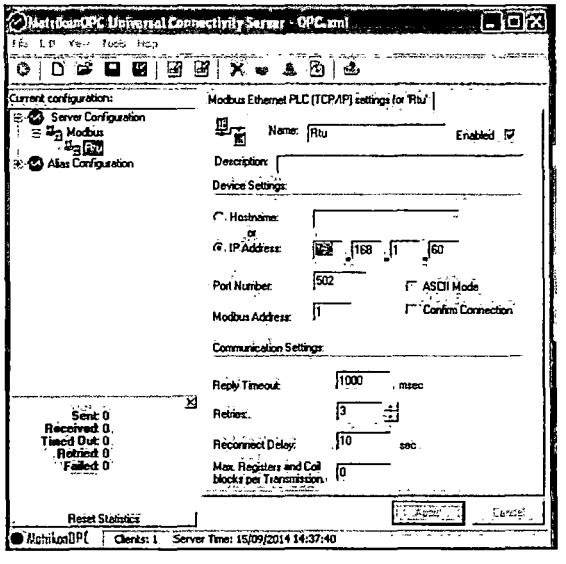


Figura 3.22: Ventana de Matrikon para configurar la dirección IP del PLC a conectar.

CAPÍTULO 4
MODELADO Y SIMULACIÓN DEL SISTEMA

4.1. Adquisición de Datos de la Planta de Presión

Para el proceso de adquirir datos se usó el software Labview de National Instrument, así como el Software Unity Pro para la programación del Controlador.

4.1.1. Programación en Unity Pro

Esta rutina se muestra en la Figura 4.1 ,la secuencia consiste en utilizar las funciones básicas de la planta, es decir ponerla en marcha y parada, además de guardar las variables físicas (SENSOR, VARIADOR, VALVULA, POTENCIOMETRO) en su correspondiente espacio de memoria en el PLC (M_SENSOR, M_VARIADOR, M_VALVULA, M_POT), para facilitar la manipulación de estas.

Esta rutina realizada en lenguaje ladder, permite poner en funcionamiento la planta de presión y leer datos del sensor de presión y controlar la apertura de la válvula proporcional, en las Tablas 4.1 y 4.2 se muestran las variables utilizadas en el proceso.

Nombre de Variable	Tipo	Dirección Física
START	EBOOL	%I0.2.2
STOP	EBOOL	%I0.2.3
LIGHT	EBOOL	%Q0.2.19
BIT_MOTOR	EBOOL	%Q0.2.18
SENSOR	INT	%IW0.1.1
VARIADOR	INT	%QW0.1.4
VALVULA	INT	%QW0.1.5

Tabla 4.1: Direcciones Físicas del PLC utilizadas.

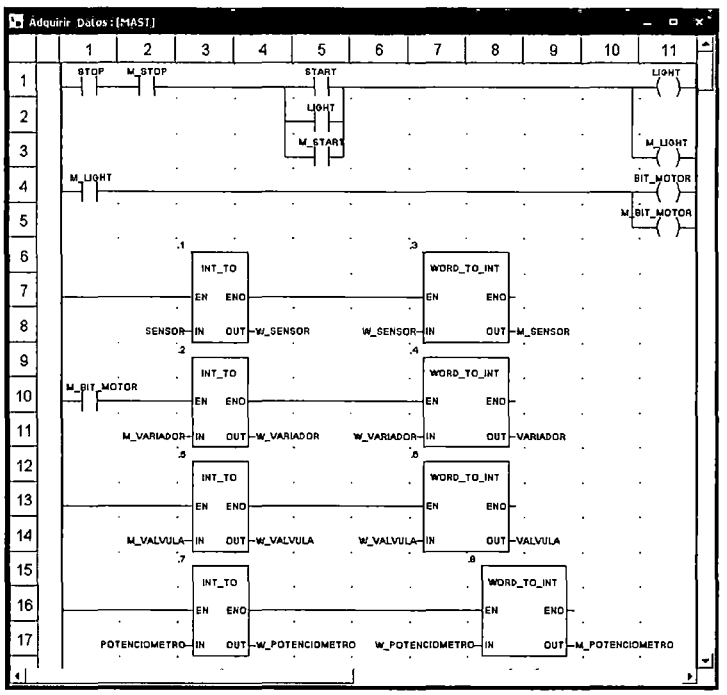


Figura 4.1: Rutina realizada en Lenguaje Ladder para la Adquisición de Datos.

Nombre de Variable	Tipo	Dirección memoria
M_START	EBOOL	%M0
M_STOP	EBOOL	%M1
M_LIGHT	EBOOL	%M2
M_BIT_MOTOR	EBOOL	%M3
M_SENSOR	INT	%MW1
M_VARIADOR	INT	%MW0
M_VALVULA	INT	%MW4

Tabla 4.2: Espacio de memoria del PLC utilizadas.

4.1.2. Programación en Labview

La secuencia consiste en dar la orden de arranque y parada mediante los botones START y STOP respectivamente y un indicador visual de que el sistema está funcionando mediante la lámpara LIGHT, permite leer los datos tomados por el sensor de presión, así también es posible controlar la apertura de la válvula mediante el *Slide* "Válvula" y controlar la velocidad de giro de la bomba mediante el *Slide* "Variador", el cual se mantiene constante al valor de 10050 que equivale a que la bomba funcione a 60 Hz. Las variables SENSOR y VALVULA son almacenadas y exportadas en un reporte en formato Excel. Estas

variables han sido enlazadas a su correspondiente variable en el registro del PLC mediante la opción "Data Binding" de Labview, como se muestra en la Tablas 4.3:

Variable_LabView	Variable_PLC	Dirección Física
START	M_START	$Modbus.Rtu,0:1$
STOP	M_STOP	$Modbus.Rtu,0:2$
LIGHT	M_LIGHT	$Modbus.Rtu,0:1$
VARIADOR	M_VARIADOR	$Modbus.Rtu,4:1$
SENSOR	M_SENSOR	$Modbus.Rtu,4:2$
VALVULA	M_VALVULA	$Modbus.Rtu,4:5$

Tabla 4.3: Direcciones Físicas del PLC utilizadas.

En la Figura 4.2 mostramos el panel frontal de programación en labview para adquirir datos del sensor de presión y controlar la válvula proporcional, y exportarlo en un documento de Excel. En la Figura 4.3 se muestra la programación en LabView, realizado en lenguaje Gráfico, para adquirir datos del sensor de presión y controlar la válvula proporcional, y exportarlo en un documento de Excel.

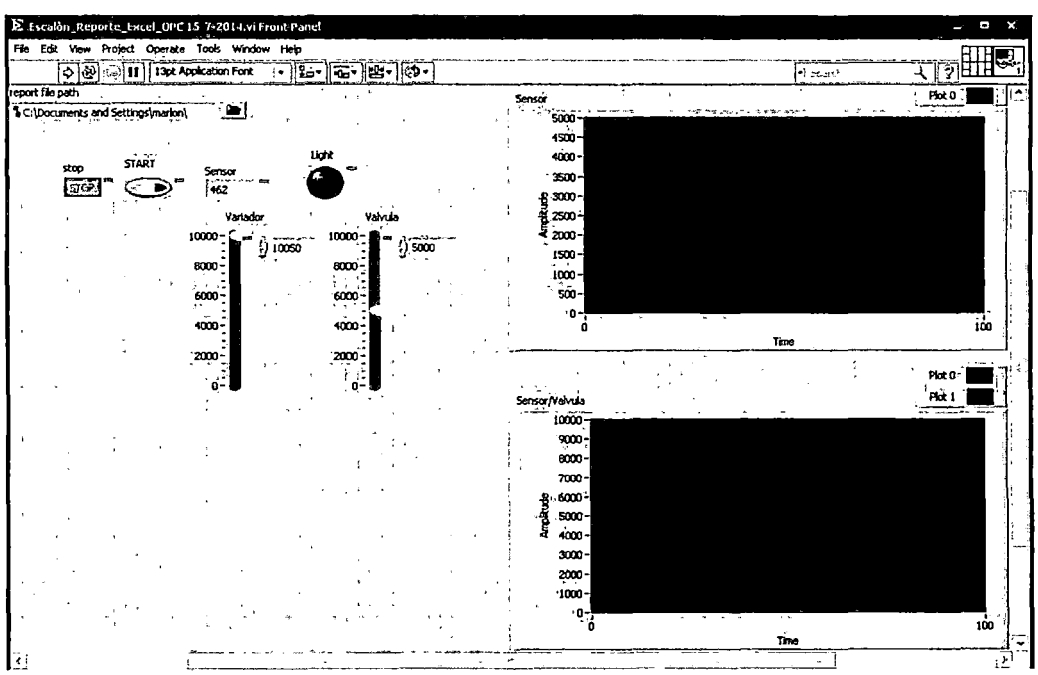


Figura 4.2: Panel frontal de programación en LabView para adquisición de datos.

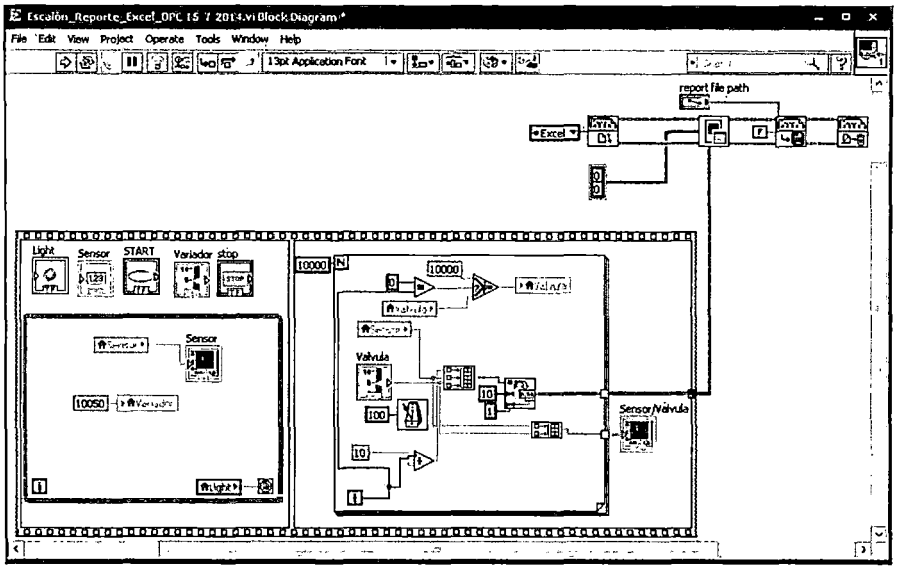


Figura 4.3: Diagrama de Bloques de programación en LabView para adquisición de datos.

4.2. Procesamiento de datos

Para el procesamiento de datos se acondicionó la señal de entrada a la válvula proporcional para emular un comportamiento de una válvula normalmente abierta, siendo en realidad normalmente cerrada. Para esto se basó en lo siguiente:

Valor	Comportamiento
0	Totalmente Cerrada
10000	Totalmente Abierta

Tabla 4.4: Comportamiento normal de la Válvula Proporcional.

Lo que sería:

$$y = 10000 - x \tag{4.1}$$

Donde:

y: Señal Procesada

x: Señal inicial

4.2.1. Programación en Matlab

En un editor Script de Matlab se procedió a plasmar esto en una nueva rutina, la cual carga los datos almacenados en Excel, realiza el procesamiento, y los guarda en las variable "u" en el caso de que sea una entrada y "y" en el caso de que se trate de una salida.

Esta Programación se muestra a continuación:

```
A = xlsread('Prueba_01.xlsx');%Respuesta de la planta a señal binaria aleatoria 0-5000
B = xlsread('Prueba_02.xlsx');%Respuesta de la planta a señal binaria aleatoria 0-7500
C = xlsread('Prueba_03.xlsx');%Respuesta de la planta señal escalonada aleatoria
u1 = 10000 - A(:,3)';%Procesamiento de la señal de entrada
u2 = 10000 - B(:,3)';%Procesamiento de la señal de entrada
u3 = 10000 - C(:,3)';%Procesamiento de la señal de entrada
y1 = A(:,2);Respuesta a señal binaria aleatoria 0-5000
y2 = B(:,2);Respuesta a señal binaria aleatoria 0-7500
y3 = C(:,2);Respuesta a señal escalonada aleatoria
figure(1)
subplot(3,1,1)
plot(A(:,3))
title('Señal PRBS')
xlabel('Tiempo (ms)')
ylabel('Amplitud')
subplot(3,1,2)
plot(u1)
title('Señal PRBS procesada')
xlabel('Tiempo (ms)')
ylabel('Amplitud')
subplot(3,1,3)
plot(y1)
title('Respuesta')
xlabel('Tiempo (ms)')
ylabel('Amplitud')
figure(2)
subplot(3,1,1)
plot(B(:,3))
title('Señal PRBS')
xlabel('Tiempo (ms)')
ylabel('Amplitud')
subplot(3,1,2)
plot(u2)
title('Señal PRBS procesada')
xlabel('Tiempo (ms)')
ylabel('Amplitud')
subplot(3,1,3)
plot(y2)
title('Respuesta')
xlabel('Tiempo (ms)')
ylabel('Amplitud')
figure(3)
subplot(3,1,1)
plot(C(:,3))
```

```
title('Señal escalonada aleatoria')
xlabel('Tiempo (ms)')
ylabel('Amplitud')
subplot(3,1,2)
plot(u3)
title('Señal escalonada aleatoria procesada')
xlabel('Tiempo (ms)')
ylabel('Amplitud')
subplot(3,1,3)
plot(y3)
title('Respuesta')
xlabel('Tiempo (ms)')
ylabel('Amplitud')
```

En la Figura 4.4 se muestra la respuesta obtenida de la planta a un señal binaria aleatoria 0 y 5000, lo que equivale a que la válvula se encuentre totalmente abierta y 50 % abierta respectivamente. En la Figura 4.5 se muestra la respuesta obtenida de la planta a una señal binaria aleatoria 0-7500, lo que equivale a que la válvula se encuentre totalmente abierta y 25 % abierta respectivamente. En la Figura 4.6 se muestra la respuesta obtenida de la planta a una señal escalonada aleatoria.

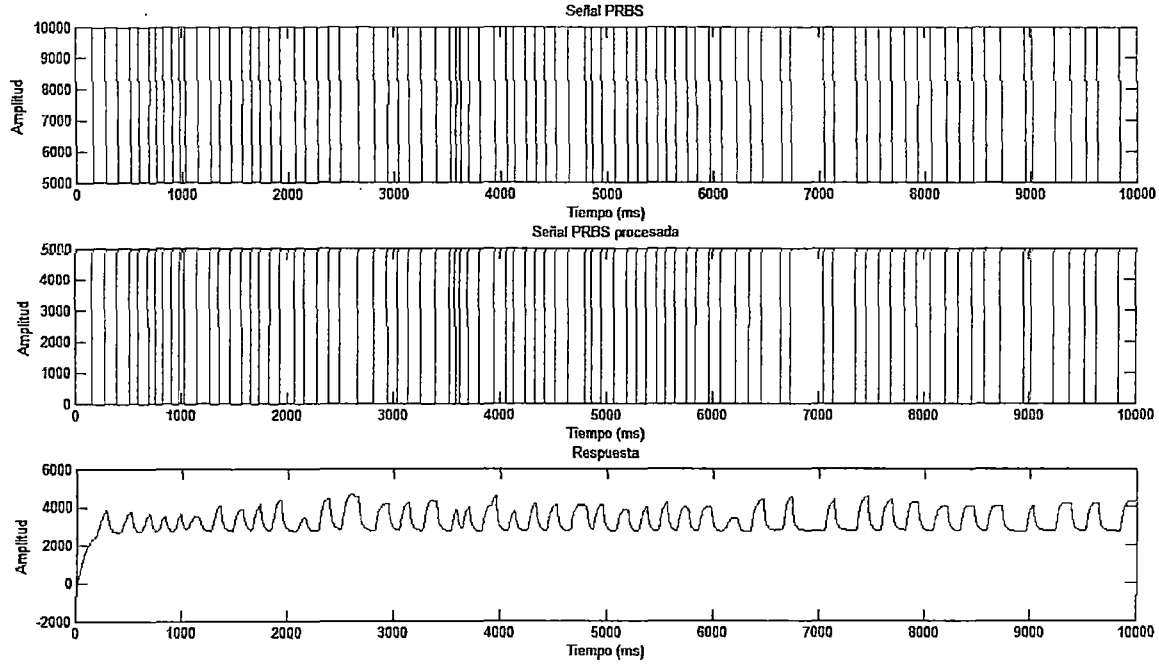


Figura 4.4: Respuesta del sistema a una entrada binaria aleatoria 0 - 5000.

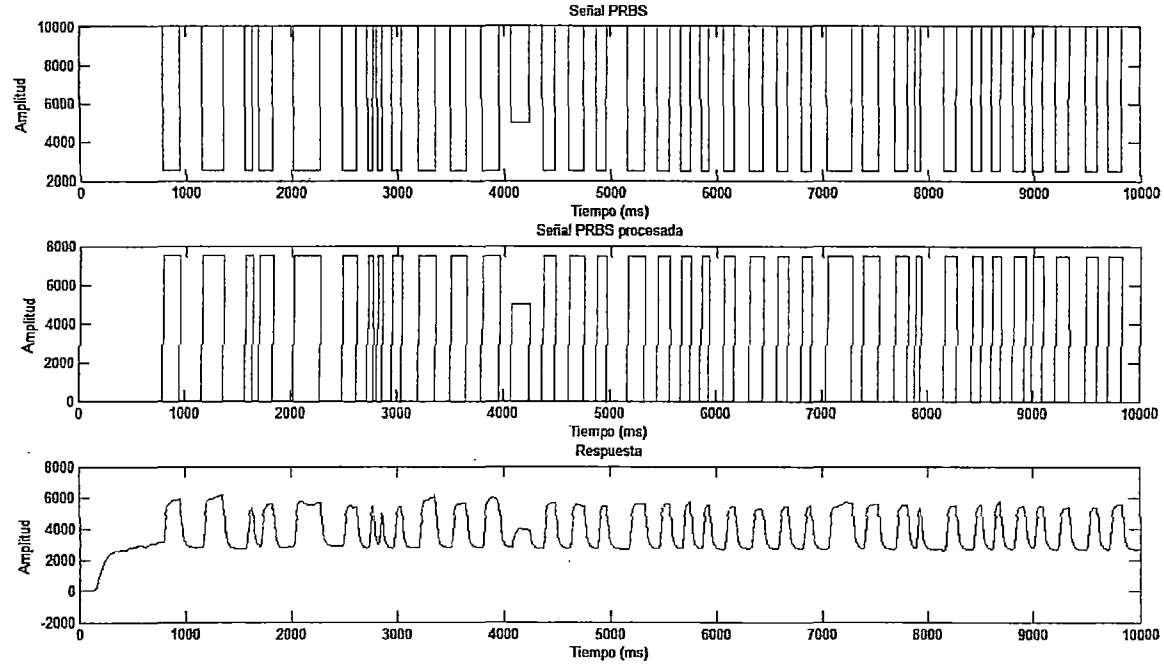


Figura 4.5: Respuesta del sistema a una entrada binaria aleatoria 0 - 7500.

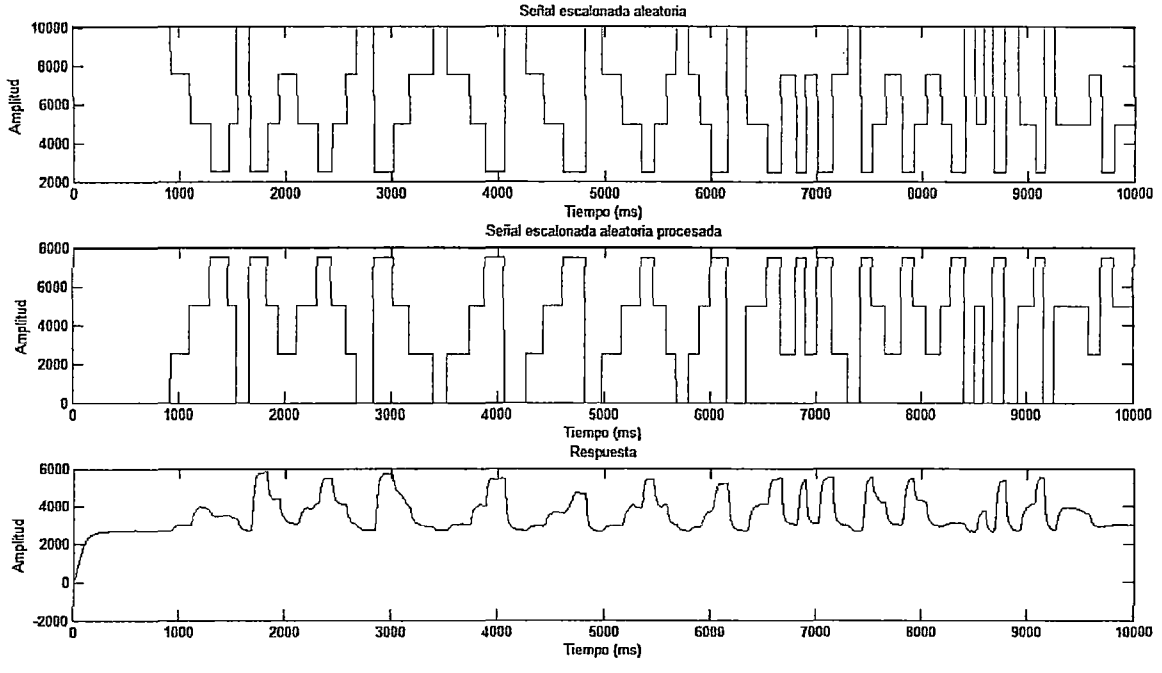


Figura 4.6: Respuesta del sistema a una entrada escalonada aleatoria.

4.3. Estimación de la Función de Transferencia de la Planta de Presión

Para la obtención de la función de transferencia se utilizó la herramienta de Matlab *System Identification*, para el cual es necesario cargar los datos obtenidos de la planta, es decir una señal de entrada con su correspondiente salida.

Así se probó con los tres grupos de datos obtenidos, se obtuvieron las respectivas funciones de transferencia, y se comprobó si al ser excitada con la misma señal de entrada respondía de manera similar a la planta real, siendo la función de transferencia obtenida con el segundo grupo de datos la que más se asemejó, a continuación se describe el proceso:

a) **Paso 1:** En el menú "Import data", se selecciona la opción "Data Time", que nos permitirá seleccionar nuestro grupo de datos, se coloca ademas un "Tiempo de Muestreo" de 0.1 seg, que es el mismo con los que se obtuvieron los datos. En la Figura 4.7 se muestra la ventana principal con los datos experimentales importados en la herramienta System identification.

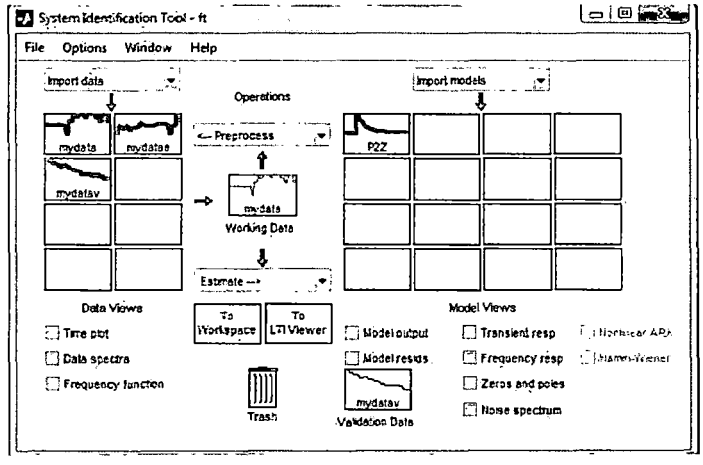


Figura 4.7: Ventana principal de Herramienta System Identification.

b) **Paso 2**: En el menú "Process", seleccionamos la opción "Select", que nos perimitirá seleccionar que grupo de datos queremos utilizar, seleccionaremos la primera mitad para la Estimación del modelo, y la segunda mitad para el proceso de Validación. En la Figura

4.8 se muestran los datos importados originales en la herramienta System Identification, en la Figura 4.9 se muestran los datos de estimación, que fueron obtenidos seleccionando la primera mitad de los datos principales, y en la Figura 4.10 se muestra los datos de validación, que fueron obtenidos seleccionando la segunda mitad de los datos principales.

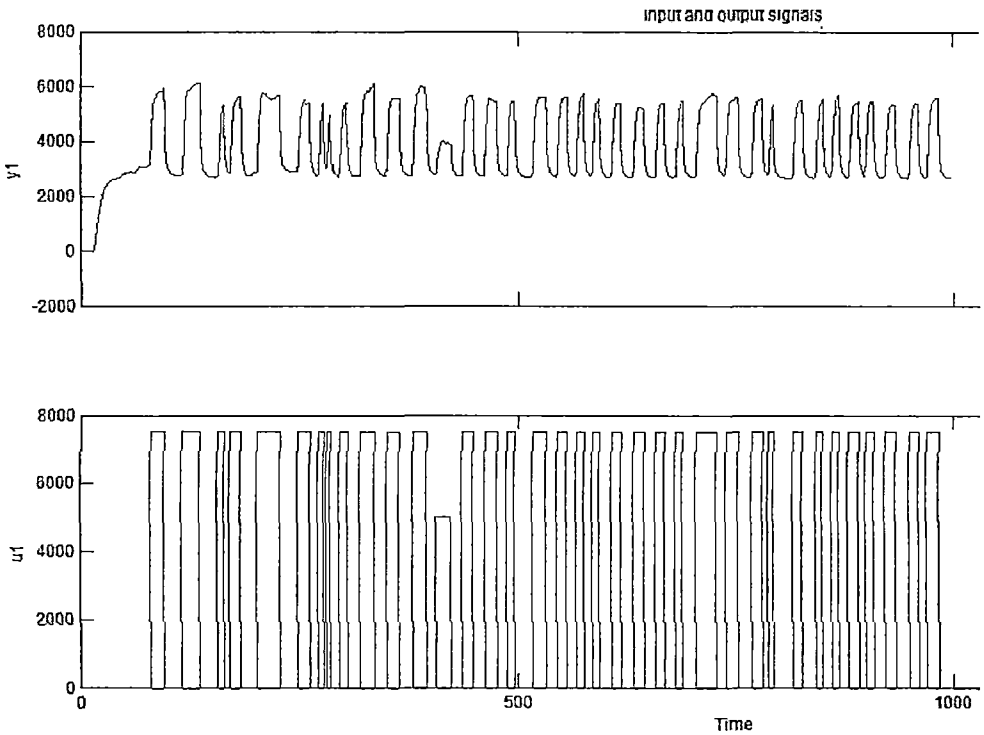


Figura 4.8: Datos originales, obtenidos de la planta real.

c) **Paso 3:** En el menú "Estimate", se selecciona la opción "Process Model" y seleccionamos un modelo con 2 Polos y 1 Zero reales. En la Figura 4.11 se muestra el modelo utilizado para describir el comportamiento de la planta. Dando click en "Estimate" se obtiene la siguiente Función de Transferencia:

$$G(s) = K_p \frac{1 + T_z s}{(1 + T_{p1} s)(1 + T_{p2} s)} \tag{4.2}$$

Donde: K_p = 1.0244

T_{p1} = 478.36

T_{p2} = 2.5272

T_z = 188.66

Que equivale a:

$$G(s) = \frac{0{,}1599s + 0{,}0008474}{s^2 + 0{,}3978s + 0{,}0008272} \tag{4.3}$$

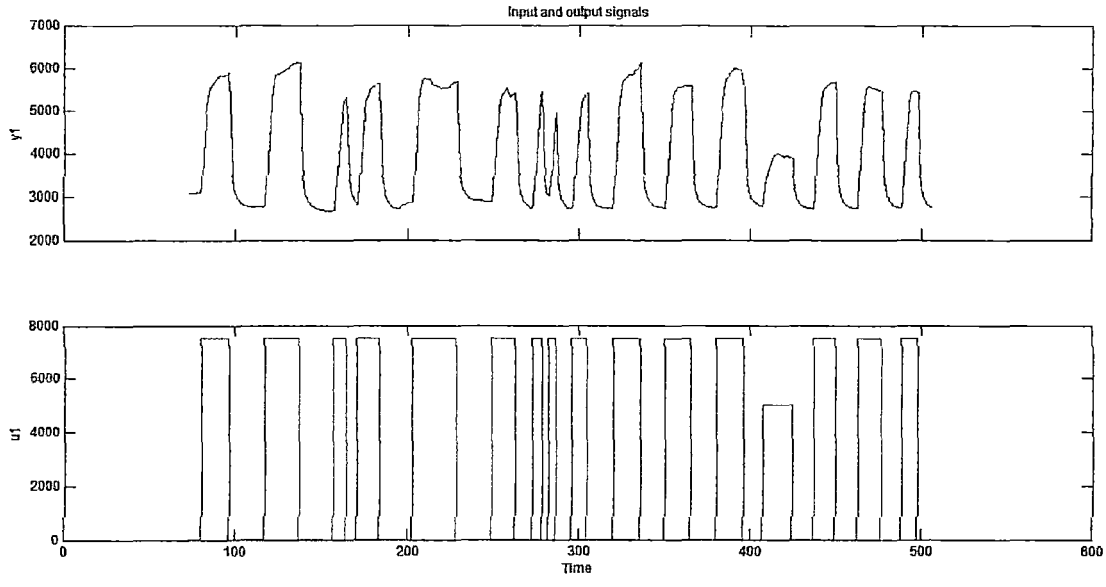


Figura 4.9: Datos seleccionados para la estimación del modelo.

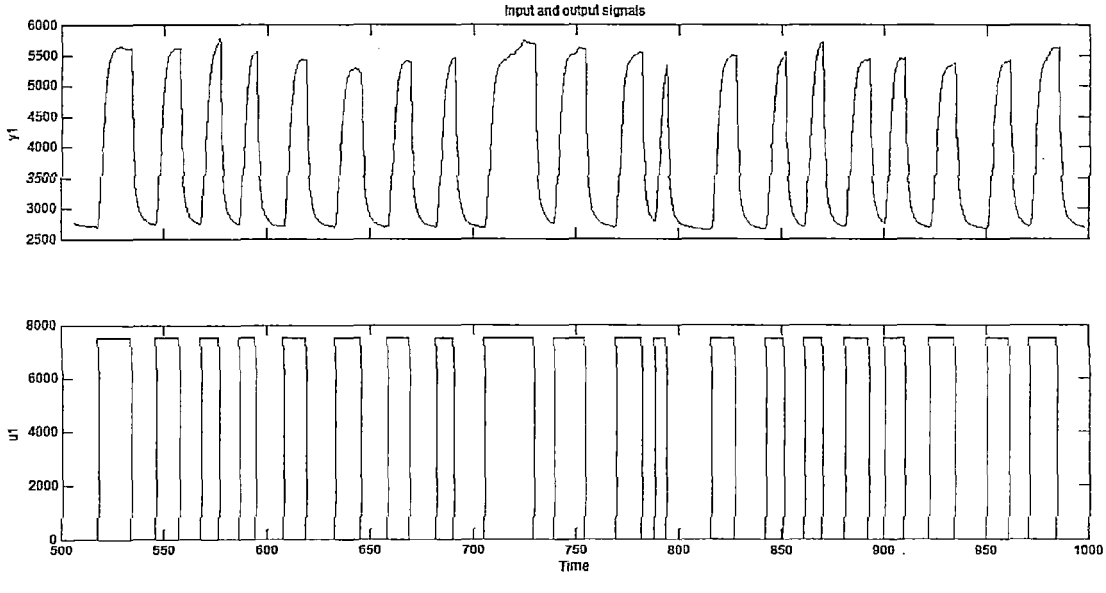


Figura 4.10: Datos seleccionados para la validación del modelo.

d) **Paso 4:** Con la opción "Model Output" podemos visualizar cuanto se ajusta el modelo estimado a la respuesta real del sistema mediante la opción "Best Fits", que nos muestra que el modelo obtenido se asemeja en un 77.18 %. La Figura 4.12 muestra la comparación entre la señal real y los datos obtenido por el modelo.

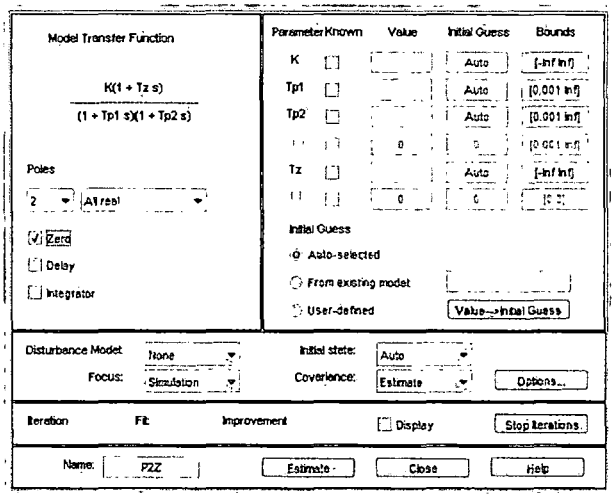


Figura 4.11: Modelo utilizado para la estimación de la Función de Transferencia.

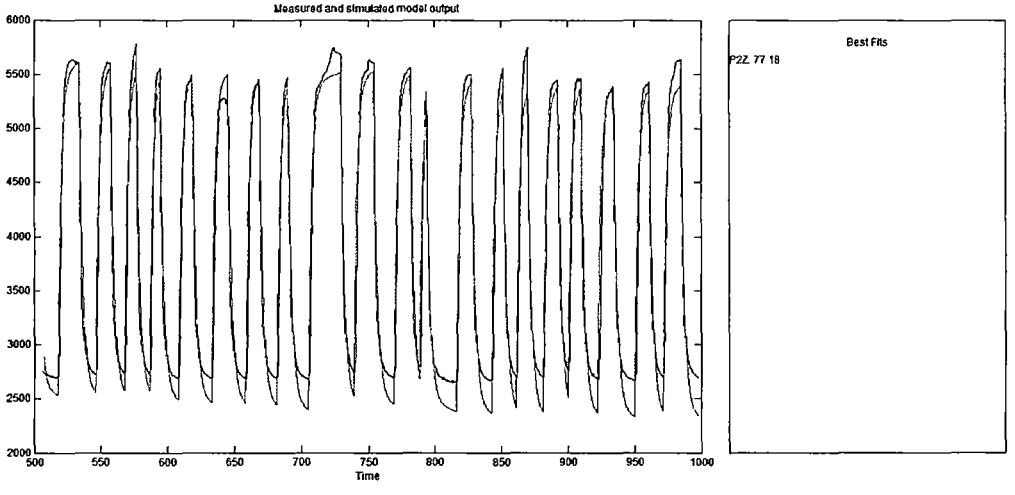


Figura 4.12: Comparación entre la señal real y datos del modelo estimado.

e) **Paso 5:** La opción "LTI Viewer" permite visualizar la respuesta del modelo a una entrada escalón unitario, como se muestra en la Figura 4.13, mientras que la opción "Zeros and Poles" permite visualizar el Lugar Geométrico de las raíces del sistema, mostrado en la Figura 4.14.

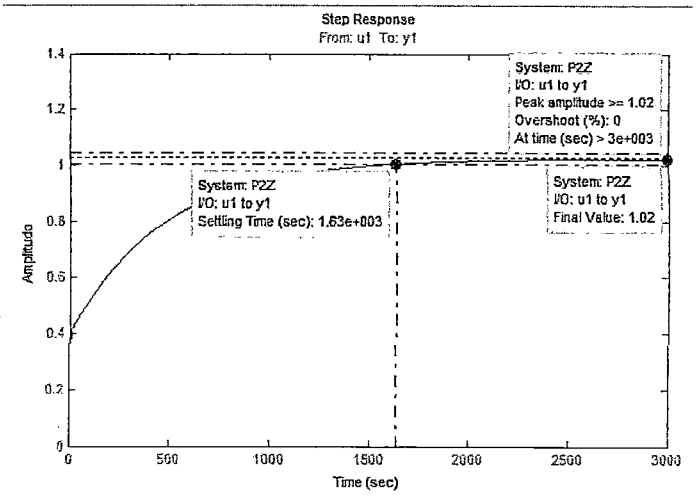


Figura 4.13: Respuesta del modelo estimado al escalón unitario.

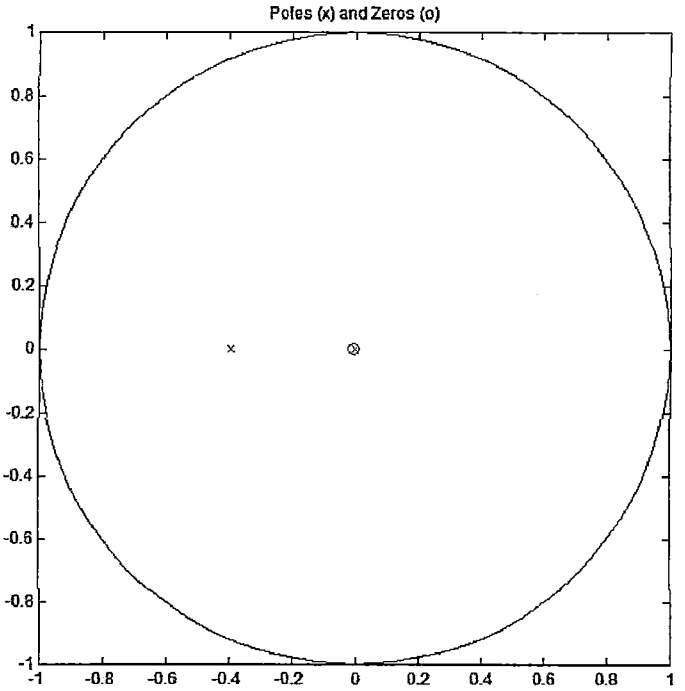


Figura 4.14: Lugar Geométrico de las Raíces del modelo estimado.

4.4. Simulación del Modelo

Se realizó una rutina en el editor Script de Matlab para comparar gráficamente la respuesta real del sistema y la respuesta simulada del modelo obtenido, para ello se hizo

uso el comando *lsim*, que permite simular un sistema a una respuesta dada, este código se muestra a continuación:

```
H = tf([0.1599 0.0008474],[1 0.39780 0.0008272]);%Funcion de transferencia estimada
t = 0:0.1:999.9;%Tiempo de Simulacion
ys1 = lsim(H,u1,t)';%Respuesta del sistema a señal binaria aleatoria 0-5000
ys2 = lsim(H,u2,t)';%Respuesta del sistema a señal binaria aleatoria 0-7500
ys3 = lsim(H,u3,t)';%Respuesta del sistema a señal escalonada aleatoria
figure(1)
plot(t,u1,t,y1,t,ys1)
legend('Señal de entrada','Respuesta Real','Respuesta Simulada')
figure(2)
plot(t,u2,t,y2,t,ys2)
legend('Señal de entrada','Respuesta Real','Respuesta Simulada')
figure(3)
plot(t,u3,t,y3,t,ys3)
legend('Señal de entrada','Respuesta Real','Respuesta Simulada')
```

Los resultados de este programa se muestran en las Figuras 4.15, 4.16 y 4.17.

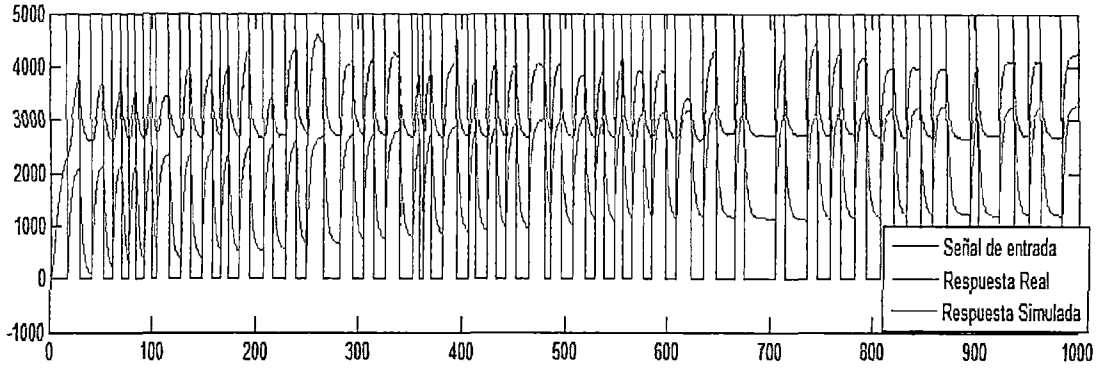


Figura 4.15: Respuesta del sistema a señal binaria aleatoria 0-5000.

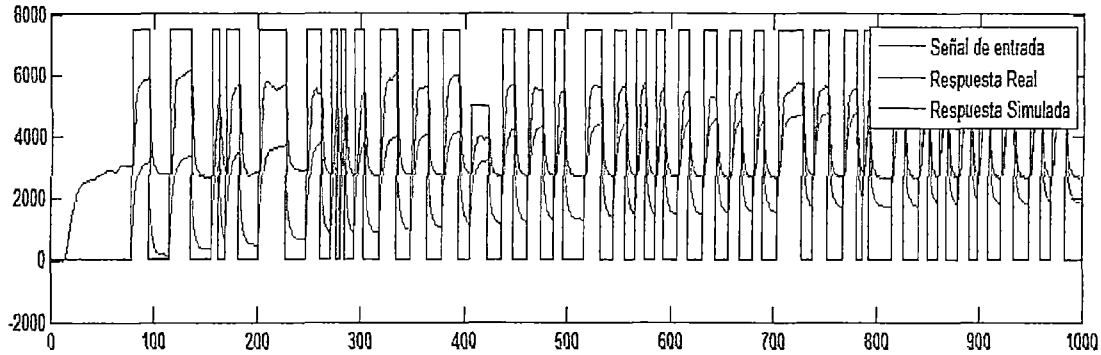


Figura 4.16: Respuesta del sistema a señal binaria aleatoria 0-7500.

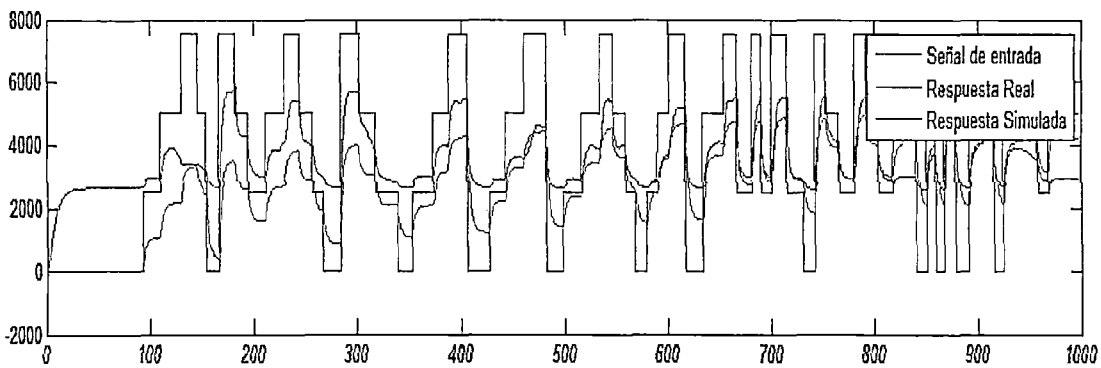


Figura 4.17: Respuesta del sistema a señal escalonada aleatoria.

4.5. Diseño del Controlador

4.5.1. Herramienta de Matlab: SISOTOOL

Para diseñar el controlador se utilizó la herramienta de Matlab $SISOTOOL$, que permite diseñar un compensador por distintos métodos, entre ellos lugar geométrico de las raíces, para el diseño de nuestro controlador hemos hecho uso de este método, variando experimentalmente las raíces del sistema hasta obtener los requerimientos previamente planteados, sobre pico menor de 20 %, tiempo de estabilización menor a 40 segundos y error de estado estacionario nulo.

La Figura 4.18 muestra la ventana "Compensator Editor", de la herramienta $SISO$-

$TOOL$, en donde es posible agregar Ceros y Polos para modificar el lugar geométricos de las raíces del sistema.

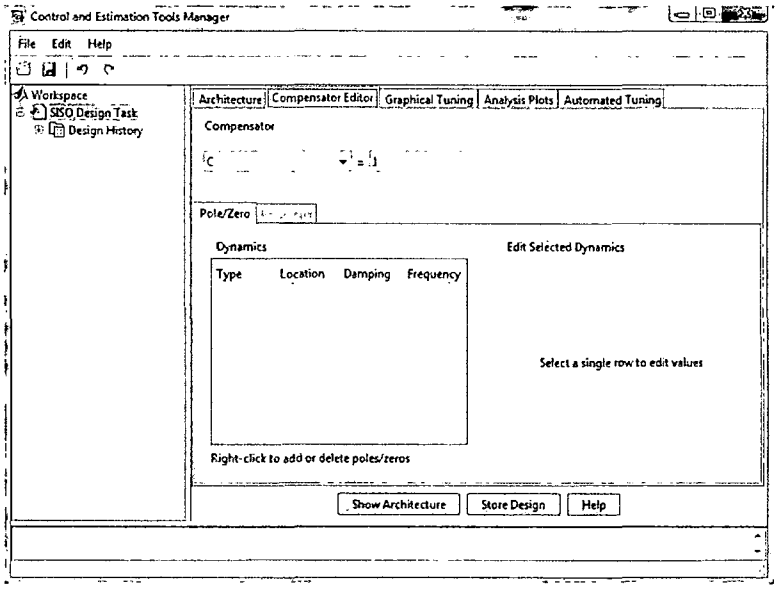


Figura 4.18: Ventana "Compensator Editor" de la herramienta SISOTOOL.

La Figura 4.19 muestra el lugar geométrico de las raíces del sistemaen lazo abierto, en donde la ganancia puede ser modificada para obtener los requerimientos planteados. Así también pueden ser agregados compensadores para cumplir las especificaciones de comportamiento.

Controlador Proporcional (P)

Para el diseño del controlador tipo Proporcional, basta con modificar la ganancia del compensador C, hasta encontrar el comportamiento del sistema que más se aproxima a los requerimientos, Figura 4.20.

En la Figura 4.21 la línea vertical es el requerimiento de tiempo, y las líneas oblicuas son el requerimiento de sobrepico, mientras los polos deseados en lazo cerrado (puntos color fucsia) estén cerca o sobre la intersección de estas líneas, los requerimientos serán cumplidos.

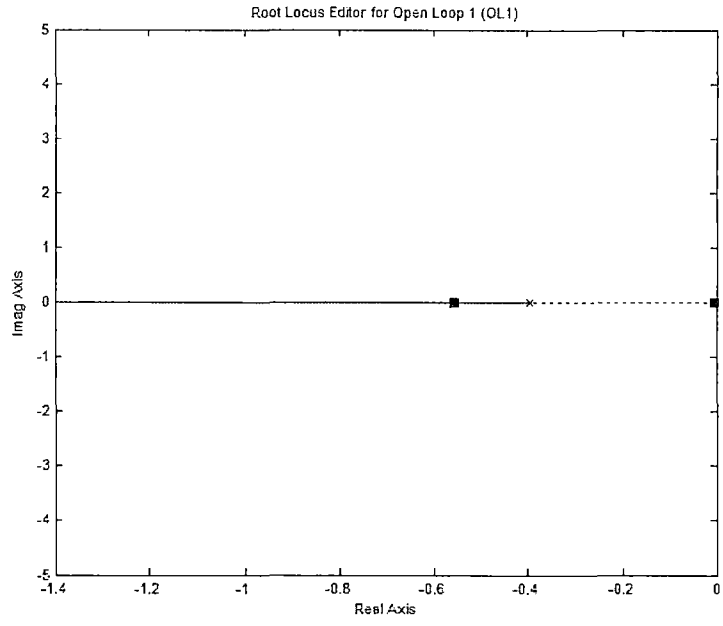


Figura 4.19: Lugar Geométrico de las Raíces del Sistema.

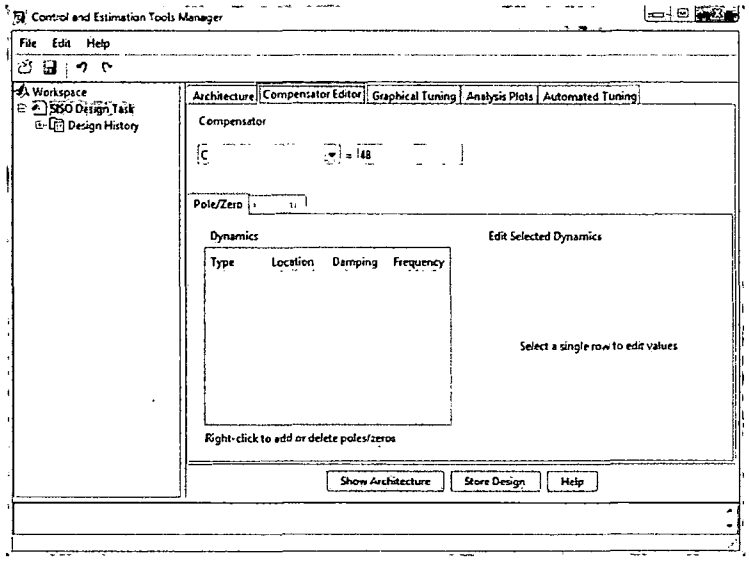


Figura 4.20: Ventana "Compensator Editor" de la herramienta SISOTOOL para controlador tipo P.

Se puede apreciar que el lugar geométrico de las raíces de este sistema con compensador Proporcional no hace posible ubicar los polos deseados cerca a la intersección de las líneas de requerimientos, por lo que estos no se cumplen con cualquier valor de

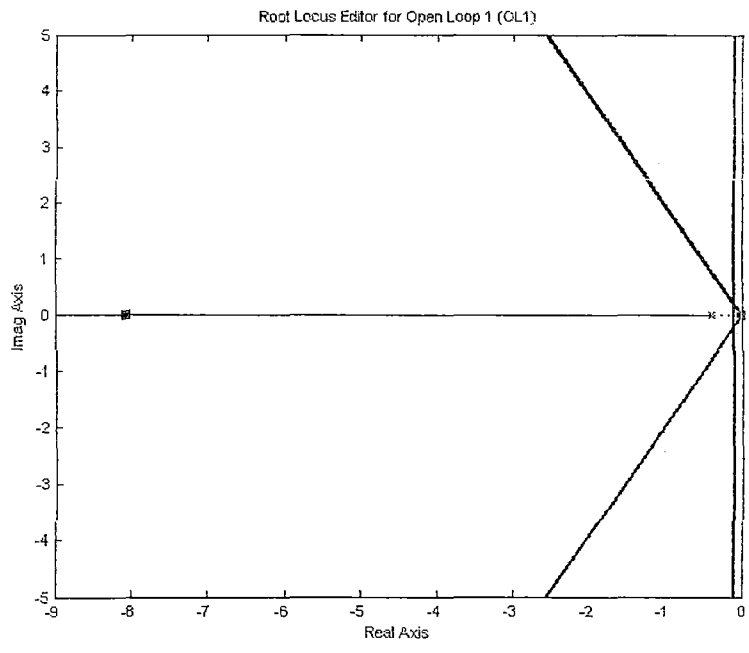


Figura 4.21: Lugar Geométrico de las Raíces del Sistema con el Controlador tipo P.

ganancia.

En la Figura 4.22 se ve la respuesta al escalón unitario del sistema con compensador proporcional de ganancia 48, se aprecia que no se cumplen los requerimientos planteados.

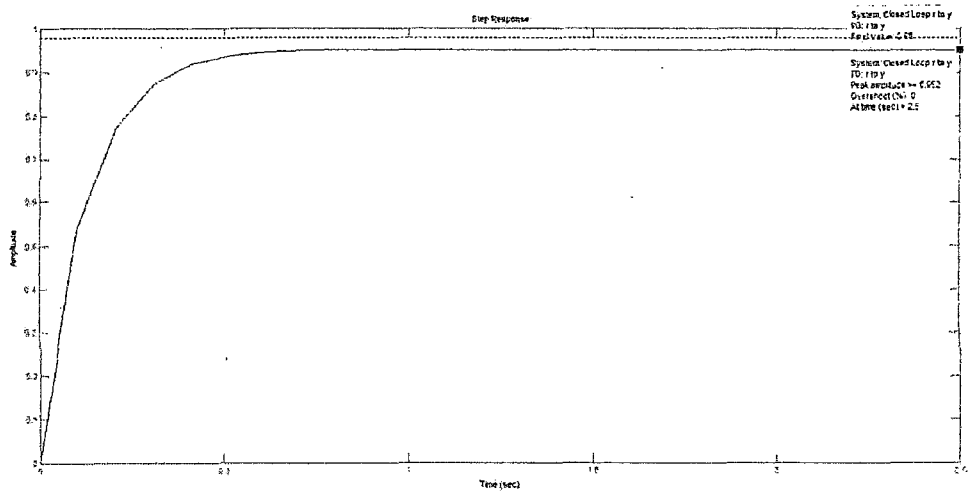


Figura 4.22: Respuesta al escalón unitario del Sistema con el Controlador tipo P.

Controlador Proporcional Integral (PI)

En la Figura 4.23 se aprecia que al compensador se han añadido un Integrador (Polo en el origen) y un Cero, esta combinación hace posible que el compensador tenga la forma básica del controlador PI.

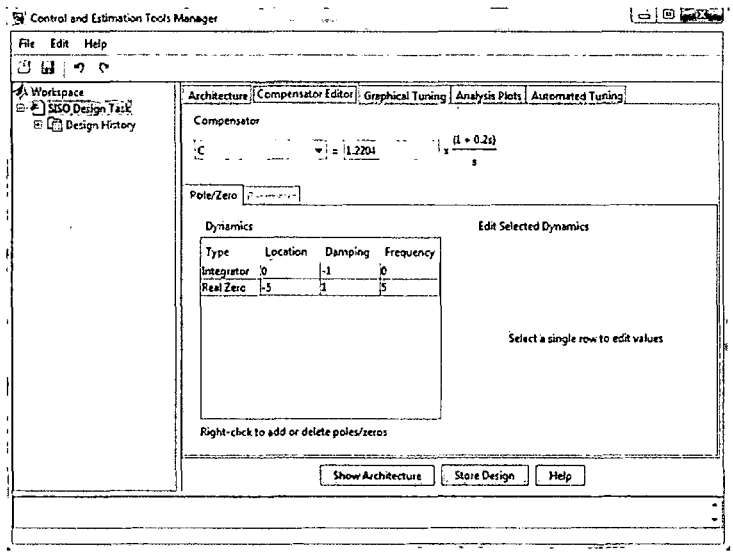


Figura 4.23: Ventana "Compensator Editor" de la herramienta SISOTOOL con controlador tipo PI.

En la Figura 4.24 se puede ver que el lugar geométrico a tomado la forma de un círculo alrededor del Cero Real, lo que hace posible que los polos deseados en lazo cerrado puedan ubicarse cerca a la intersección de las líneas que indican el requerimiento del sistema.

En la Figura 4.25 podemos apreciar que los requerimientos planteados son cumplidos, presenta un error de estado estacionario nulo, un sobrepico de 17.9 % y un tiempo de estabilización de 17.4 s.

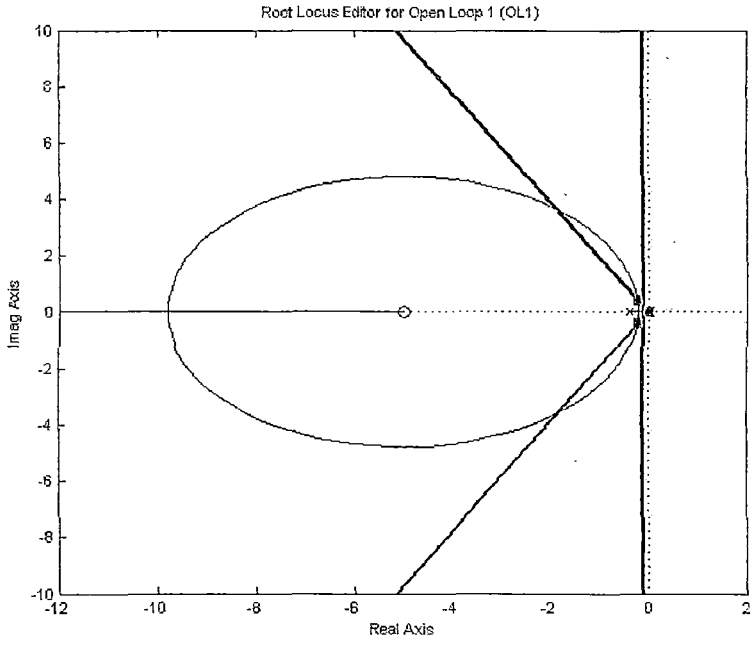


Figura 4.24: Lugar Geométrico de las Raíces del Sistema con el Controlador tipo PI.

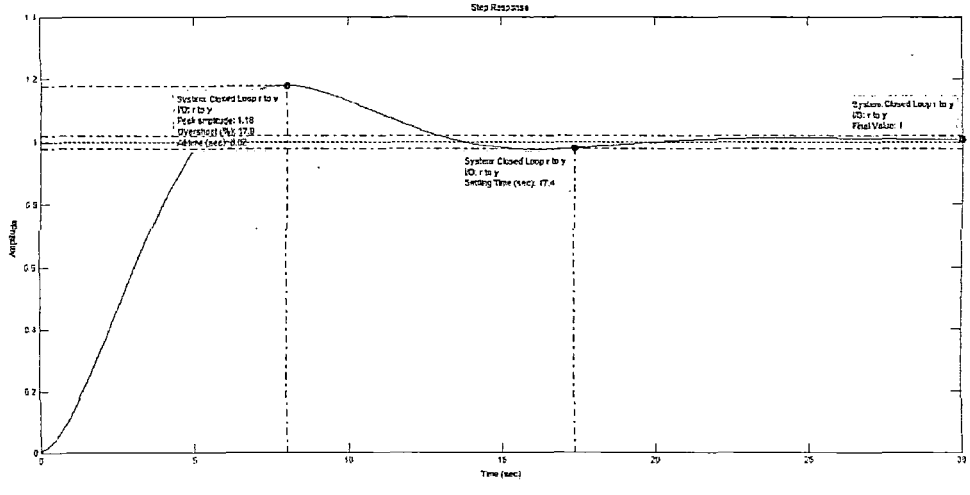


Figura 4.25: Respuesta al escalón unitario del Sistema con el Controlador tipo PI.

Controlador Proporcional Integral Derivativo (PID)

Para el diseño del controlador PID, al compensador se agregaron un Integrador (Polo en el origen), y 2 Ceros distintos, como se aprecia en la Figura 4.26, mientras mas

grande sea el valor de los ceros, mas se aproxima al origen el lugar de la raíces. Esto permite ubicar los polos deseados en lazo cerrado cerca a la intersección de las líneas de requerimiento, como se aprecia en la Figura 4.27.

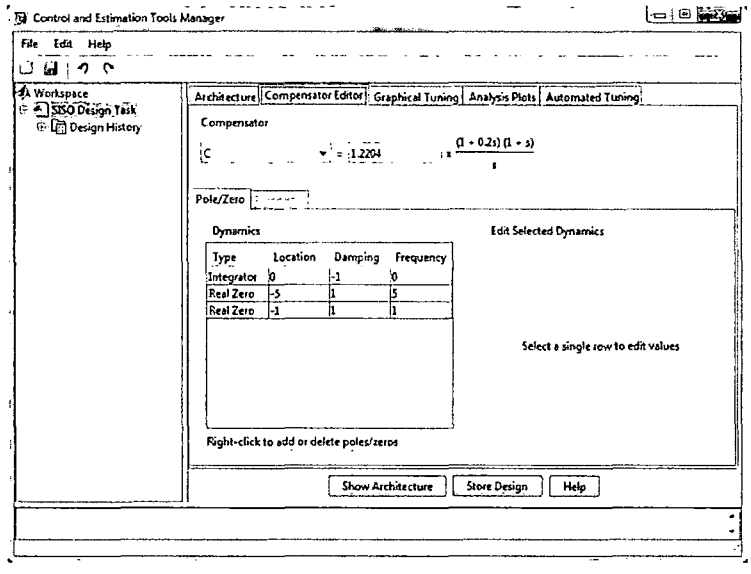


Figura 4.26: Ventana "Compensator Editor" de la herramienta SISOTOOL con Controlador tipo PID.

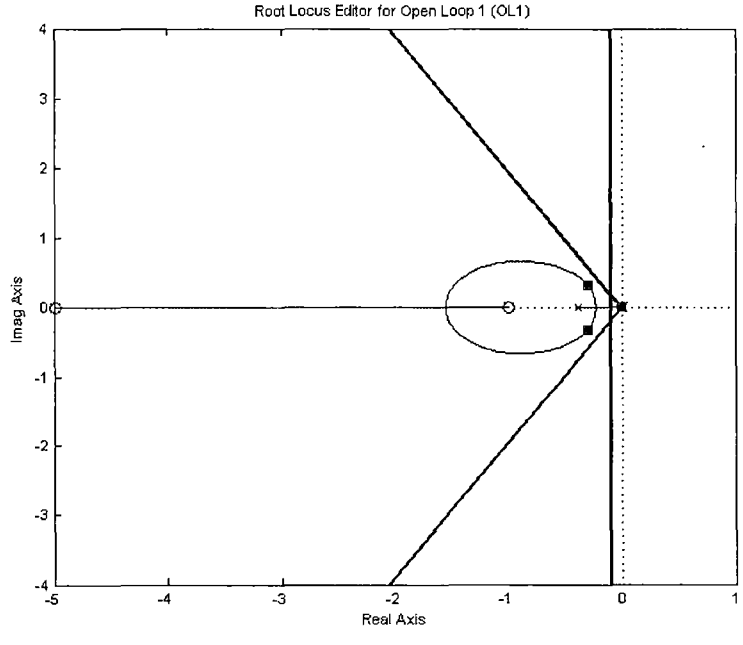


Figura 4.27: Lugar Geométrico de las Raíces del Sistema con el Controlador tipo PID.

En la Figura 4.28 se aprecia la respuesta al escalón unitario de sistema con com-

pensador PID, en donde el error de estado es estacionario es 0, el porcentaje de sobrepico a disminuido a 6.05 %, y el tiempo de estabilización es de 13.5 s, cumpliendo con los esperado al agregar la acción derivativa.

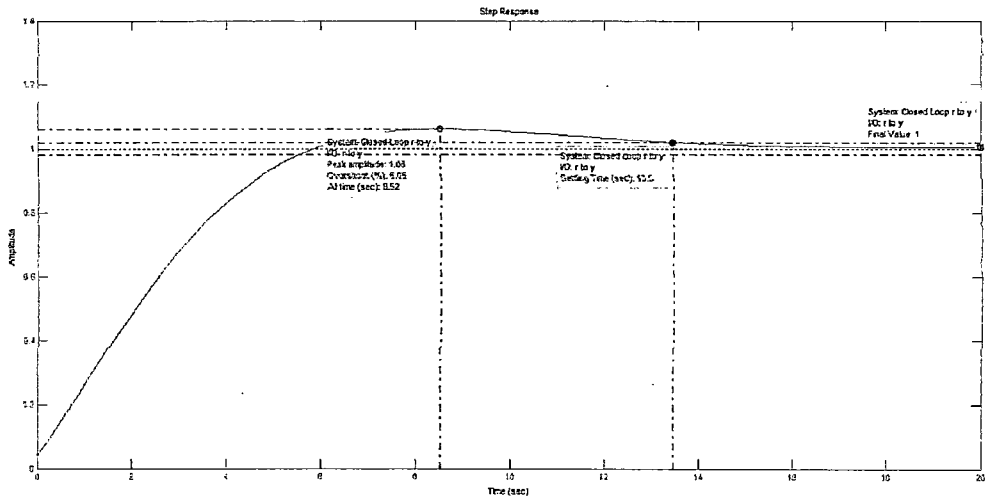


Figura 4.28: Respuesta al escalón unitario del Sistema con el Controlador tipo PID.

4.5.2. Programación del Controlador en Unity Pro

Para la simulación del controlador se realizó una rutina en lenguaje FDB (Diagrama de Bloques de Funciones), en donde se utiliza el bloque de PIDFF como controlador. En la Figura 4.29 se muestra la secuencia en Lenguaje FDB del controlador.

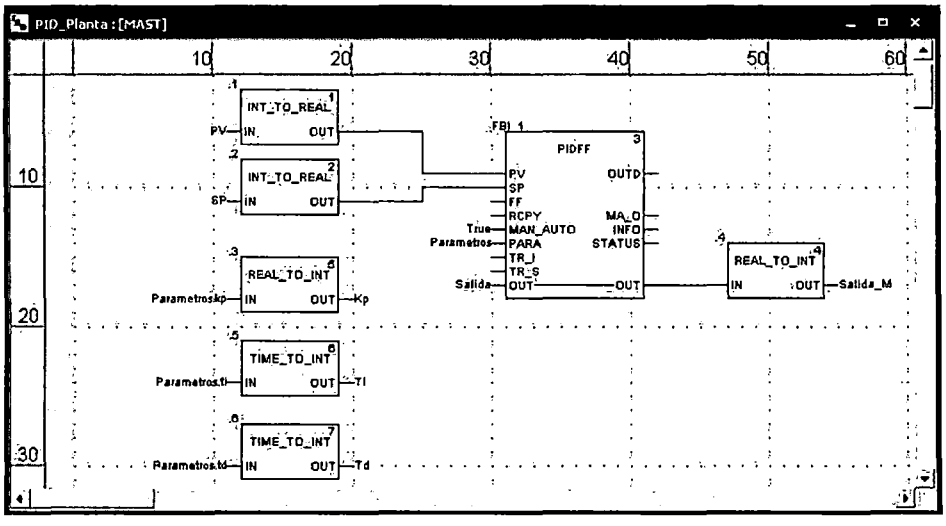


Figura 4.29: Respuesta al escalón unitario del Sistema con el Controlador.

4.5.3. Programación del Controlador en Labview

Se realizó una secuencia en LabView para simular la función de transferencia obtenida con la herramienta *System Identification*, y el controlador obtenido con la herramienta *SISOTOOL* de Matlab, para eso se usa el toolkit Control Disign and Simulation de Matlab, que permite el uso de variables en espacio "s" y realizar simulación en tiempo real.

La Figura 4.30 muestra el Panel Frontal del Programa, en donde se pueden ver las variables Set_Point, Kp, Ti y Td, y los indicadores PV_Presion, Señal de Control. En la Figura4.31 se muestra el Diagrama de Bloques del programa.

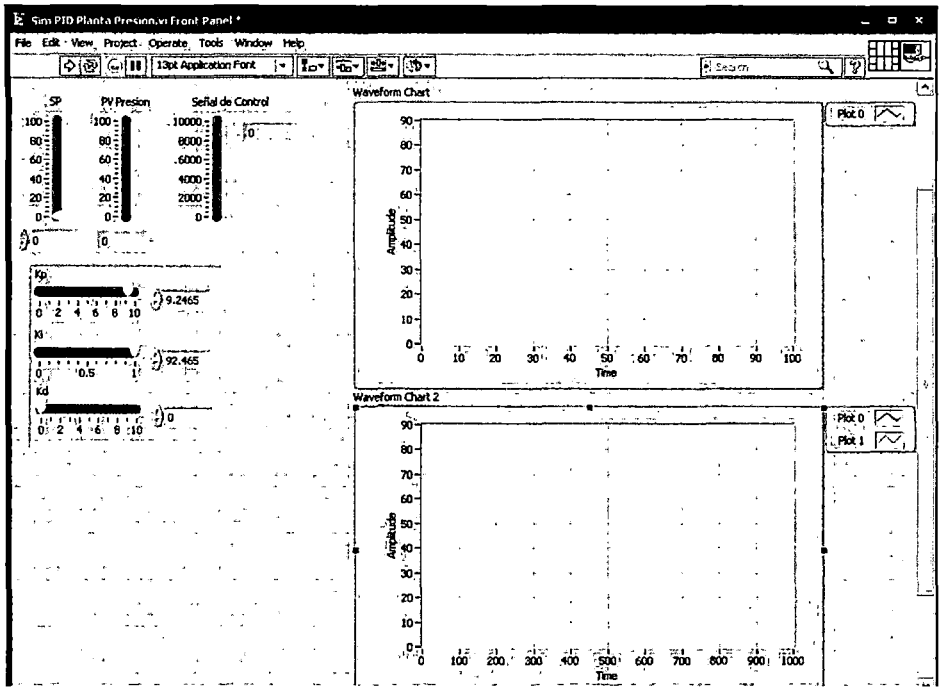


Figura 4.30: Panel Frontal de la programación en LabView de la simulación del Sistema.

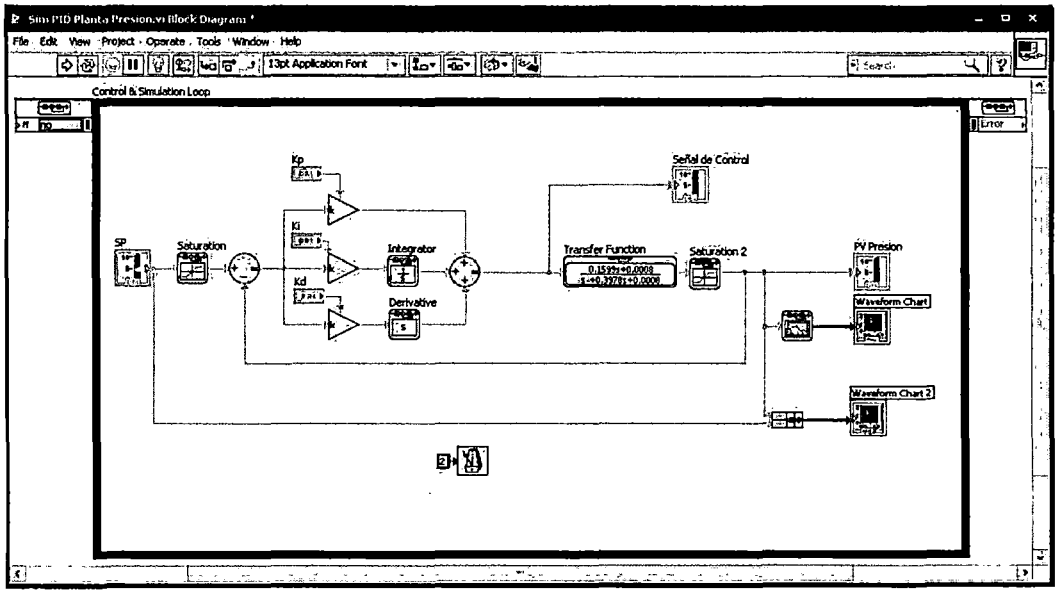


Figura 4.31: Diagrama de bloques de la programación en LabView de la simulación del Sistema.

4.6. Diseño del Sistema SCADA

Para prescindir de un sistema SCADA licenciado, en este proyecto de investigación, hacemos uso de un software libre, llamado *Soft Control*, este software fue proyecto de investigación realizado hace tres años, desarrollado principalmente en el lenguaje de programación Java, debido a que este leguaje es el líder de los lenguajes de programación para software libre, dispone de librerías de código abierto útiles y confiables, además de tener un costo muy bajo, es muy rápido y sus programas pueden ejecutarse en diferentes sistemas operativos.

Para este proyecto de investigación y con la asesoría de un ingeniero en programación, pudimos mejorar y adaptar *Soft Control* a la planta de presión del laboratorio de ingeniería electrónica. A este sistema SCADA mejorado, le llamamos *CONTROL SCADA APLICATION*, que fue renovado con la finalidad de añadir más funciones que sean de utilidad para el alumnado de la escuela profesional de ingeniería electrónica.

4.6.1. Java

Java es un lenguaje de programación y la primera plataforma informática creada por Sun Microsystems en 1995. Es la tecnología subyacente que permite el uso de programas punteros, como herramientas, juegos y aplicaciones de negocios. Java se ejecuta en más de 850 millones de ordenadores personales de todo el mundo y en miles de millones de dispositivos, como dispositivos móviles y aparatos de televisión.

Con la programación en Java, se pueden realizar distintos aplicativos, como son applets, que son aplicaciones especiales, que se ejecutan dentro de un navegador al ser cargada una página HTML en un servidor WEB. Por lo general los applets son programas pequeños y de propósitos específicos.

Otra de las utilidades de la programación en Java es el desarrollo de aplicaciones, que son programas que se ejecutan en forma independiente, es decir con la programación Java, se pueden realizar aplicaciones como un procesador de palabras, una hoja que sirva

para cálculos, una aplicación gráfica, etc. En resumen cualquier tipo de aplicación se puede realizar con ella.

La programación en Java, permite el desarrollo de aplicaciones bajo el esquema de Cliente Servidor, lo que lo hace capaz de conectar dos o más computadoras u ordenadores, ejecutando tareas simultáneamente, y de esta forma logra distribuir el trabajo a realizar.

4.6.2. MySQL

MySQL es un sistema de administración de bases de datos (Database Management System, DBMS) para bases de datos relacionales. Es la base de datos de código libre más popular y, posiblemente la mejor del mundo. Su continuo desarrollo y su creciente popularidad están haciendo de MySQL un competidor cada vez más directo de gigantes en la materia de las bases de datos como Oracle.

4.6.3. Alcance del Software

El software se encarga de monitorear los procesos que se ejecuten en la interfaz de manera remota es decir desde una ubicación dentro de la red de la planta.

También permitirá controlarlos y modificar sus parámetros, es capaz de almacenar *datos de todos los procesos y además presenta reportes históricos de los datos almacenados*, el sistema presenta comunicación Ethernet, cabe resaltar que el sistema es multiplataforma debido al uso del lenguaje de programación java, por lo tanto el sistema operativo en el cual funcionaría el software no sería una limitación.

4.6.4. Capacidades Generales

El sistema será capaz de:

a) Supervisar los procesos mediante indicadores, gráficos (animaciones) en tiempo real.

b) Controlar adecuadamente los mismos y además variar los parámetros de los dispositivos a los cuales estará conectado, en este caso parámetros del PLC.

c) Conectar dispositivos a través de comunicaciones Ethernet.

d) Presentar reportes históricos de los datos previamente almacenados.

e) Almacenará datos en una base de datos de MySQL, permitiendo al usuario especificar el tiempo de guardado.

4.6.5. *Restricciones Generales*

a) El sistema trabaja independientemente de otros sistemas, es decir no está prevista la integración con otro tipo de software.

b) Existe un protocolo diseñado para la comunicación entre el sistema y los dispositivos a conectar.

c) Se podrá manejar hasta 6 variables por proceso independiente del tipo (Entrada o Salida)

4.6.6. Entorno Operacional

El sistema funciona en cualquier sistema operativo, gracias al lenguaje Java, además es una aplicación Stand -Alone, es decir de escritorio, es necesario que el paquete de software Microsoft Office sea instalado en el equipo en que funcionará el sistema, el cual usa el aplicativo Microsoft Excel como medio para exportar reportes.

4.6.7. Modelo Conceptual

El sistema maneja el concepto de proceso como un grupo de variables comunicadas con dispositivos externos, es por eso que el sistema necesita almacenar aquellos datos que son imprescindibles para ellos como el nombre, el grupo de donde se obtienen

los datos del OPC, el driver de comunicación y la dirección IP, por otro lado el proceso posee 6 variables como máximo, cada una de ellas posee su nombre, el tipo de variable que indica si es tipo Input(Entrada) u Output(Salida), de la misma manera posee valores para las alarmas que indiquen un alto, bajo y un nivel apagado. Por último posee un indicador de almacenamiento y de gráfica en tiempo real. Cada variable de los procesos posee datos, los cuales se almacenan en la base de datos con fecha y su valor correspondiente. Este modelo se ilustra en la Figura 4.32.

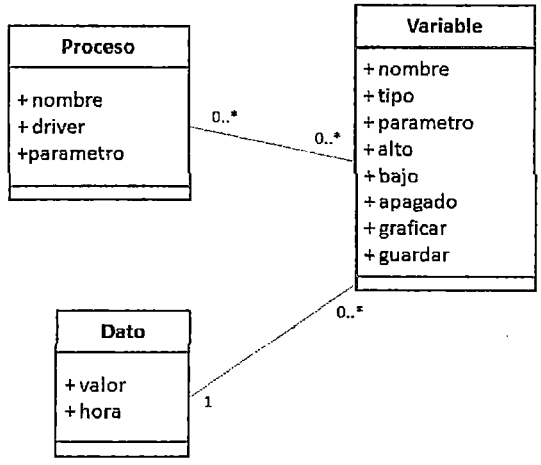


Figura 4.32: Modelo Conceptual.

4.6.8. Modelo relacional de la base de datos del sistema

El sistema necesita almacenar principalmente los datos de las variables de los procesos, pero para que dichos datos se almacenen de forma ordenada se necesita guardar los procesos y variables correspondientes, es por eso que se creó una tabla Proceso con los parámetros Nombre, Grupo (Nombre del Grupo creado en el servidor OPC), Driver que es la nombre con que se reconoce el software qure brinda comunicación OPC y el parámetro1 que es la dirección IP del PLC.

Como el proceso puede tener muchas variables y la misma variable puede repetirse en otro proceso, se creó una tabla intermedia ProVar para relacionar tanto la tabla Proceso como la tabla Variable. La tabla variable almacena los datos de las variables, los cuales son el nombre, tipo, función, alarmas, almacenamiento y graficado.

Por último la tabla de datos principal del sistema, posee los campos de valor y fecha del dato almacenado, esa tabla es la que nos permite hacer los reportes históricos. Este modelo se ilustra en la Figura 4.33.

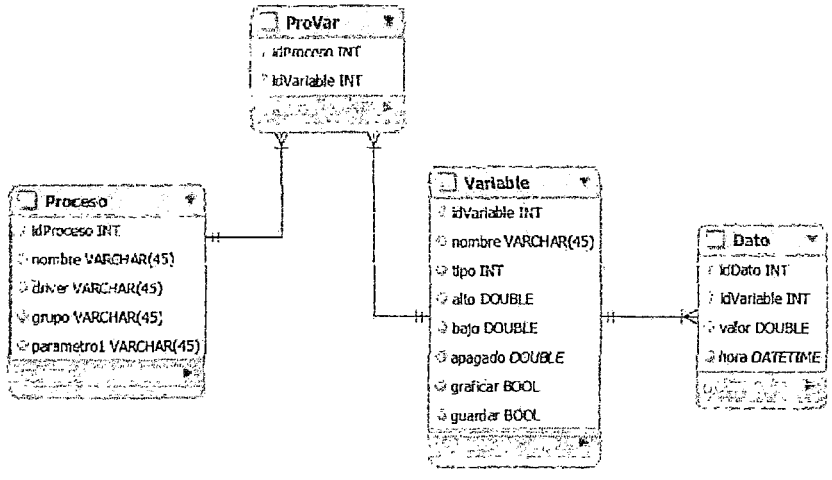


Figura 4.33: Modelo relacional de datos del sistema.

4.6.9. Arquitectura Lógica

El sistema tendría los componentes de base de datos (BD), lógica de negocio (LN), interfaz gráfica de usuario interna (GUI), además de la interfaz de comunicación para los distintos protocolos que manejaría el sistema. Dichos componentes estarían relacionados entre sí. Esta arquitectura se muestra en la Figura 4.34.

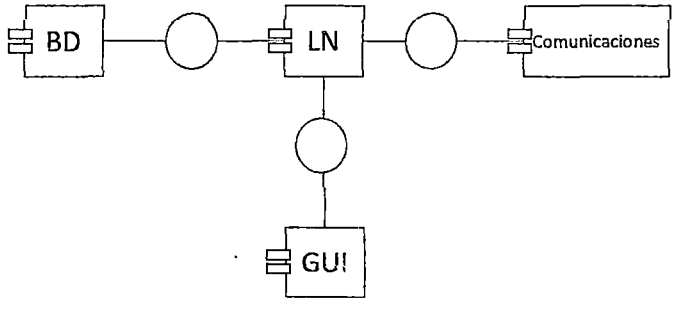


Figura 4.34: Arquitectura Lógica del Sistema.

4.6.10. Diseño de la Lógica de Negocios

La lógica del sistema se representa en la Figura 4.35 a modo de diagrama de clases, además se presentan los atributos de estas y sus métodos.

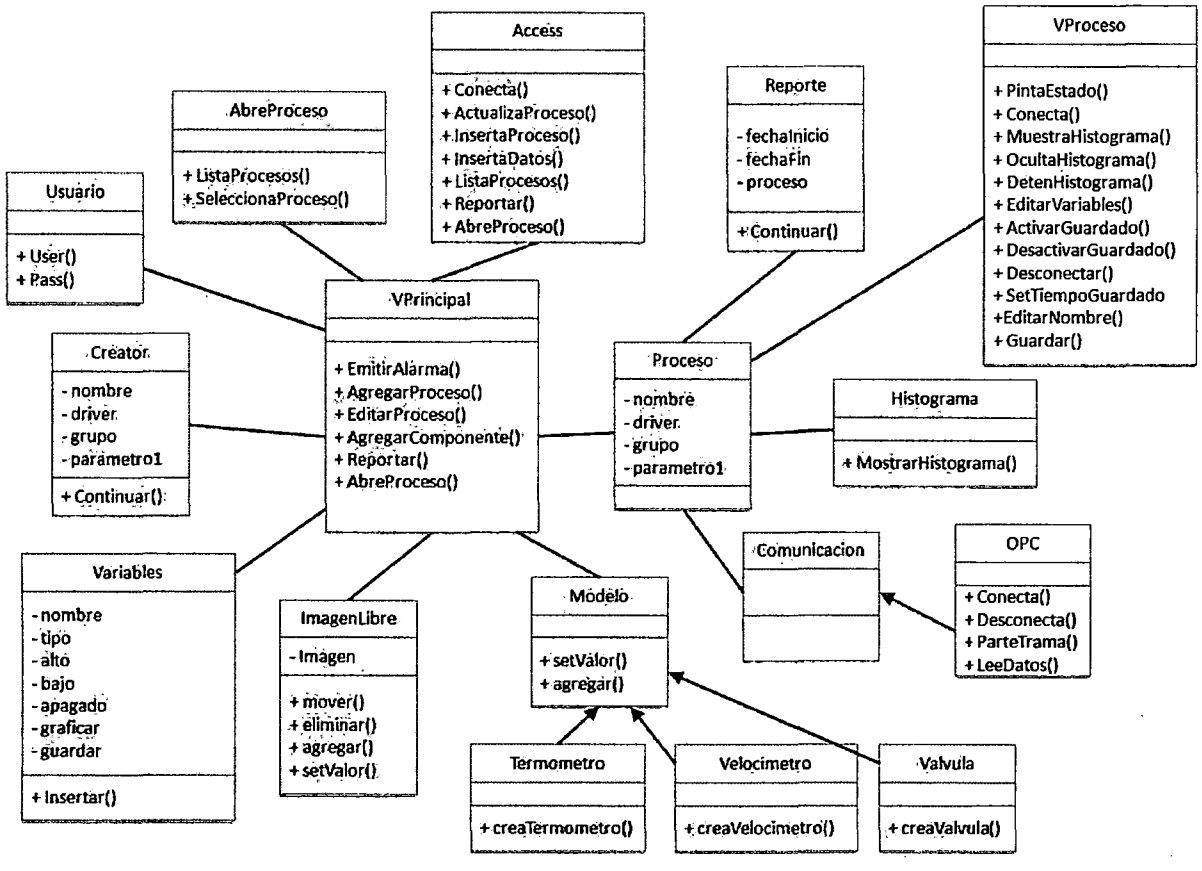


Figura 4.35: Diagrama de Clases del Sistema.

A continuación se detalla cada clase utilizada en la implementación, que métodos tendrán y como interactuarán entre ellas y su respectiva descripción técnica.

Clase VPrincipal:

- EmitirAlarma()

  El método EmitirAlarma se encarga de emitir avisos de alarma de estados, conexiones y otros tipos de avisos en el panel principal, este método es invocado por cada sub-

proceso en el que se ejecutan los procesos dentro del sistema, es decir cada vez que una variable cambie de estado o se pierda la conexión, etc. El hilo correspondiente al proceso llamará dicho método.

- AgregarProceso()
  El método AgregaProceso se encarga de inicializar la ventana de la clase Creator, con el cual se podrá crear un nuevo proceso al sistema, dicho método es llamado por un evento de clic en el menú de la ventana principal.

- EditarProceso()
  El método EditaProceso se encarga de hacer visible las ventana de los procesos que se encuentran siendo monitoreados desde la ventana Principal, el método EditaProceso es llamado por un evento de clic en los componentes de los respectivos procesos en el panel de VPrincipal.

- AbreProceso()
  El método AbreProceso se encarga de Inicializar la ventana AbreProceso para la inicialización de un proceso previamente guardado en la base de datos, dicho método es llamado por un evento de clic en el menú de la clase VPrincipal.

- AgregarComponente()
  El método AgregarComponente es el método que inserta las imágenes libres, modelos, visualizadores y procesos al panel principal de la clase VPrincipal, dicho método es llamado por eventos de clic en los componentes del menú en la clase VPrincipal.

- Reportar()
  El método reportar se encarga de inicializar la ventana Reporte en el cual se realizará una consulta de datos históricos de la base de datos para así presentar una gráfica del proceso en un intervalo de tiempo elegido por el usuario, dicho método es llamado por un evento de clic en el menú de la ventana VPrincipal.

Clase Creator:

- Continuar()
  El método continuar se encarga de inicializar la ventana Variables para proceder con el proceso de creación de procesos, específicamente pasar a la parte en donde se

configuran las variables del proceso, el método comprueba primero los valores ingresados sean válidos antes de pasar al siguiente paso, dicho método es llamado por un evento de clic en el botón de la clase Creator.

Clase Variables:

- Insertar()
  El método Insertar se encarga de crear un nuevo objeto de la clase Proceso con los datos obtenidos de la clase Creator y Variables, así como inicializar la ventana VProceso de cada proceso insertado en el panel de la clase Principal, dicho método es llamado por un evento de clic en el botón de la clase Variables.

Clase AbreProcesos:

- ListaProcesos()
  *El método ListaProceso es el método que es llamado al inicializar la ventana Lista-*Proceso, dicho método obtiene todos los procesos de la base de datos y los coloca en la lista del panel de esa clase, dicho método es llamado por el método constructor de ListaProceso al momento de inicializar la ventana.
- SeleccionaProceso()
  El método selecciona proceso se encarga de obtener los datos del proceso seleccionado en la lista de dicha clase y abrirlo con todas sus características previamente guardadas, dicho método es llamado por un evento de clic en el botón de la clase AbreProceso.

Clase Access:

- Conecta()
  El método conecta se encarga de establecer la comunicación entre el sistema y su base de datos en el programa Microsoft Access, dicho método es llamado al iniciar el programa, específicamente es llamado en el método constructor de VPrincipal.

- ActualizaProceso()

  El método ActualizaProceso se encarga de realizar una sentencia de UPDATE en la base de datos actualizando todos los datos del proceso seleccionado, dicho método es llamado por un evento de clic en el menú de la ventana VProceso, específicamente en el menú guardar.

- InsertaProceso()

  El método InsertaPrceso se encarga de realizar la sentencia INSERT en la base de datos, es decir se encarga de ingresarlos a la base de datos, obviamente antes de llegar a dicho método la información del proceso es verificada para que su registro sea válido, dicho método es llamado al igual que el método ActualizaProceso por un evento de clic en el menú guardar de la clase VProceso.

- InsertaDatos()

  El método InsertaDato se encarga de guardar los datos relacionados a las variables de los procesos en la base de datos, dicho método es llamado por un contador de tiempo de cada proceso al cual el usuario definió previamente el intervalo de guardado.

- ListaProcesos

  El método ListaProceso se encarga de obtener la lista de procesos guardados en la base de datos, dicho método es llamado por el método ListaProceso de la clase AbreProceso.

- Reportar()

  El método Reportar realiza una consulta a la base de datos pidiendo los registros de los datos de un proceso determinado en un periodo de tiempo determinado, dicho método es llamado por el método Reportar de la clase Reporte.

- AbreProceso()

  El método abre proceso hace una búsqueda en la base de datos de un proceso determinado y extrae sus datos para luego ser utilizados en crear un objeto de la clase Proceso y ser agregado en el panel de VPrincipal, dicho método es llamado por el evento de clic del botón de la clase AbreProceso.

Clase Reporte:

- Mostrar()

  El método Mostrar se encarga de crear el gráfico donde se mostrarán los datos históricos del reporte y además de llamar al método Reportar de la clase BaseDatos, dicho método es llamado en el constructor de la clase Reporte, es decir al inicializarse la ventana.

Clase VProceso:

- Conecta()

  El método Conecta se encarga de llamar a la clase de comunicación para lograr conectar el proceso con los dispositivos de trabajo, el método Conecta es llamado por un evento de clic en el botón conectar de la ventana VProceso.

- *MuestraHistograma()*

  El método MuestraHistograma se encarga de llamar al constructor de la clase Histograma para poder así crear el histograma en tiempo real y poder monitorear las variables del proceso, dicho método es llamado por un evento de clic en el menú de histograma de la ventana VProceso.

- OcultaHistograma()

  El método OcultaHistograma se encarga de esconder de la pantalla el histograma del proceso hasta que el usuario decida mostrarlo nuevamente, dicho método es llamado por un evento de clic en el menú de histograma de la ventana VProceso.

- DetenHistograma()

  El método DetenHistograma se encarga de detener el contador de tiempo que agrega nuevos datos al histograma, por lo tanto el histograma se quedaría estático hasta que el usuario decida continuarlo, dicho método es llamado por un evento de clic en el menú de histograma de la ventana VProceso.

- EditarVariables()

  El método EditaVariables se encarga de abrir la ventana Variables de cada proceso para poder editar los parámetros y los identificadores de comunicación, dicho método es llamado por un evento de clic en el menú de la clase VProceso.

- ActivarGuardado()

  El método ActivarGuardado se encarga de empezar el contador de tiempo para el guardado de los datos hasta que el usuario decida detenerlo, dicho método es llamado por un evento de clic en el menú de guardado de datos en la clase VProceso.

- DesactivarGuardado()

  El método DesactivarGuardado detiene el contador de tiempo que se encarga de guardar los datos de las variables hasta que el usuario desee continuar almacenando, dicho método es llamado por un evento de clic en el menú desactivar en la clase VProceso.

- Desconectar()

  Se encarga de detener el subproceso de comunicación entre el proceso y los dispositivos hasta que el usuario desee continuar la comunicación, dicho método es llamado por un evento de clic en el menú desconectar en la clase VProceso.

- SetTiempoGuardado()

  El método SetTiempoGuardado se encarga de pedir al usuario que ingrese un periodo de tiempo, el cual se usará como intervalo de guardado de los datos tomados de la comunicación y serán guardados en la base de datos, el método es llamado por un evento de clic en el menú de tiempo de guardado en la clase VProceso.

- EditarNombre()

  El método EditarNombre permite al usuario ingresar un nuevo nombre al proceso, para que se guarde el nuevo nombre hay que hacer clic en guardar después de editar el nombre, el método EditarNombre es llamado por un evento de clic en el menú de editar nombre de la clase VProceso.

- Guardar()

  El método Guardar se encarga de guardar la información del proceso en la base de datos, incluyendo el nombre, las variables, identificadores y todas las configuraciones realizadas al mismo, dicho método es llamado por un evento de clic en el menú guardar de la clase VPrincipal.

Clase Histograma:

- MostrarHistograma()

  El método MostrarHistrograma se encarga de preparar el gráfico ya sea el histograma para los datos en tiempo real o el histograma para los reportes, según sea el tipo de orden del objeto de dicha clase, el histograma como reporte se pone en el panel de dicha clase para mostrarlo al usuario. El método MostrarHistograma es llamado por un evento de clic en el menú de mostrar histograma en VProceso o por un evento de clic en el botón de aceptar de la clase Reporte.

Clase ImagenLibre:

- Mover()

  Se encarga de colocar el objeto de la clase ImagenLibre en la posición que se encuentra el mouse dentro del panel de VPrincipal, dicho método es llamado por un evento de arrastre de ratón dentro del panel de VPrincipal.

- Eliminar()

  El método Eliminar se encarga de eliminar la imagen libre del panel VPrincipal en el caso que así se desee, dicho método es llamado por un evento de clic en el menú eliminar del submenú del objeto ImagenLibre.

- Agregar()

  El método Agregar se encarga de colocar la imagen libre en el panel de VPrincipal, dicho método es llamado al crear el objeto ImagenLibre.

- SetValor()

  El método SetValor se encarga de darle el valor de la variable que representa la ImagenLibre en caso de ser del tipo de visualizador para poder así monitorear una variable desde el panel principal de la clase VPrincipal, dicho método es llamado cada vez que se recibe un dato en el subproceso de comunicación de la respectiva variable.

Clase Modelo:

- SetValor()
  El método SetValor se encarga de visualizar el valor asociado a la variable del modelo correspondiente, dicho método es llamado cada vez que se recibe un nuevo dato de la variable representada.

- *Agregar()*
  El método Agregar se encarga de colocar el objeto Modelo al panel principal de la clase VPrincipal para así tener una mejor supervisión del proceso sin tener que abrirlo y ver el histograma, el método es llamado al inicializarse el objeto Modelo.

Clase Termometro:

- creaTermómetro()
  Se encarga de crear el modelo termómetro con sus parámetros de valor mínimo y máximo, dicho método es llamado al inicializarse el objeto.

Clase Velocimetro:

- creaVelocimetro()
  Se encarga de crear el modelo velocímetro con sus parámetros de valor mínimo y máximo, dicho método es llamado al inicializarse el objeto.

Clase Valvula:

- creaValvula()
  Se encarga de crear el modelo valvula con sus parámetros de valor mínimo y máximo, dicho método es llamado al inicializarse el objeto.

Clase OPC:

- Conecta()
  Se encarga de establecer la comunicación mediante el protocolo OPC entre un proceso y un dispositivo conectado al servidor, dicho método es llamado por el método conectar de la clase VProceso.

- Desconecta()
  Se encarga de detener el proceso de comunicación entre el dispositivo conectado y el proceso, el método es llamado por el método desconectar de la clase VProceso o por un evento de error en el subproceso de comunicación.

- LeeDatos()
  Se encarga de inicializar el subproceso de comunicación por el protocolo OPC, recibiendo datos y analizándolos.

4.6.11. Librerías Java Usadas

Para la implementación del sistema se usaron diversas librerías externas al API nativo de JAVA, las cuales cumplen determinadas funciones dentro del sistema, dichas librerías son las siguientes:

JFree Chart

Esta librería se usó en la parte de control y monitoreo de los procesos en planta, al igual que se usó para presentar los reportes históricos.

Es una librería de código libre para programación en lenguaje JAVA, con ésta *librería es muy sencillo realizar gráficos, cuadros en las aplicaciones, dicha librería posee* una amplia documentación de sus métodos y de la construcción de todos los tipos de gráficos. La librería puede crear no solo gráficos en 2D sino también en 3D, gráficos de barra, líneas de tiempo, gráficos de pronósticos, etc.

La ventaja de su uso es que la librería es *"open source"* lo que quiere decir que es gratis, además dispone de una fácil integración con el IDE Netbeans lo cual facilita el trabajo

DateChooser

Esta librería es perfecta para seleccionar datos de tipo fecha en el sistema, se usó en la parte de selección del rango de fechas para el reporte histórico.

DateChooser es una librería para lenguaje java con componentes de la librería *swing, contiene 3 componentes para la implementación: Un panel, un editor de fechas* y una ventana de diálogo, los cuales son totalmente compatibles con cualquier IDE de programación.

Dicha librería permite principalmente:

- Seleccionar fechas, periodos, horas, etc.
- Permite configurar su apariencia dentro de la ventana
- Editar las celdas de información

4.6.12. Entorno Gráfico del Sistema

a) En la Figura 4.36 se muestra la pantalla inicial del SCADA, que permite el ingreso al *sistema, con el Usuario y la clave respectiva.*

b) Al ingresar nos mostrara la pantalla principal presentada en la Figura 4.37.

c) En la barra de herramientas encontramos el menú proceso, mostrada en la Figura 4.38, que nos permite Crear, abrir o guardar un esquema.

d) La opción MULTIMEDIA mostrada en la Figura 4.39 presenta alternativas como: Importar Imagen, Importar animación, Visualizadores, Componentes.

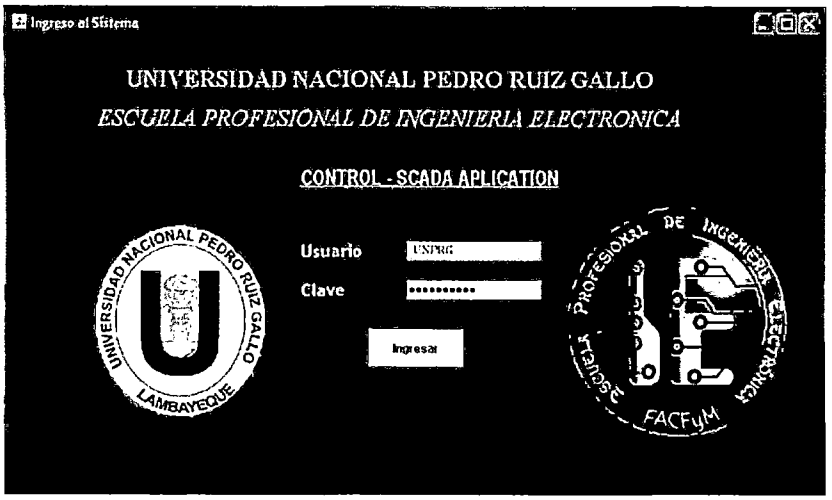


Figura 4.36: Pantalla de Bienvenida del Software SCADA.

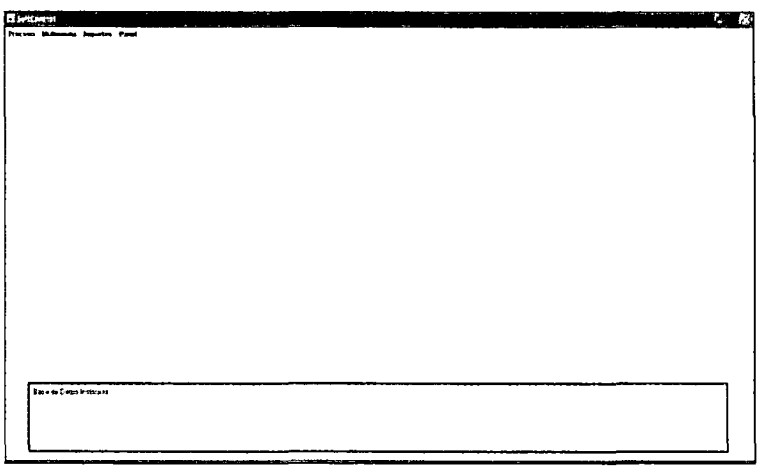


Figura 4.37: Pantalla de Principal del Software SCADA.

e) En la Figura 4.40 se presenta el sub Menú importar imagen, que nos permite cargar al esquema cualquier imagen en formato png que tengamos almacenado en el equipo

f) En la Figura 4.41 se presenta el sub Menú importar Animación, que nos permite elegir un visualizador en tiempo real en forma de Termómetro o Velocímetro.

g) En el sub Menú Visualizadores, presentado en la Figura 4.42 permite cargar imágenes previamente establecidas al esquema.

h) El sub Menú componentes, presentado en la Figura 4.43, no permite agregar 3 tipos de botones, con la funcionalidad de interactuar con el proceso.

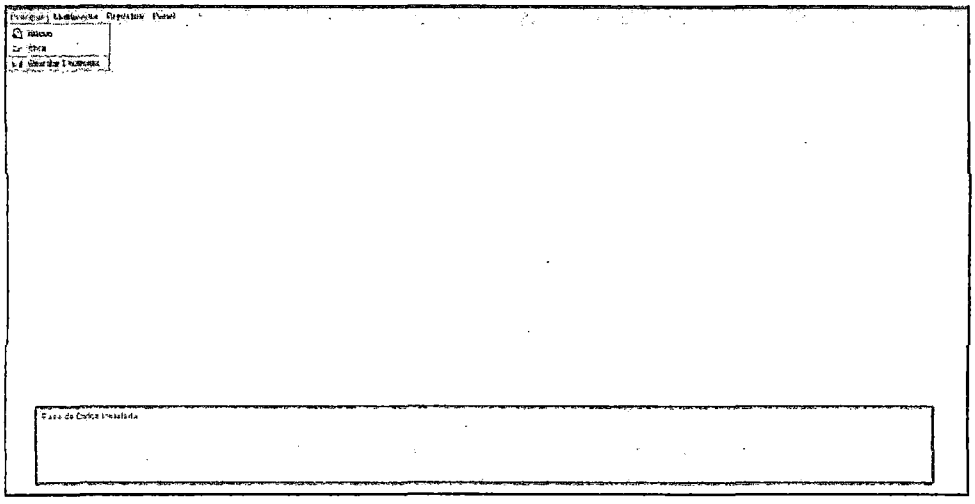


Figura 4.38: Opción Proceso del Software SCADA.

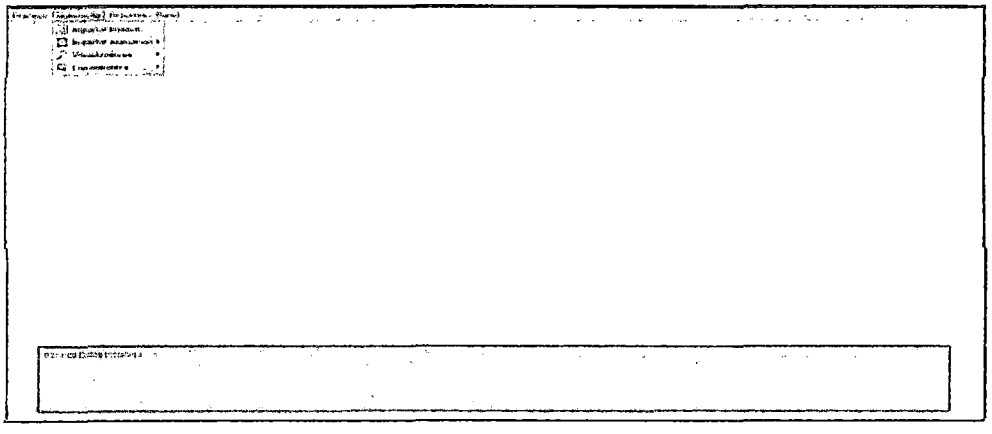


Figura 4.39: Opción Multimedia del Software SCADA.

i) También se encuentra el Menú Reporte, mostrado en la Figura 4.44 que permite visualizar la data de un periodo y el Menú Panel que nos permite cambiar el color de fondo del esquema.

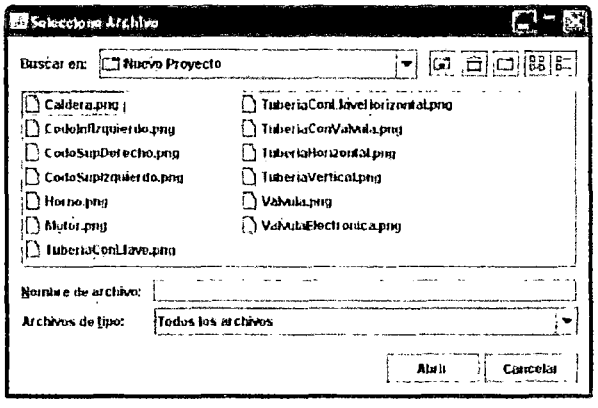


Figura 4.40: Sub Opción Importar Imagen del Software SCADA.

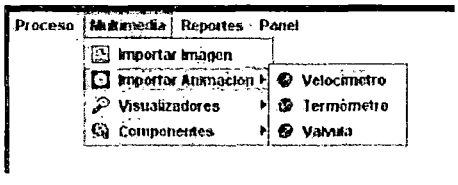


Figura 4.41: Sub Opción Importar Animación del Software SCADA.

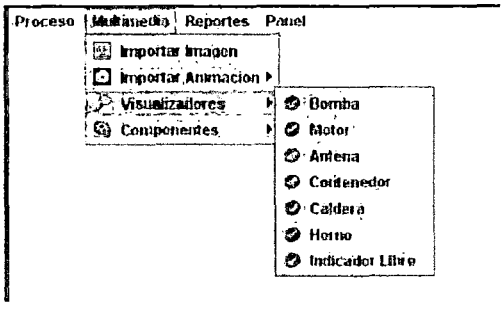


Figura 4.42: Sub Opción Importar Visualizadores del Software SCADA.

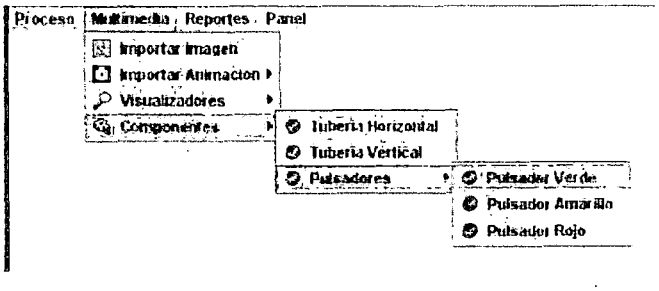


Figura 4.43: Sub Opción Importar Componentes del Software SCADA.

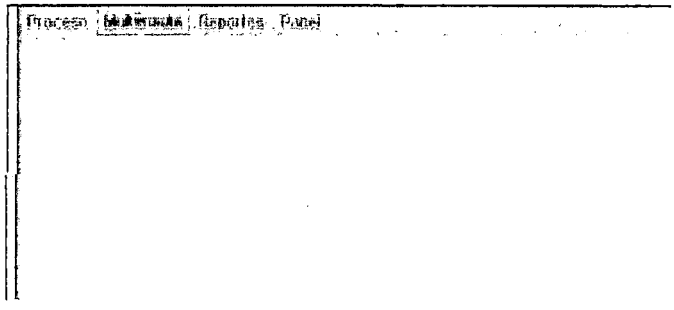


Figura 4.44: Opciones Reporte y Panel del Software SCADA.

CAPÍTULO 5
DISEÑO DE LA EXPERIMENTACIÓN

5.1. Implementación del Controlador en Planta Real

5.1.1. Programación en el Unity Pro

A la secuencia realizada anteriormente, mostrada en la Figura 4.29, se agregaron algunas variables que permitan el control de las constante K_p, T_i y T_d del controlador, así como las variables del poceso: Set_Point, Presion, Apertura, PV, SP, S_Control, esta secuencia se muestra en la Figura 5.1. También se utilizó la misma secuencia en Ladder que permite poner en marcha la planta mostrada en en el Capítulo 4 en la Figura 4.1.

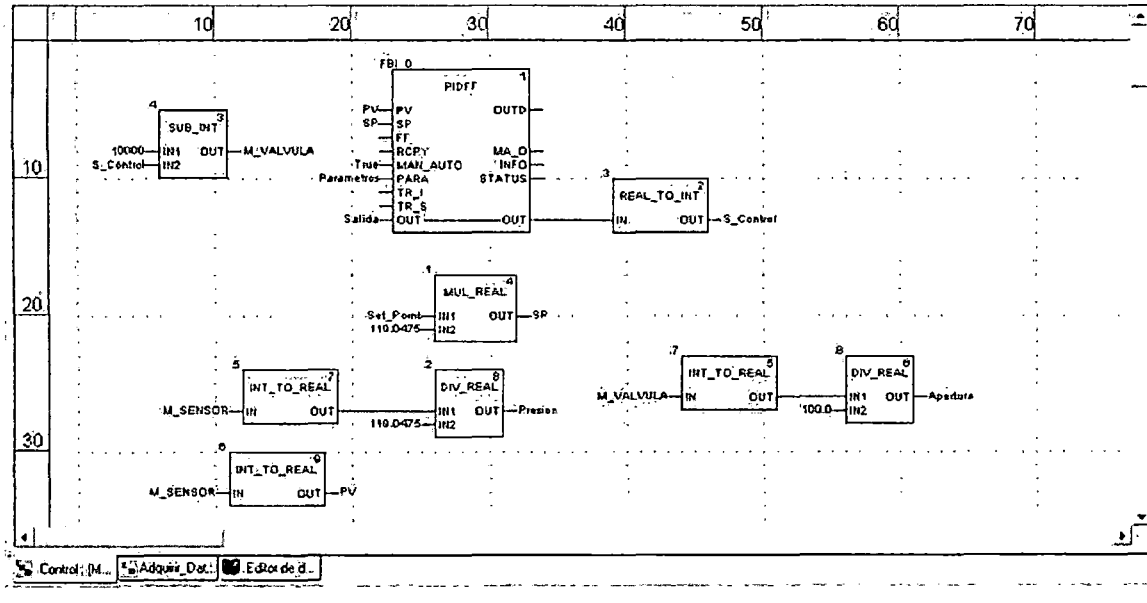


Figura 5.1: Secuencia completa del Controlador.

La Tabla 5.1 muestra las nuevas variables que se agregaron para el diseño general

del controlador.

Variable	Tipo	Dirección memoria
M_START	EBOOL	%M0
M_STOP	EBOOL	%M1
M_LIGHT	EBOOL	%M2
M_BIT_MOTOR	EBOOL	%M3
M_VARIADOR	INT	%MW0
M_SENSOR	INT	%MW1
M_VALVULA	INT	%MW4
PV	REAL	%MW8
SP	REAL	%MW10
S_Control	INT	%MW12
Presion	REAL	%MW14
Set_Point	REAL	%MW16
Apertura	REAL	%MW18
K_p	REAL	%MW32
T_i	TIME	%MW34
T_d	TIME	%MW36

Tabla 5.1: Variables empleadas para el diseño del Controlador.

5.1.2. Programación en Labview

La secuencia consiste en dar la orden de arranque y parada mediante los botones START y STOP respectivamente y un indicador visual de que el sistema está funcionando mediante la lámpara LIGHT, así como manipular las variables K_p, T_i, T_d, Set Point, y muestra en histogramas las variable Set Point vs Presión, también contiene visualizadores de las variable Válvula y Señal de Control.

Estas variables han sido enlazadas a su correspondiente variable en el registro del PLC mediante la opción Data Binding de Labview, como se muestra en la Tabla 5.2. Quedando la prueba final con Labview, como se muestra en las Figuras 5.2 y 5.3.

Variable	Tipo	Direccion
START	M_START	$Modbus.Rtu,0:1$
STOP	M_STOP	$Modbus.Rtu,0:2$
LIGHT	M_LIGHT	$Modbus.Rtu,0:3$
VARIADOR	M_VARIADOR	$Modbus.Rtu,4:1$
SENSOR	M_SENSOR	$Modbus.Rtu,4:2$
VALVULA	M_VALVULA	$Modbus.Rtu,4:5$
Señal de Control	Señal_de_Control	$Modbus.Rtu,4:13$
Presion	Presion	$Modbus.Rtu,4:15F$
Set Point	Set_Point	$Modbus.Rtu,4:17F$
K_p	K_p	$Modbus.Rtu,4:33F$
T_i	K_p	$Modbus.Rtu,4:35$
T_d	K_p	$Modbus.Rtu,4:37$

Tabla 5.2: Variables empleadas para el diseño del Controlador.

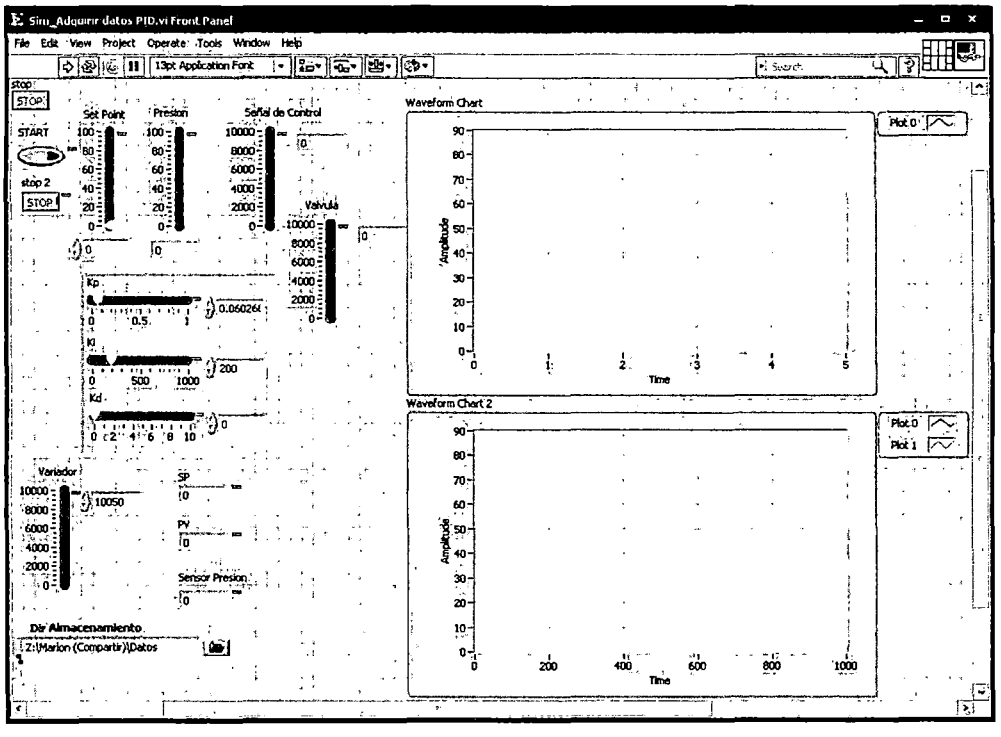


Figura 5.2: Panel frontal de la programación final en LabView.

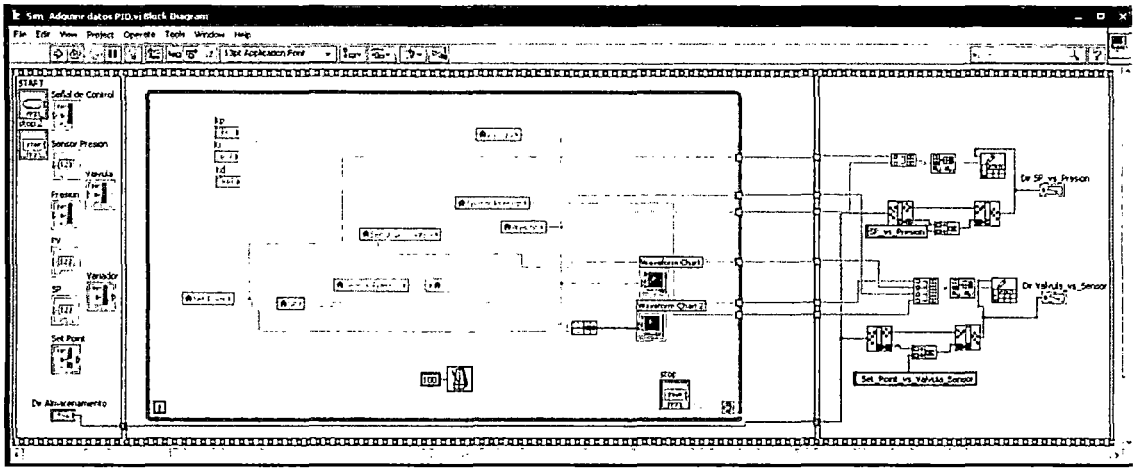


Figura 5.3: Diagrama de bloques de la programación Final en LabView.

5.2. Creación de las Variables en Matrikon OPC

Todas las variables se dividieron en 4 grupos, que son Controlador, Válvula, Variador y ON-OFF.

a) El primer grupo, llamado "Controlador" contiene las variables, Set point, K_p, T_d, T_i, $Variable\ de\ Proceso$. Este grupo contiene todas las variables usadas en el proceso de Control como se muestra en la Figura 5.4.

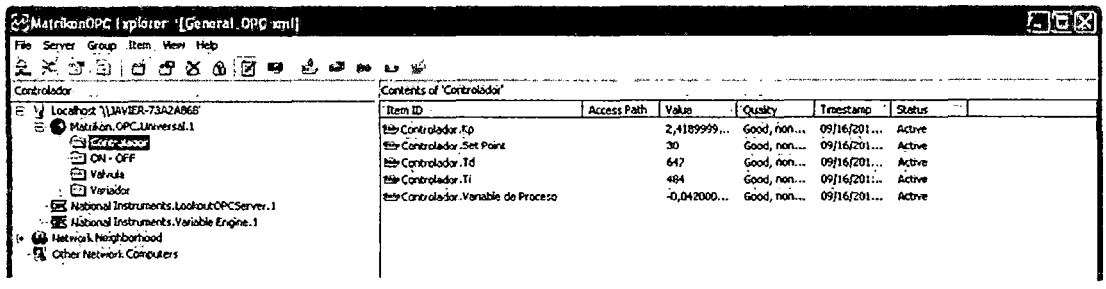


Figura 5.4: Grupo "Controlador" y variables en el OPC.

b) El segundo grupo "ON - OFF", contiene las variables qe son usadas en el encendido y apagado de la planta, estas variables son mostradas en la Figura 5.5.

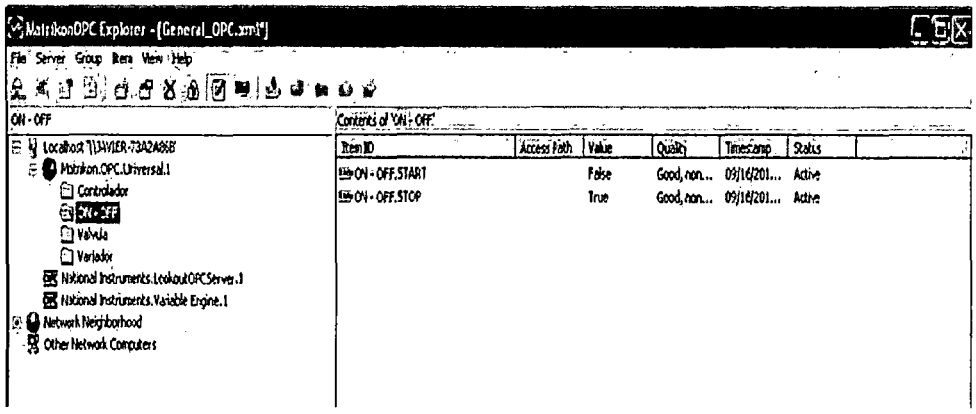


Figura 5.5: Grupo "ON - OFF" y variables en el OPC.

c) El grupo "Valvula" contiene las variables relacionadas con el accionar de la válvula, estas son: $Valvula$, $Senal\_Control$ y $Valvula\_Apertura$. Este grupo se muestra en la Figura 5.6.

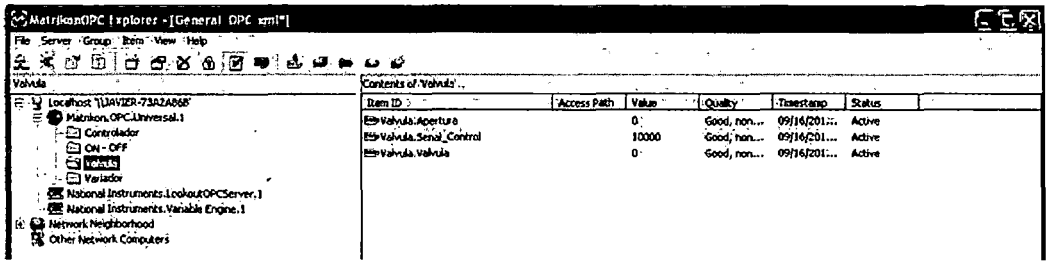


Figura 5.6: Grupo "Valvula" y variables en el OPC.

d) El grupo "Variador" sólo contiene la variable $Variador$, que envía un comando al variador para modificar su velocidad de giro. Este Grupo es mostrado en la Figura 5.7.

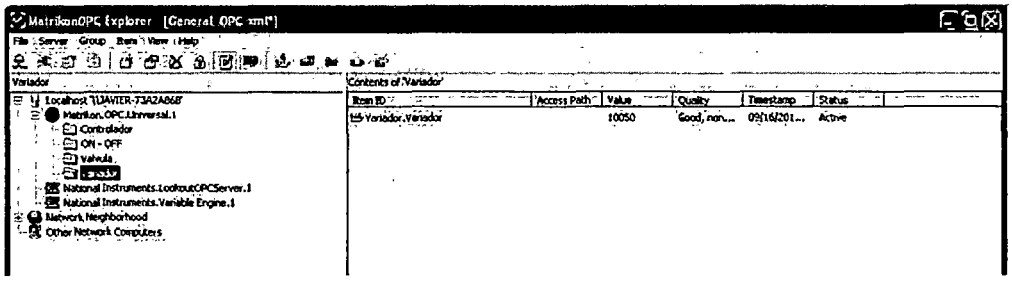


Figura 5.7: Grupo "Variador" y variable en el OPC.

5.3. Creación de las Variables en el SCADA

a) Comenzamos creando los procesos en el sistema SCADA, colocando los mismos nombres que se pusieron en el matrikon OPC.

b) El primer proceso que crearemos es "Controlador", vamos a la opcion *Proceso*, *Crear Proceso* y seguimos las instrucciones que sale en la ventana que se muestra en la Figura 5.8.

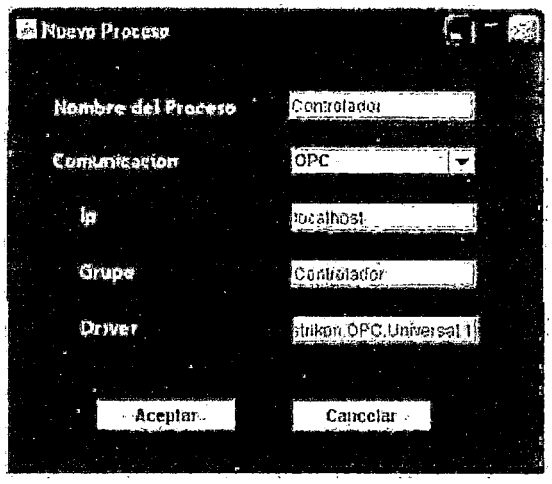


Figura 5.8: Creación del proceso "Controlador".

c) Luego se deben de crear las variables del proceso, estas deben ser creadas con los mismos nombres que han sido creadas en matrikon OPC, editamos rango, y seleccionamos que grafica se guarda con la letra "Y" y las que no vamos a usar con la letra "N", como se muestra en la Figura 5.9.

d) Después de haber creado las variables, estas se guardan en dicho proceso, en la parte superiror se encuentran las variables que han sido incluidas en el proceso y en la parte inferior se muestra un histograma en tiempo real que es posible edita el rango desde la opción *Histograma*. Damos click en la opción *Conectar*. El resultado se muestra en la Figura 5.10.

El proceso "Controlador" creado se muestra en la pantalla principal del SCADA, como *muestra la Figura 5.11*.

e) Analogamente se crearon los procesos "Válvula", y "Variador", los resultados se muestran en las Figuras 5.12, 5.13 y 5.14.

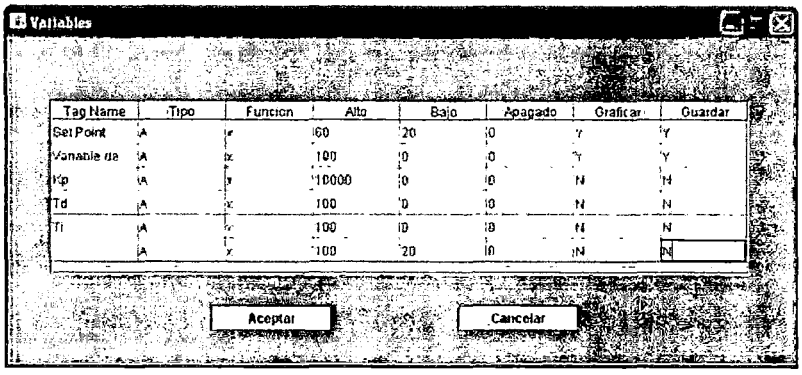


Figura 5.9: Creación de las variables del proceso "Controlador".

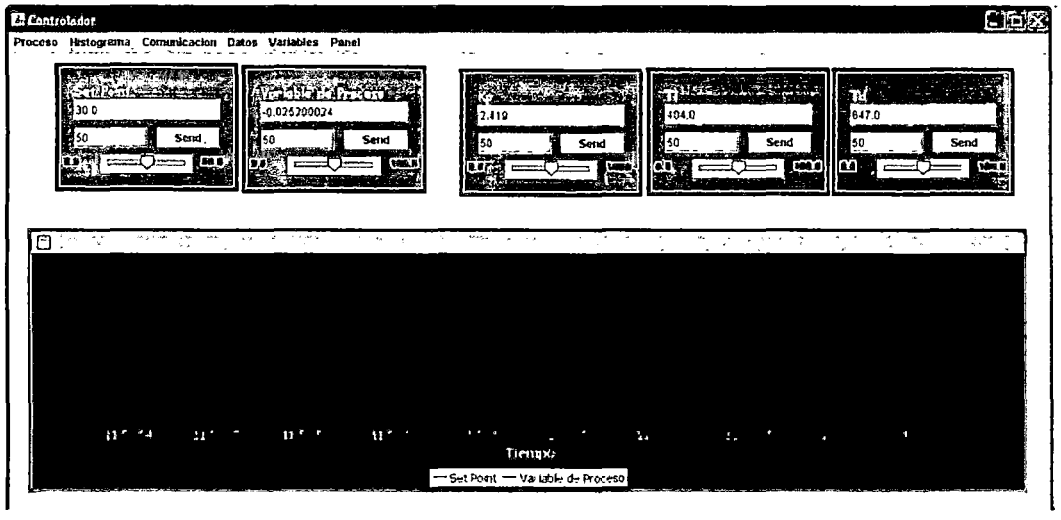


Figura 5.10: Ventana del Proceso "Controlador".

f) Para la puesta en marcha y parada de la planta, se usaron las variables del Grupo "ON - OFF", se escogieron 4 pulsadores, dos para START y dos para STOP, los pulsadores verdes se configuraron para enviar el comando $True$ mientras los rojos para enviar el comando $False$ en ambos casos. La manera de Configurar los Pulsadores se presentan en la Figura 5.15.

g) Para inserta un indicador visual, se importa una animación del Menú, en este caso se usó la animación velocímetro en donde se tiene que enlazar con una variable de un Proceso creado y seleccionar un rango. Se crearon 2 animaciones para las variables $Presion$ y $Apertura$, que indican la Presión del proceso y el porcentaje de apertura de la válvula en tiempo real. En la Figura 5.16 se muestra este procedimiento.

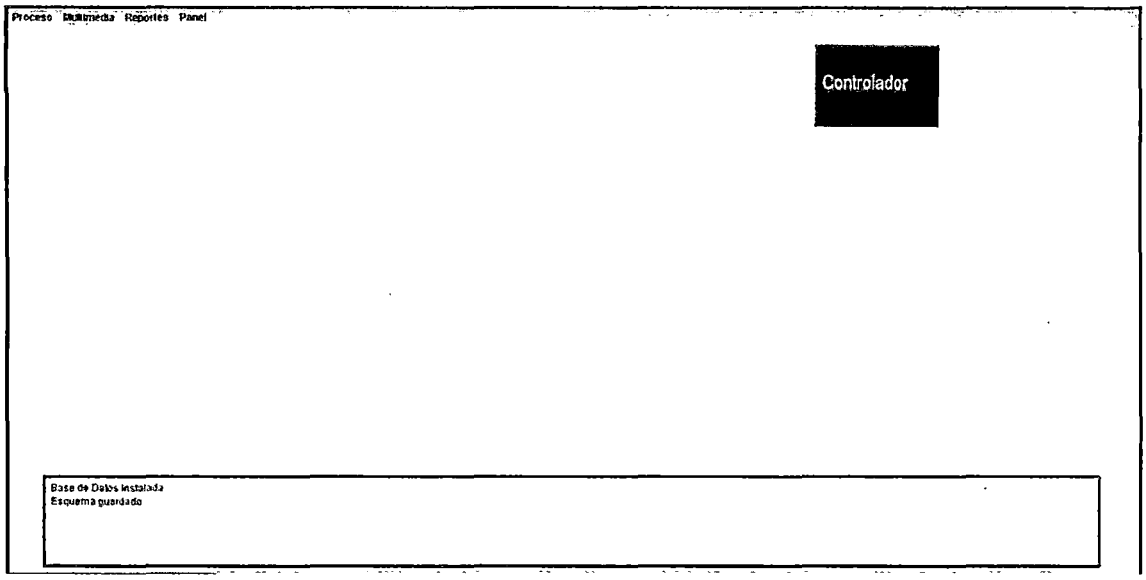


Figura 5.11: Ventana Principal del SCADA, con el Proceso "Controlador" creado.

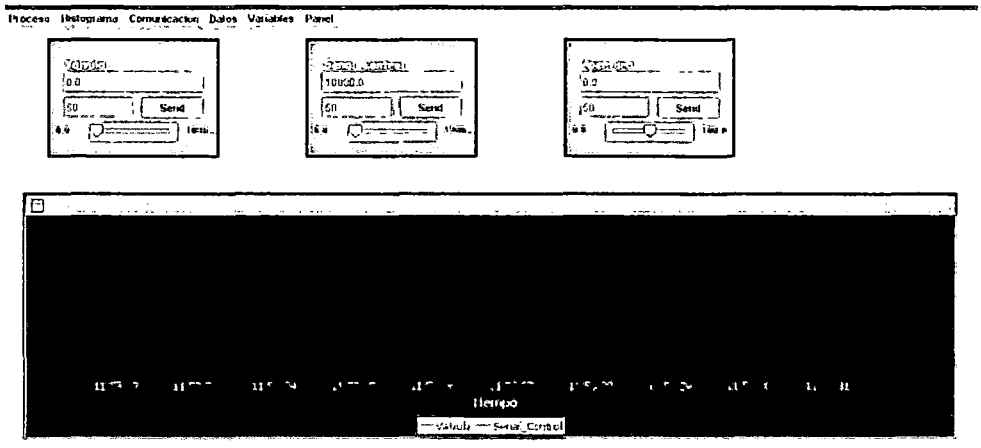


Figura 5.12: Ventana del Proceso "Valvula".

h) El Esquema General del proceso es mostrado en la Figura 5.17.

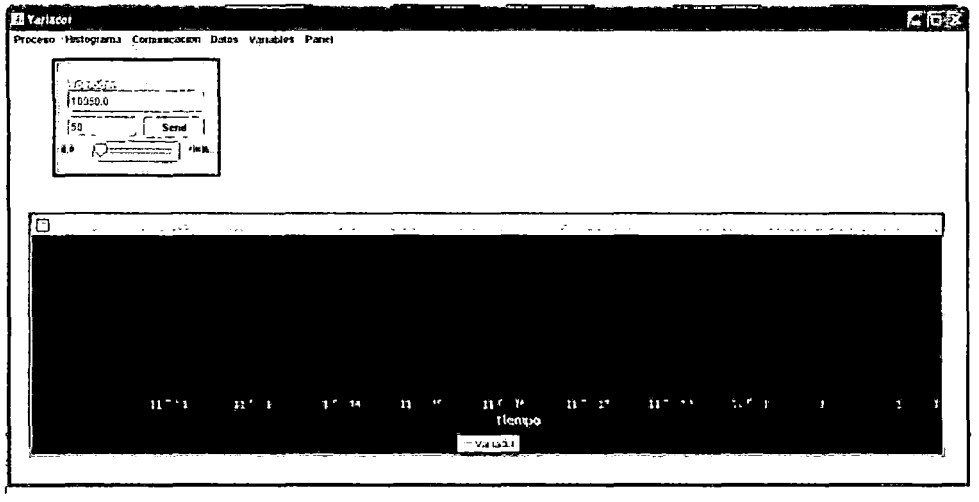


Figura 5.13: Ventana del Proceso "Variador".

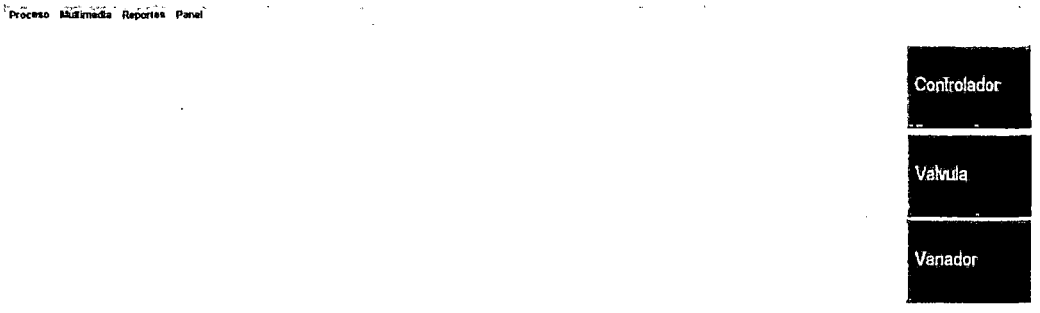


Figura 5.14: Pantalla principal del SCADA con los tres procesos creados.

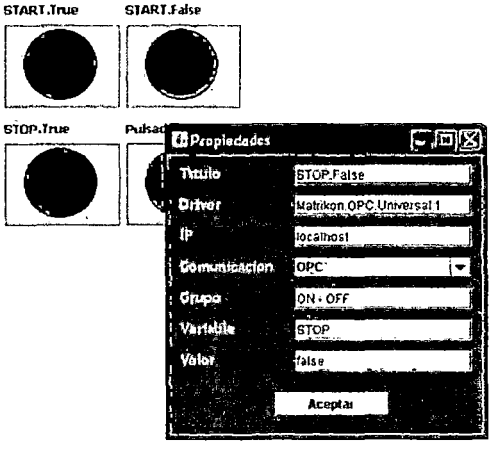


Figura 5.15: Configuración del comando $False$ para el botón STOP.

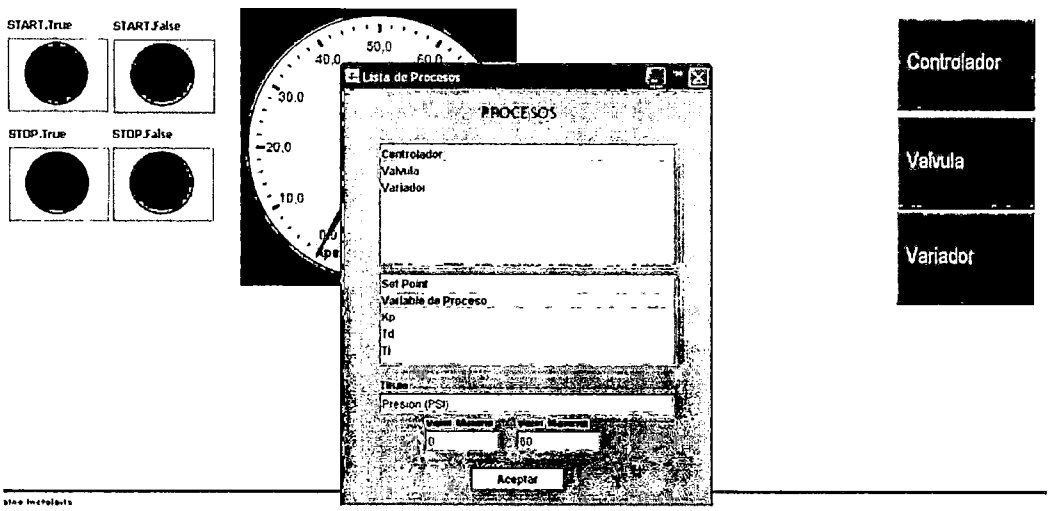


Figura 5.16: Configuración del indicador "Presión(PSI)".

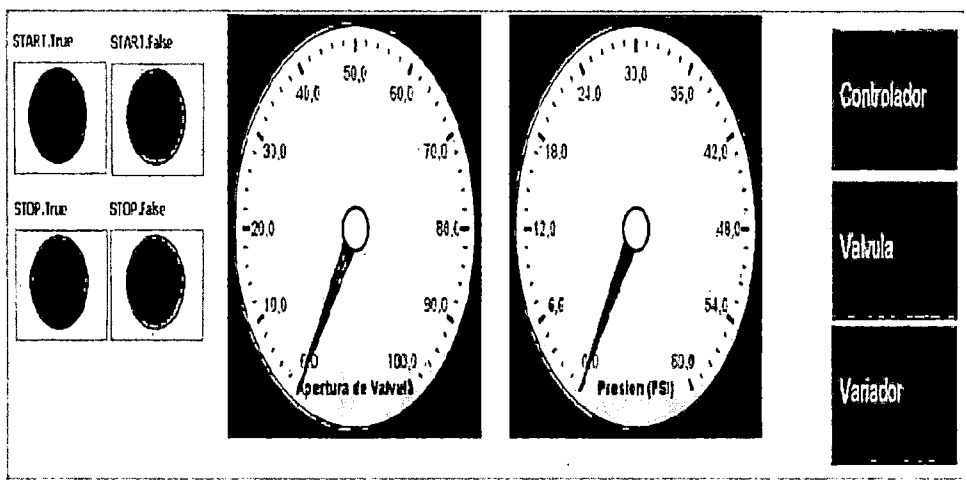


Figura 5.17: Esquema General del proceso realizado en el SCADA.

CAPÍTULO 6
RESULTADOS

6.1. Resultados Experimentales

Los sistemas de control P, PI, y PID diseñados fueron verificados mediante experimentación. Se llevaron a cabo 3 tipos de experimentos:

- Control de presión de la planta empleando un controlador Proporcional (P).
- Control de presión de la planta empleando un controlador Proporcional - Integral (PI).
- Control de presión de la planta empleando un controlador Proporcional - Integral - *Derivativo (PID).*

Las especificaciones de diseño a cumplir en todos los casos son:

- Error de estado estacionario nulo.
- Porcentaje de sobrepico menor al 20
- Tiempo de estabilizacion menor a 40 s.

6.1.1. Control Proporcional (P)

El parámetro ganancia K_p del controlador PID se estableció en 8, mientras que los parámetros T_i y T_d se fijaron en 0, para obtener un controlador puramente proporcional. Estos parámetros fueron determinados durante la etapa de simulación mediante el método del lugar geométrico de las raices.

Experimentación con Labview

En la Figura 6.1 se puede apreciar que no se cumple con las especificaciones de diseño, debido a que el error de estado estacionario no es nulo, es aproximadamente 6 para este caso.

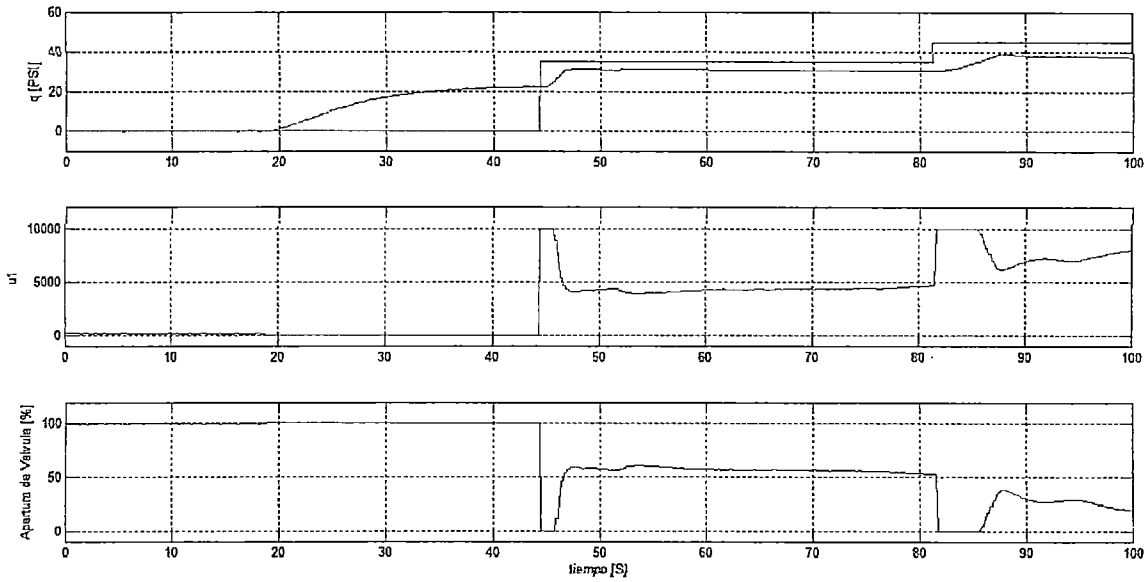


Figura 6.1: Respuesta del Sistema y Ley de Control ante un controlador Proporcional con K_p=8.

Debido a esto se desarrolló un segundo experimento del controlador proporcional, aumentando el parámetro Kp a 48, los resultados se muestran en Figura 6.2. Con este experimento el error de estado estacionario es aproximadamente de 1, con lo que queda demostrado que aumentado el parámetro K_p, el sistema no oscilará indefinidamente, si no que el error de estado estacionario se irá aproximando a 0.

Experimentación con Software SCADA

En el sistema SCADA se hizo una prueba real, en el que a K_p le dimos el valor de 32, con un $Set\ Point$ de 35. La respuesta del proceso "Controlador"se muestra en la Figura 6.3, y el Panel principal en la Figura 6.4.

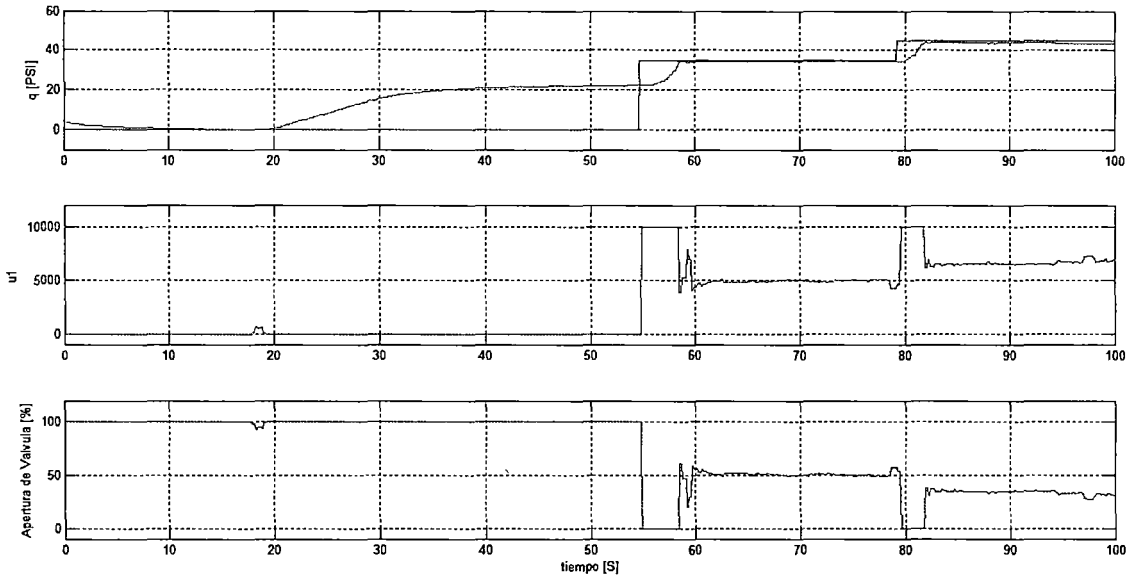


Figura 6.2: Respuesta del Sistema y Ley de Control ante un controlador Proporcional con K_p=48.

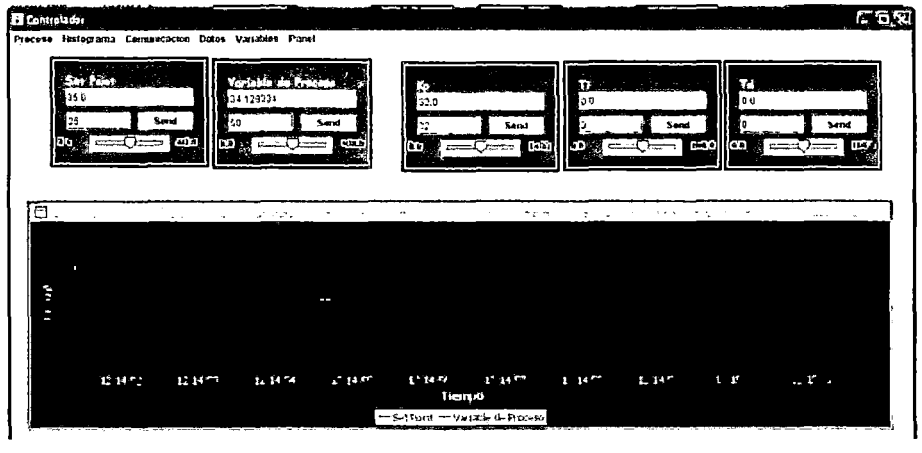


Figura 6.3: Respuesta del Sistema controlador Proporcional con K_p=32, línea azul: Variable de Proceso, línea roja: Set Point.

6.1.2. Controlador Proporcional - Integral (PI)

El parámetro ganancia K_p del controlador PID se estableció en 0.060268, mientras que los parámetros T_i y T_d se fijaron en 0.205 y 0 respectivamente, para ver un control proporcional integral. Estos parámetros fueron determinados durante la etapa de simulación mediante el método del lugar geométrico de las raices.

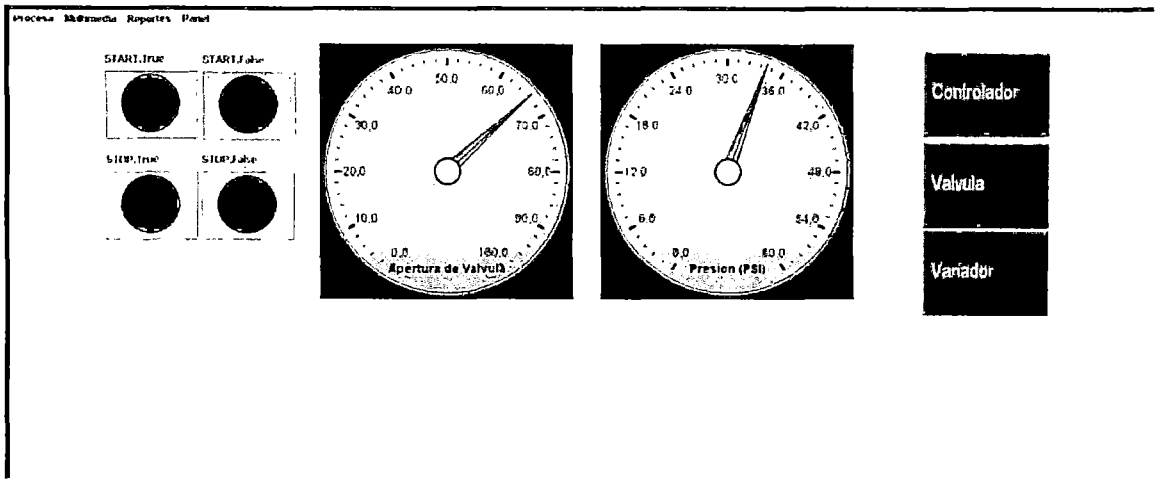


Figura 6.4: Panel Principal mostranto la respuesta del controlador P ante un Set Point de 35.

Experimentación con Labview

En la Figura 6.5 se puede apreciar que este diseño cumple con las especificaciones, con un error de estado estacionario nulo, porcentaje de sobrepico menor al 20 % y tiempo de estabilización menor a 25 segundos.

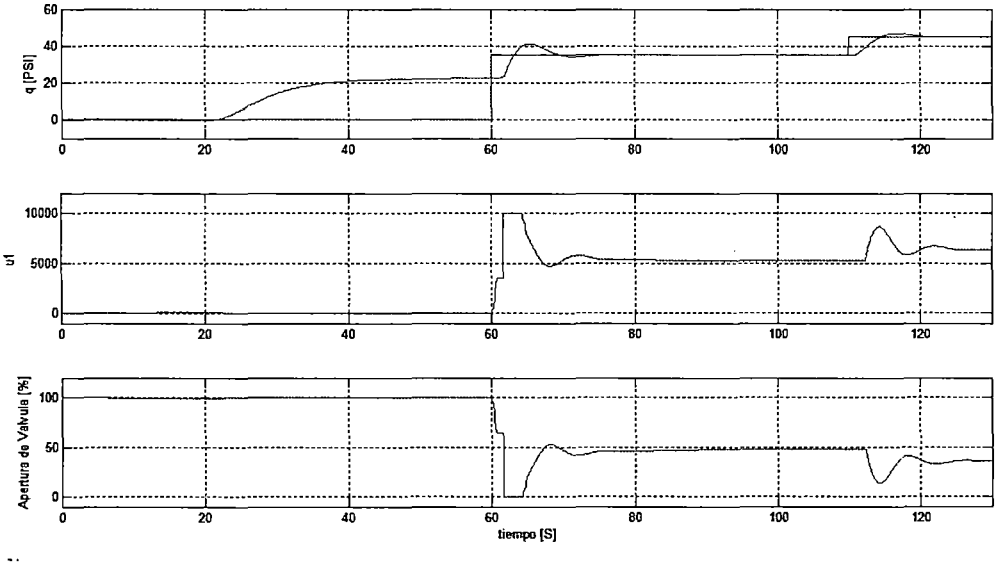


Figura 6.5: Respuesta del Sistema y Ley de Control ante un controlador PI.

Experimento con el Software SCADA

En el sistema SCADA se hizo una prueba real, en el que a K_p le dimos el valor de 0.060268, T_i = 205.0, T_d = 0, el resultado se muestra en las Figuras 6.6 y 6.7.

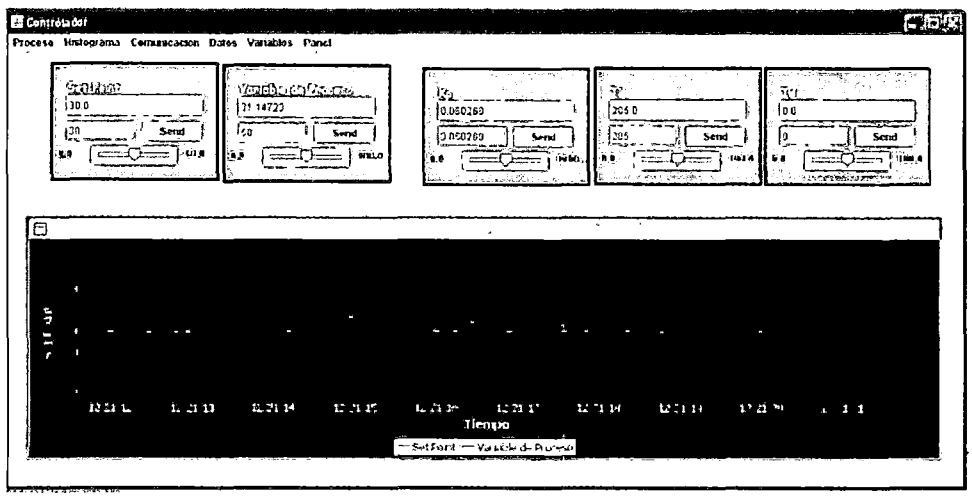


Figura 6.6: Respuesta del Sistema controlador PI, línea azul: Variable de Proceso, línea roja: Set Point.

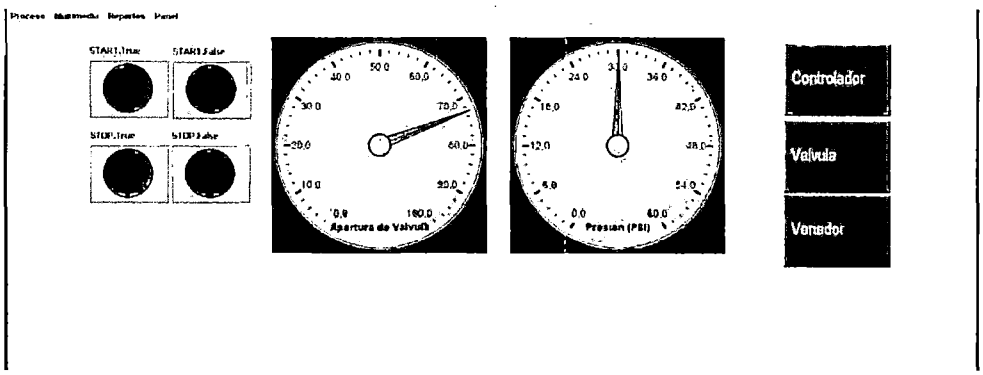


Figura 6.7: Panel Principal mostranto la respuesta del controlador PI ante un Set Point de 30.

6.1.3. Controlador Proporcional - Integral - Derivativo (PID)

El parámetro ganancia K_p del controlador PID se estableció en 2.419, mientras que los parámetros T_i y T_d se fijaron en 0.484 y 0.647 respectivamente, para ver un control proporcional integral derivativo. Estos parámetros fueron determinados durante la etapa de simulación mediante el método del lugar geométrico de las raices.

Experimentación con Labview

En la Figura 6.8 se puede apreciar que este diseño cumple con las especificaciones, con un error de estado estacionario nulo, porcentaje de sobrepico es aproximadamente 15 % y tiempo de estabilización menor a 30 segundos.

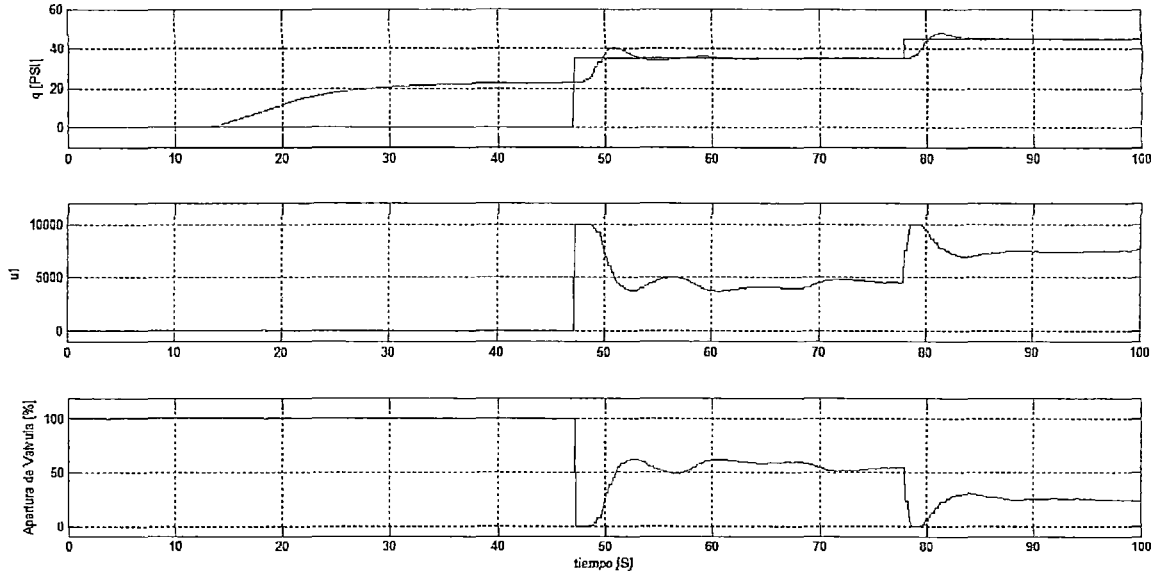


Figura 6.8: Respuesta del Sistema y Ley de Control ante un controlador PID.

Experimento con el Software SCADA

En el sistema SCADA se hizo una prueba real, en el que a K_p le dimos el valor de 2.419, T_i = 0.484, T_d = 0.647, el resultado se muestra en las Figuras 6.9 y 6.10.

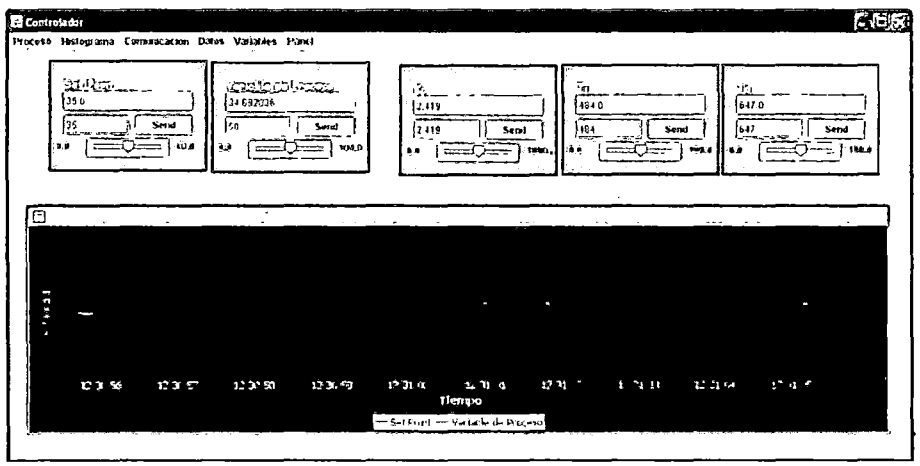


Figura 6.9: Respuesta del Sistema controlador PID, línea azul: Variable de Proceso, línea roja: Set Point.

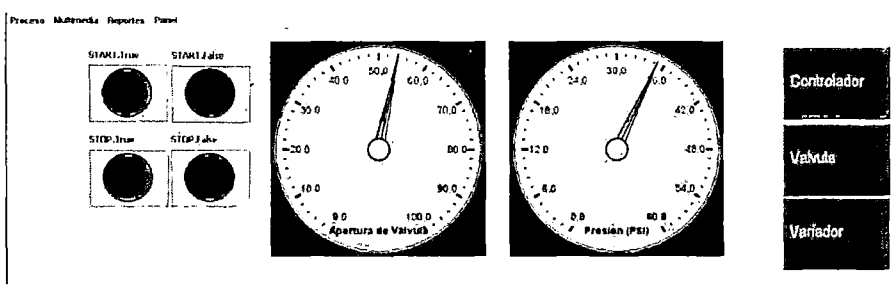


Figura 6.10: Panel Principal mostranto la respuesta del controlador PID ante un Set Point de 35.

CONCLUSIONES

- Aunque uno de los objetivos secundarios era desarrollar un controlador PID, durante las pruebas se verificó que el sistema responde de buena manera ante controladores tipo P y PI, siendo estos suficientes para realizar un control óptimo.

- Se comprobó que la función de transferencia obtenida en MATLAB, que aunque al ser simulada y comparada con la respuesta real del sistema, no cumplía con la respuesta en amplitud pero sí en la respuesta en tiempo, esto no fue un impedimento al momento de diseñar el controlador, ya que los resultados fueron satisfactorios, tanto en simulación como en la aplicación real.

- Se comprobó que los métodos de "Ziegler - Nichols" no puedes se usados en este proceso, debido a que no cumple con ninguna de las condiciones para la aplicación de los métodos.

- Se comprobó que la aplicación de software libre para un sistema SCADA funciona correctamente, es decir cumple con todos los requisitos para este proyecto, ofreciendo así una muy buena comunicación PC - PLC.

- Se cumplió con el objetivo secundario consistente en que los dos softwares libres implementados para este proyecto (SCADA - OPC), deben de ser amigables (de fácil uso), para que los alumnos de la escuela de Ingeniería Electrónica, adquieran nuevos conocimiento en el área de control.

- Antes de realizar un control en cualquier tipo de planta, debemos de estudiar su comportamiento, para así poder saber que método sería el más indicado a desarrollar y obtener buenos resultados.

RECOMENDACIONES

- Se recomienda utilizar un filtro de materiales sólidos, posiblemente una malla metálica, en la entrada de la válvula, ya que durante el mantenimiento previo que se hizo a la planta, al desarmar la válvula se encontró residuos sólidos en ella, posiblemente, que impedían su correcto funcionamiento.
- Se recomienda instalar todos los softwares necesarios para el trabajo en la planta de presión, en el sistema operativo XP, ya que todos los programas como LabView, Matlab, MatrikonOPC, SCADA, cuenta con el soporte solo para XP.

BIBLIOGRAFÍA

[1] Aguirre Zapata, David, "Desarrollo de un sistema SCADA para uso en pequeñas y medianas empresas.," 2013.

[2] J.Paulosová, L.Körösi. *Design of PID Controller for PLC*. Technical Report, Institute of Control and Industrial Informatics, Slovak University of Technology, 2010.

[3] Kuo, Benjamin C. *Sistemas de Control Automático*. Departament of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, 2012.

[4] MathWorks. System Identification Toolbox, User's Guide.

[5] MathWorks. Control System Toolbox, User's Guide.

[6] National-Instrument. LabVIEW Report Generation Toolkit for Microsoft Office User Guide.

[7] National-Instrumentl. LabVIEW, Control Design User Manual.

[8] Ogata, Katsuhiko. *Ingenieriá de Control Moderno*. University of Minesota, 2010.

[9] Osorio Diana, Flores Roa Camilo, "Obtención del modelo no paramétrico de un sistema por el método de identificación de respuesta en frecuencia," 2009.

[10] P. Arafet, F. Chang, M Torres H. Dominguez. *Métodos de Identificación dinámica*. Technical Report, Facultad de Ingeniería Eléctrica, Universidad de Oriente, 2008.

[11] Parra Rosero, Pablo. "Identificación de Procesos: usos de algoritmos en Matlab para encontrar un modelo por identificación para un proceso de medición de pH," *Tecnura*, (15):24–31 (2011).

[12] Ruge, Ilber A. "Optimización de señal de control en reguladores PID con arquitectura antireset Wind-up," *INGENIUS*, (6):3–8 (2011).

ANEXO A

PROGRAMA SOFT CONTROL

Desarrollado por el Ing. David Aguirre Zapata como proyecto de tesis, UDEP.
En el año 2008 se desarrolló un proyecto denominado: "Desarrollo e investigación de embebidos (TIC) para aplicación de automatización y control de bajo costo para pequeñas y medianas industrias", proyecto financiado por el FINCyT (Unidad coordinadora del programa de ciencia y tecnología) y desarrollado por la Universidad de Piura. Dicho proyecto tenía varios componentes, entre ellos: El desarrollo de tarjetas programables, software de bajo nivel y software de alto nivel, entre ellos un sistema SCADA.

El desarrollo del proyecto llevó más de 2 años de investigación, pruebas e informes en los distintos niveles de la organización del proyecto. Al final del mismo se obtuvieron resultados satisfactorios, tanto fue el éxito que algunos de los subproductos fueron comprados para su uso en determinadas empresas.

Con esto nació la idea de seguir desarrollando el software y mejorarlo, pues en esa versión el software tenía sus limitaciones, faltaba más investigación. Posteriormente al proyecto se desarrolló el denominado sistema "*Soft-Control*" teniendo como autor al Ing David Aguirre. "*Soft - Control*" es una versión mejorada del sistema SCADA implementado en ese proyecto.

ANEXO B

DATOS ADICIONALES SOBRE EL PLC MODICON M340

Simply Smart !
Más ingenio
e inteligencia
para una utilización
siempre más fácil.

Modicon M340
La opción *natural*

Modicon M340 surge del ingenio de las soluciones Telemecanique. Nace del profundo conocimiento de Modicon, desde los orígenes de los controladores programables.

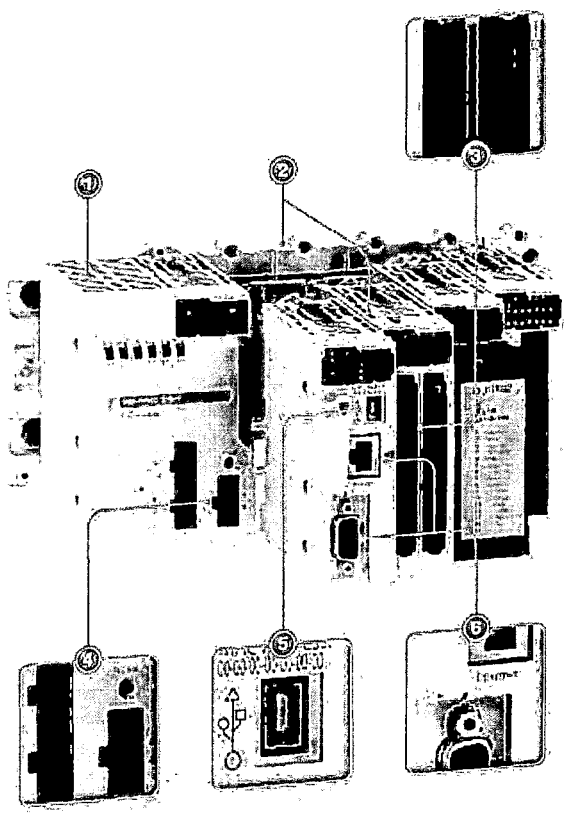
100% compacto, Modicon M340 representa una síntesis de potencia e innovación, y ofrece magníficas respuestas a las necesidades de los fabricantes de máquinas.

0% de preocupaciones: es el compañero ideal de Modicon Premium y Modicon Quantum, y responde a las necesidades de automatización de infraestructuras y procesos.
Asociado a la potencia y flexibilidad del software Unity Pro, le ofrece grandes ventajas durante todo el ciclo de vida de las aplicaciones.

Tecnología compacta 100%

Gabinetes "compactos"

Adopte las medidas convenientes de Modicon M340 y reduzca el tamaño de sus gabinetes.



① Tamaño
- Altura: 100 mm.
- Profundidad: 93 mm para su integración en armarios de 150 mm.

② Bastidor adaptable
- 4, 6, 8 o 12 módulos.
- Función de intercambio en funcionamiento "Hot Swap" para un mantenimiento sencillo.

③ Módulos de alta densidad
- 64 canales en tan sólo 32 mm.

④ Varias opciones de alimentación
- CA o CC.
- Salida de alimentación de sensores de 24 Vcc/0,9 A.

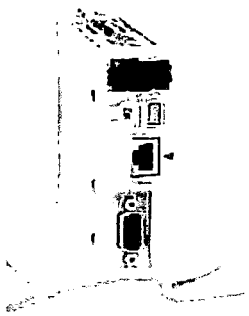
⑤ Puerto USB integrado

⑥ Puertos de comunicación
- Elija Modbus, Ethernet o CANopen.

Compacto

Pura energía, un equipo Potente!

Con su increible memoria y su asombroso rendimiento, este equipo potencia sus máquinas.



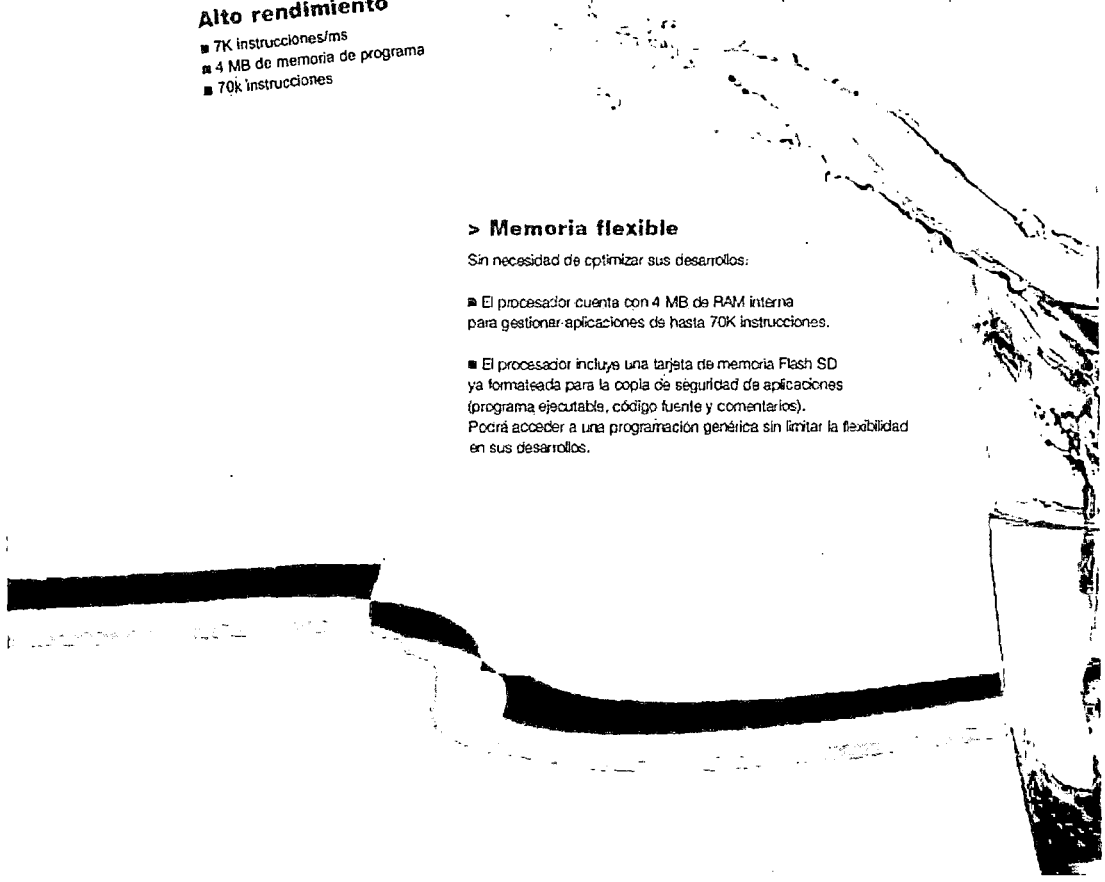
Alto rendimiento

- 7K instrucciones/ms
- 4 MB de memoria de programa
- 70k instrucciones

> Procesador de Alto Desempeño

Sean cuales fueren sus aplicaciones, sus preferencias o sus costumbres de programación, Modicon M340 siempre estará preparado.

- Con un alto rendimiento de procesamiento booleano, realizará cualquier cálculo aritmético con o sin decimales con suma facilidad. Especializado y a la vez, versátil: un nuevo equilibrio que podría convertirse rápidamente en su bien más preciado.
- ¿Desearía utilizar la potencia de los avanzados lenguajes IEC, pero piensa que su rendimiento podría verse afectado? Modicon M340 evita que el sistema se torne lento sea cual sea el lenguaje IEC que se utilice.
- ¿Procesamiento en mili-segundos? Es sencillo y posible con el sistema operativo multitarea de Modicon M340: tarea MAST, tarea FAST y 64 tareas de eventos. De esta forma, adaptará Modicon M340 a los aspectos esenciales de su aplicación.



> Memoria flexible

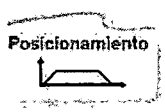
Sin necesidad de optimizar sus desarrollos:

- El procesador cuenta con 4 MB de RAM interna para gestionar aplicaciones de hasta 70K instrucciones.
- El procesador incluye una tarjeta de memoria Flash SD ya formateada para la copia de seguridad de aplicaciones (programa ejecutable, código fuente y comentarios). Podrá acceder a una programación genérica sin limitar la flexibilidad en sus desarrollos.

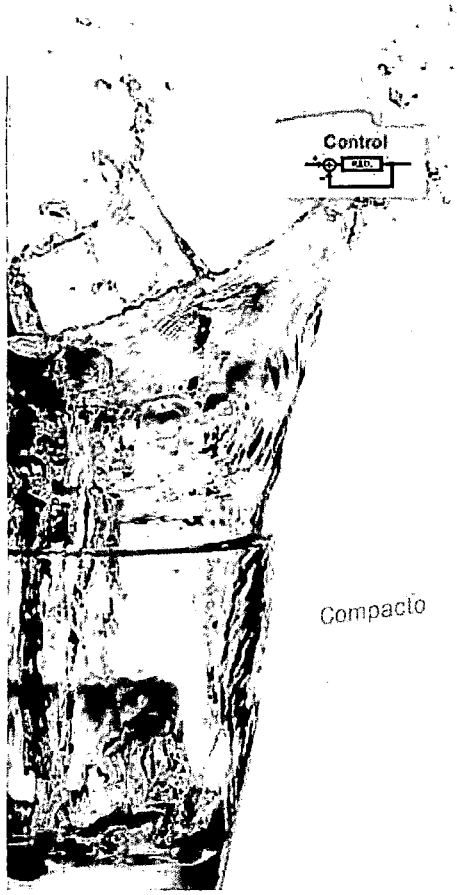
> Con alto contenido en experiencia

Debido a que las funciones tecnológicas son las que le hacen destacar, Modicon M340 le permite expresar sus conocimientos de una forma sencilla, siempre con una respuesta especializada.

■ La oferta de contaje de los controladores programables Modicon M340 incluye:
□ 2 módulos: 2 canales de 60 KHz y 8 canales de 10 KHz, alto rendimiento con contaje de 32 bits, duración del ciclo de 1 ms, 2 registros de captura y acciones reflejas con un tiempo de respuesta inferior a 500 microsegundos.
□ servicios avanzados configurables que ofrecen un filtro en cada entrada, una amplia gama de opciones reflejas, generador de impulsos, reductor de juego mecánico,
□ funciones configurables de modo de contaje y medición.



■ Está destinado a cubrir las necesidades de aplicaciones tales como: generación de alarmas de estado vacío de desbobinador, clasificación de pequeñas piezas, leva electrónica sencilla, control de velocidad.
■ Modicon M340 ofrece soluciones de posicionamiento integradas, flexibles y muy económicas para aplicaciones que necesiten ejes independientes coordinados y ejes maestro/esclavo.
■ Sin añadir ningún módulo de ejes. La solución se basa en una biblioteca MFB (bloques de función de movimiento) que cumple el estándar PLCopen. El control del servodrive y del variador de velocidad se lleva a cabo mediante comandos de MFB a través de CANopen.
■ Los MFB permiten que los variadores de velocidad Altivar controlen la velocidad de los motores, o que los variadores de velocidad Lexium o ICIA controlen la posición de los motores síncronos "Brushless".
■ Modicon M340, especialmente diseñado para los fabricantes de máquinas compactas, modulares y complejas, resulta muy adecuado para aplicaciones de manejo de materiales, transporte y embalaje secundario, así como para máquinas especiales y madereras.



■ El software Unity Pro incluye, de serie, una biblioteca de funciones de regulación. El lenguaje de bloques de función FBD IEC 61131-3 permite una programación completamente gráfica y sumamente flexible. De esta forma, podrá optimizar su algoritmo de regulación y llevar un control total del funcionamiento de su lazo.
■ Además de los reguladores PI y PID, la biblioteca incluye numerosos bloques avanzados:
□ ajuste automático de PID (sintonización automática),
□ integrador con limitación, limitador de variaciones de primer y segundo orden,
□ regulador de dos o tres posiciones, PI en caliente/frío, PIP y PPI en cascada,
□ generador de funciones,
□ conmutación de la estructura del regulador PD/PI,
□ modulación de ancho de impulso,
□ escalado de todos los valores de control.

Compacto

Inteligente

Robusto

Sencillo

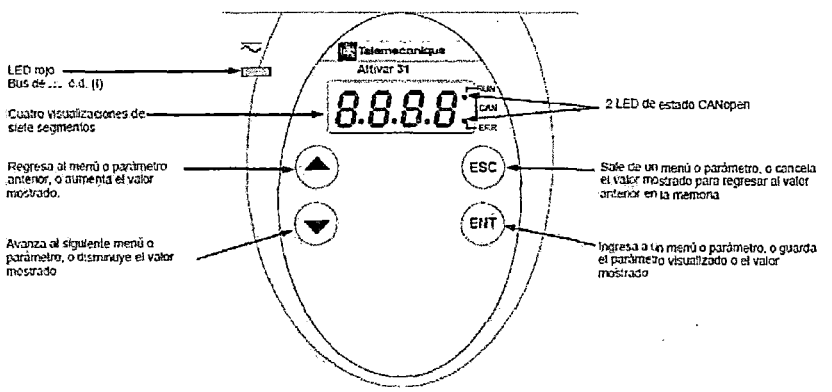
Potente

ANEXO C

DATOS ADICIONALES SOBRE EL VARIADOR ALTIVAR31

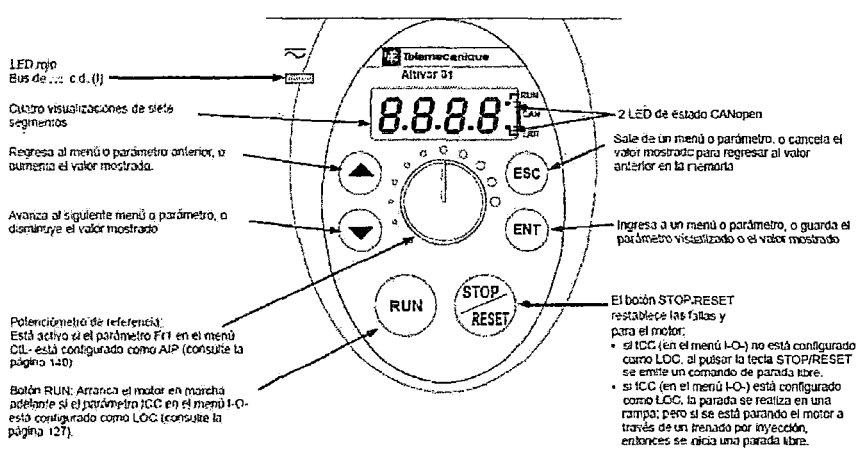
TERMINAL DE PROGRAMACIÓN Y AJUSTES DEL VARIADOR

Variadores ATV31••••••



Variadores ATV31••••••A

Los variadores ATV31••••••A tienen un potenciómetro de referencia, un botón de marcha y un botón de paro/restablecimiento.



Funciones de las teclas

- Para desplazarse por los datos rápidamente, pulse y mantenga oprimida (por más de 2 segundos) la tecla (▲) o (▼).
- Al presionar (▲) o (▼) su selección no se almacena automáticamente.
- Para guardar la selección, pulse (ENT). La terminal parpadea cuando almacena un valor.

Una visualización normal sin fallas ni comandos de marcha muestra:

- el valor de uno de los parámetros de visualización (consulte la página 178). La visualización por omisión es la frecuencia del motor, por ejemplo 43.0. La visualización parpadea en el modo de limitador de corriente.
- Init: secuencia de iniciación
- rdY: el variador está listo
- dcb: frenado por inyección de ⎓ (c.d.) en curso
- nSt: parada libre, consulte la siguiente sección.
- FSt: parada rápida
- tUn: autoajuste en curso

Si existe una falla, la visualización parpadea.

nSt: parada libre

Si la visualización muestra el código nSt, una de las siguientes condiciones puede estar sucediendo:

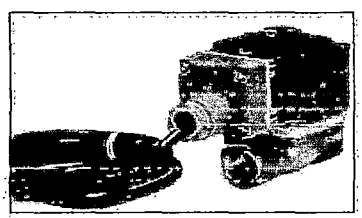
1. Con la configuración de fábrica, al energizar el variador de velocidad después de restablecer manualmente una falla o un comando de paro, los comandos de marcha adelante, marcha atrás y de paro por inyección de ⎓ (c.d.) se deberán restablecer para poner en marcha el variador. Si no se retablecen estos comandos el variador mostrará el mensaje "nSt" y no arrancará. Si la función de rearranque automático está configurada no es necesario restablecerlos.
2. Si el canal de referencia o el canal de control es asignado a Modbus o CANopen (consulte la página 130), el variador mostrará nSt al energizarlo y permanecerá parado hasta que el bus de comunicación envía un comando.
3. Si está presente un comando de marcha adelante o marcha atrás, cuando el variador es energizado y está configurado para un control de 2 ó 3 hilos con transición "trn" (consulte la página 127), el variador mostrará nSt y no se pondrá en marcha sino hasta que se suspende y vuelve a emitir el comando de marcha y se proporciona una referencia de velocidad.

ANEXO D

DATOS ADICIONALES DE LA VALVULA PROPORCIONAL

EV260B **NPT**

Características

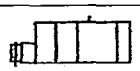


- Para la regulación progresiva del caudal en plantas industriales
- Tiempo de reacción corto
- Características lineales en el rango de regulación
- Se cierra ante una caída de tensión (función anticaídas)
- Tensión de 24 V cc
- De 4 a 20 mA estándar o de 0 a 10 V cc para señal de control
- Para agua, aceite y líquidos neutros similares
- Rango de caudal de agua: 0,8 - 5 m³/h (0,9 - 6 US gal/min)
- Protección de la bobina: IP 67

Datos técnicos, válvula

Instalación	Se recomienda un sistema de electroválvulas vertical
Rango de presión	0,5 - 10 bar (7,3 - 145 psi)
Presión de prueba	15 bar (216 psi)
Alcance	Mejor que 1:20 (5 - 100%)
Temperatura ambiente	–25 a +50°C (–13 a +122°F)
Temperatura del fluido	–10 a +80°C (+14 a +176°F)
Viscosidad	Máx. 50 cSt
Materiales	Cuerpo de la válvula: Latón, nº 2.0402 Armadura: Acero inoxidable, nº 1.4105 / AISI 430 FR Tubo de la armadura: Acero inoxidable, nº 1.4306/AISI 304 L Muelle: Acero inoxidable, nº 1.4568 Orificio: Acero inoxidable, nº 1.4305 / AISI 303 Vástago: Acero inoxidable, nº 1.4105 / AISI 430 FR Clapet: FKM Anillo del asiento y guía: PTFE Diafragma: PTFE Juntas tóricas: NBR

Opciones de la bobina



Modelo BK
sin generador de señales
señal de control de 300 a 600 mA
Véase pág. 4

Modelo BM
con generador de señales
señal de control de 0 a 10 V
Véase pág. 4

Modelo BL
con generador de señales
señal de control de 4 a 20 mA
Véase pág. 4

Datos técnicos, bobina

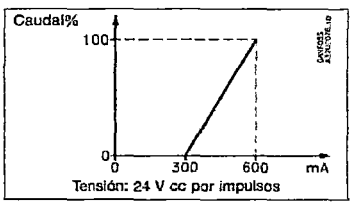
Tensión	Sin generador de señales: 24 V ±10%, tensión ca rectificada de onda completa Con generador de señales: 21 - 30 V cc
Señal de control	Sin generador de señales: 300 - 600 mA Con generador de señales: 4 - 20 mA o 0 - 10 V
Potencia bobina	Máx. 20 W
Aislamiento del bobinado	400 kΩ para la señal de control de 0-10 V, 250 Ω para la señal de control de 4-20 mA
Resistencia de la bobina	23,5 Ω a una temperatura ambiente de +20°C (+68°F)
Aislamiento del bobinado	Clase H de conformidad con el IEC 85
Conexión	*Sin generador de señales: Caja de terminales Pg 13.5* Con generador de señales: 3 cables núcleo de 2 m, Pg 13.5
Protección de la bobina, IEC 529	IP 67
Temperatura ambiente	–25°C a +50°C (-13 a 120°F)
Régimen de trabajo	Continuo

EV260B

Características de caudal

Modelo de bobina BK
Sin generador de señales

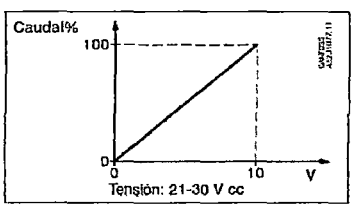
La versión básica consiste en una válvula con una bobina para corriente directa por impulsos. La tensión de 24 V CC se puede establecer con corriente alterna rectificada de onda completa. La válvula empieza a abrirse con una corriente de bobina de aprox. 300 mA y se abre completamente con una corriente de bobina de 600 mA. El ratio entre la corriente de bobina y el caudal entre los dos puntos extremos es directamente proporcional.



Tensión: 24 V cc por impulsos

Modelo de bobina BM
Con generador de señales y señal de control de 0-10

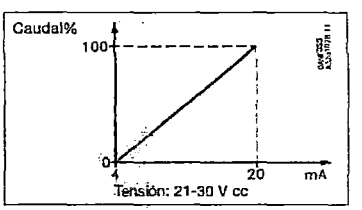
El ratio entre la señal de control y el caudal es directamente proporcional en el rango de *regulación.*



Tensión: 21-30 V cc

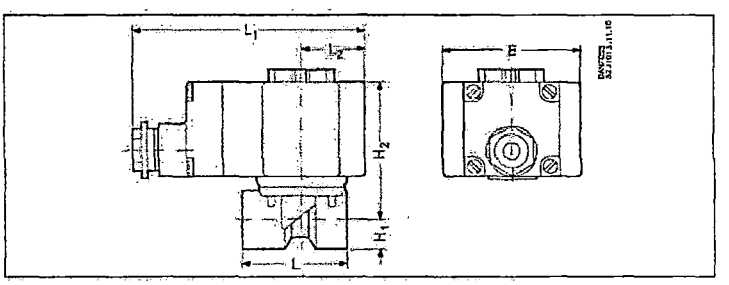
Modelo de bobina BL
Con generador de señales y control de 4-20 mA señal

El ratio entre la señal de control y el caudal es directamente proporcional en el rango de regulación.



Tensión: 21-30 V cc

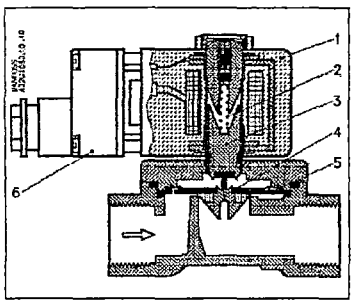
Dimensiones y peso



Modelo	L [mm]	L_1 [mm]	L_2 [mm]	H_1 [mm]	H_2 [mm]	B [mm]	Peso sin generador de señales [kg]	Peso con generador de señales [kg]
EV260B 6 B	62	112 ¹)	30	13	71	68	1,02	1,22
EV260B 10 B	62	112 ¹)	30	13	71	68	1,02	1,22
EV260B 15 B	81	112 ¹)	30	15	74	68	1,17	1,37
EV260B 20 B	98	112 ¹)	30	18	79	68	1,71	1,91

¹) Con el generador de señales, la medida L1 es de 128 mm.

Función



1. *Bobina*
2. *Muelle de cierre*
3. *Armadura*
4. *Orificio piloto*
5. *Diafragma*
6. *Caja de terminales*

La regulación proporcional de la apertura y cierre de las válvulas EV260B se alcanza mediante la regulación progresiva de la corriente de la bobina y de la fuerza de conexión de la bobina.
Cuando aumenta la corriente de la bobina, la fuerza de conexión de ésta (1) excederá en un punto concreto la fuerza equivalente del muelle de cierre (2). La armadura (3) se mueve verticalmente, abriendo el orificio piloto (4) del diafragma (5), el cual debido al efecto servo sigue el movimiento de la armadura.
La válvula se abre completamente cuando la corriente de la bobina alcanza su valor máximo.

Mediante la regulación progresiva de la corriente de la bobina, la armadura se puede colocar en cualquier posición en el tubo de la armadura y ajustar la válvula a cualquier posición entre completamente cerrada y completamente abierta.
El rango efectivo de la corriente de bobina para las válvulas proporcionales EV260B sin generador de señales es de aprox. 300-600 mA.
Las válvulas EV260B se encuentran también disponibles con un generador de señales incorporado en la caja de terminales (6) de la bobina. Los terminales de salida del generador de señales están conectados a la bobina.
El generador de señales regula la corriente de la bobina de manera que sea proporcional a la señal de entrada (señal de control).
La señal de control puede ser una
- señal de tensión de 0-10 V cc
  o una
- señal de corriente de 4 a 20 mA

Pedido

Válvula

Con. NPT	Material junta	Valor		Temperatura de fluido		Selección del modelo		Cód. sin bobina	Presión diferencial admisible	
				Mín.	Máx.	Mod. principal	Especificación		Mín.	Máx.[2])
[pulg.]		C_v [US gal/m]	K_v [m³/h]	[°C / °F]	[°C / °F]				[bar / psi]	[bar / psi]
1/4	PTFE	0,9	0,8	-10 / +14	+80 / +176	EV260B 6B	N14T NC000	032U8062	0,5 / 7,3	10 / 145
3/8	PTFE	0,9	0,8	-10 / +14	+80 / +176	EV260B 6B	N38T NC000	032U8063	0,5 / 7,3	10 / 145
3/8	PTFE	1,6	1,3	-10 / +14	+80 / +176	EV260B 10B	N38T NC000	032U8064	0,5 / 7,3	10 / 145
1/2	PTFE	1,6	1,3	-10 / +14	+80 / +176	EV260B 10B	N12T NC000	032U8065	0,5 / 7,3	10 / 145
1/2	PTFE	2,5	2,1	-10 / +14	+80 / +176	EV260B 15B	N12T NC000	032U8066	0,5 / 7,3	10 / 145
3/4	PTFE	6,0	5,0	-10 / +14	+80 / +176	EV260B 20B	N34T NC000	032U8067	0,5 / 7,3	10 / 145

Bobina

Descripción	Tensión	Señal de control	Especificación	Código
Con generador de señales	24 V ca rectificada de onda completa	300 - 600 mA	BK 024 D	018Z6987
Con generador de señales	21 a 30 V cc	0 - 10 V	BM 024 D	018Z0290
		4 - 20 mA	BL 024 D	018Z0291