INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS MONTERREY
DIVISIÓN DE INGENIERÍA Y ARQUITECTURA
PROGRAMA DE GRADUADOS EN INGENIERÍA

Controlador digital de procesos basado en Redes de Petri.

TESIS

PRESENTADA COMO REQUISITO PARCIAL PARA OBTENER EL GRADO ACADÉMICO DE:

MAESTRO EN CIENCIAS
ESPECIALIDAD EN AUTOMATIZACIÓN

POR:

LUIS ENRIQUE RAZO ZARAGOZA

MONTERREY, N.L. **DICIEMBRE DE 2005**

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS MONTERREY

DIVISIÓN DE INGENIERÍA Y ARQUITECTURA
PROGRAMA DE GRADUADOS EN INGENIERÍA

Los miembros del comité de tesis recomendamos que el presente proyecto de tesis presentado por el Ing. Luis Enrique Razo Zaragoza sea aceptado como requisito parcial para obtener el grado académico de:

Maestro en Ciencias
Especialidad en Automatización

Comité de Tesis:

_____
M. C. Luis Rosas Cobos
Asesor

_____
Dr. José de Jesús Rodríguez Ortiz
Sinodal

_____
M.C. Armando Céspedes Mota
Sinodal

Aprobado:

_____
Dr. Federico Viramontes Brown
Director del Programa de Graduados en Ingeniería
Diciembre, 2005

Controlador digital de procesos basado en Redes de Petri.

por

Luis Enrique Razo Zaragoza

Instituto Tecnológico de Monterrey

Campus Monterrey

Asesor de tesis: M.C. Luis Rosas Cobos

Resumen

En el presente trabajo se presenta la metodología para implementar controladores PID y difusos digitales, así como un nuevo controlador que se basa al igual que los controladores difusos en reglas del tipo *if then*, este nuevo controlador se puede utilizar en procesos no lineales y variantes en el tiempo. Otra de las características del controlador desarrollado es que no es necesario sintonizar ni requiere de un modelo matemático, lo cual facilita su implantación en cualquier proceso. El controlador mediante Redes de Petri realiza un cálculo aproximado del tiempo muerto del proceso a controlar en el primer cambio de referencia, y además en cada cambio de referencia calcula su ganancia, con estas acciones se logra controlar procesos variantes en el tiempo sin necesidad de mover parámetros en el controlador.

Para comprobar el buen desempeño del controlador desarrollado se presentan resultados obtenidos en cambios de referencia en un proceso no lineal simulado, el cual es el llenado automático de un tanque de agua cónico, así como cuatro procesos lineales de primer orden simulados. Se utiliza además un proceso real el cual es el control de temperatura. Se realiza una comparativa del desempeño de cada controlador en cada proceso, demostrando que el controlador desarrollado tiene un mejor desempeño que los controladores PID y difusos cuando existen cambios en los parámetros del proceso a controlar.

En el proceso de control de temperatura se muestra el desempeño de los tres controladores en 2 velocidades del motor, sintonizando los controladores PID y difuso en la primera velocidad. En el llenado automático de nivel de agua de un tanque cónico simulado se presenta el desempeño de los controladores en distintos niveles del tanque, mostrando claramente que el proceso se vuelve más lento en niveles más altos.

Adicionalmente se muestran las ecuaciones necesarias para simular procesos lineales de primer orden con tiempo muerto y un proceso no lineal, los cuales se utilizan para la demostración del desempeño de los controladores.

Finalmente se presentan conclusiones de los resultados obtenidos en las pruebas realizadas así como trabajos futuros que se pudieran realizar siguiendo la misma línea de investigación.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Antecedentes.

Actualmente el control automático es parte integral de la mayoría de los procesos de manufactura modernos, en los que se desea controlar variables tales como presión, temperatura, humedad, viscosidad, flujo, entre otros, ya que su utilización proporciona los medios para conseguir un comportamiento óptimo de los sistemas dinámicos, mejorar la productividad y simplificar el trabajo de muchas operaciones manuales repetitivas y rutinarias. Debido a los grandes avances logrados en el campo de la micro-electrónica, gran parte de los sistemas de control están basados en el control por computadora. Unas de las principales ventajas de las computadoras es la rapidez con la que realiza cálculos y su bajo costo.

En años recientes los avances en la teoría de control moderna se han centrado en el control robusto y adaptable. El control robusto busca que el desempeño del controlador no se vea afectado por pequeñas variaciones en los parámetros del proceso, y en control adaptativo el controlador se va ajustando en línea a los cambios paramétricos del proceso. Lo que se pretende tanto con el control robusto como con el control adaptable es mantener el buen desempeño del controlador por largos periodos de tiempo y de esta manera depender menos del experto en control.

Una de las limitaciones de los controladores convencionales PID es su mal desempeño ante no linealidades. Los sistemas lineales en la práctica no existen, ya que todos los sistemas físicos son no lineales en algún grado, pero cuando las magnitudes de las señales en un sistema de control están limitadas en intervalos en los cuales los componentes del sistema exhiben una característica lineal, el sistema es esencialmente lineal. Cuando las magnitudes de las señales se extienden más allá del intervalo de porción lineal, dependiendo de la severidad de la no linealidad, el sistema se debe considerar como no lineal. En los últimos años los controladores difusos han tenido una gran aceptación debido principalmente a su aplicación sobre modelos altamente no lineales, ya que en la vida real los sistemas no lineales son muy comunes. Otra ventaja de este tipo de controladores es que, a diferencia de los controladores convencionales

PID, no se requiere un modelo matemático del proceso, el cual en algunos casos en ambiente real (por ejemplo, un proceso de control industrial) es muy complicado obtenerlo. Una de las principales características de los sistemas difusos es que combinan la información de expertos humanos (lenguaje natural) con mediciones y modelos matemáticos basados en leyes físicas.

En trabajos recientes se han desarrollado nuevas técnicas de autosintonización de controladores PID lo cual ha facilitado y disminuido el número de pasos a seguir para adaptar un controlador de este tipo en un proceso. Los controladores difusos aunque no requieren de la ecuación matemática para adaptarse a un proceso, si requieren de un ajuste de sus conjuntos difusos. En el presente trabajo, que es el desarrollo de un controlador digital de procesos, se incorporan conceptos de lógica difusa (tales como la utilización del conocimiento del experto para no requerir de un modelo matemático del proceso), así como avances de autosintonización realizados en controladores PID, de tal manera que se obtenga un controlador robusto y adaptable, el cual requiere el mínimo número de pasos para incorporarse a un proceso y por lo tanto quien realice la adaptación no requiera conocimiento avanzado de ingeniería de control. Para tener un desarrollo estructurado del diseño del controlador se utilizan Redes de Petri, las cuales son una herramienta matemática de propósito general que sirven para describir las relaciones existentes entre condiciones y eventos. El motivo por el cual se eligió esta herramienta matemática es porque con ellas se puede modelar y visualizar tipos de comportamiento teniendo paralelismo, concurrencia, sincronización y recursos compartidos.

1.2. Objetivo.

Los objetivos del presente trabajo son:

1. Desarrollar un controlador digital que tenga las siguientes características:
    a. Que se pueda utilizar en procesos en estado inicial estable no lineales y/o variantes en el tiempo.
    b. Que tenga un mejor desempeño que los controladores convencionales PID y difusos ante variaciones en los parámetros del proceso.
    c. Que no requiere de un modelo matemático para su adaptación a un proceso.
    d. Que tenga una mínima interacción con el usuario.
2. Modelar el funcionamiento del controlador mediante Redes de Petri.

1.2.1. Alcances.

El controlador desarrollado está enfocado a procesos estables de una entrada una salida (SISO), lineales y no lineales, y variantes e invariantes en el tiempo. Para la evaluación del controlador se implementa adicionalmente un controlador PID y un controlador difuso, para poder realizar una comparativa de su desempeño.

Los procesos que se utilizan para la realización de pruebas son:

- Un proceso real para el control de temperatura.
- Procesos simulados lineales de primer orden.
- Un proceso no lineal simulado.

1.3. Contenido.

En el presente capítulo se presenta una introducción de los temas que se tratan en el presente trabajo, así como los objetivos y alcances.

En el capítulo dos se presenta la teoría de los conceptos que se utilizaron para la realización de los controladores implementados. Los temas que se incorporan son: control de procesos por computadora, transformada z, controladores PID digitales, sistemas difusos y Redes de Petri.

En el capítulo tres se presenta la Red de Petri que modela el funcionamiento del controlador desarrollado, y se define detalladamente cada uno de sus elementos. Adicionalmente en este capítulo se muestra la metodología utilizada para la elaboración de dicho controlador.

En el capítulo cuatro se describe las funciones de los íconos con los que cuenta la HMI (Human Machine Interface), y se muestran las ecuaciones matemáticas que se utilizaron tanto para realizar la simulación de procesos como para implementar los controladores PID y difuso.

En el capítulo cinco se muestra los resultados obtenidos de las pruebas realizadas con los tres controladores en los procesos simulados y el proceso real. Los procesos simulados son 4 lineales de primer orden y uno no lineal, el cual simula el llenado automático de un tanque de agua cónico.

Finalmente en el capítulo seis se presentan las conclusiones de los resultados obtenidos en las pruebas, y se mencionan trabajos adicionales que se pudieran realizar en el controlador mediante Redes de Petri.

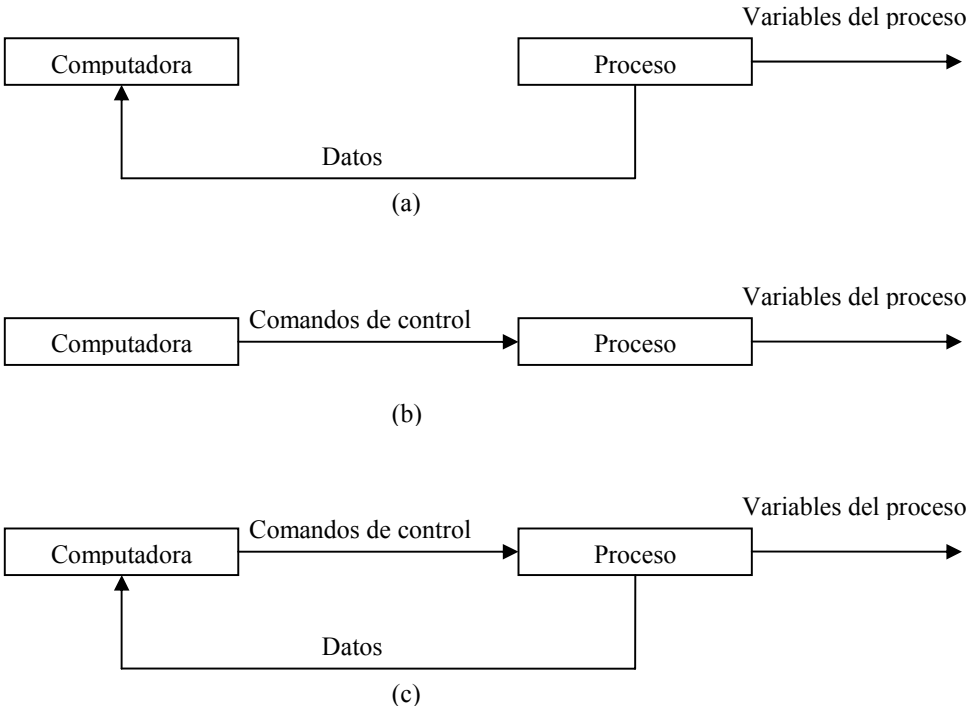
Capítulo 2

Marco Teórico

2.1. Control de procesos por computadora.

Existen varias formas en que se puede utilizar una computadora para controlar un proceso. Antes que nada hay que distinguir entre monitorear un proceso y controlar un proceso. En la figura 2-1 se muestran ambas acciones; cuando se monitorea, la computadora se utiliza únicamente para almacenar datos del proceso, y cuando se controla, la computadora regula el proceso. Existen dos tipos de sistemas de control:

- *Sistemas de control de lazo abierto.* Son aquellos en los cuales la respuesta del proceso no tiene efecto sobre la acción de control. En estos sistemas no se mide la salida ni se realimenta para compararla con la entrada. El usuario sólo tiene a su disposición la opción de manipular directamente el actuador.
- *Sistemas de control de lazo cerrado.* También denominados realimentados, en estos sistemas la respuesta del proceso tiene efecto sobre la acción de control, ya que ésta es medida y comparada con la entrada y dependiendo del resultado que se obtenga es la acción de control que se realiza. Este tipo de sistemas control es más común y es el que se utiliza para el desarrollo del controlador digital para procesos no lineales mediante Redes de Petri.



(a) Monitoreo, (b) sistema control de lazo abierto, y (c) sistema de control de lazo cerrado

Figura 2-1: Monitoreo y control por computadora.

La figura 2-2 muestra más detalladamente las partes de un sistema de control digital de lazo cerrado. La salida del proceso *y(t)* es una señal en tiempo continuo, la cual se convierte en una señal digital mediante el convertidor análogo digital (A-D). La conversión se hace en el tiempo de muestreo, *T*. La computadora interpreta la señal recibida, $\{y(kT)\}$, como una secuencia de números, procesa las mediciones utilizando algoritmos, y da una nueva secuencia de números, $\{u(kT)\}$. Esta nueva secuencia es convertida a una señal análoga con el convertidor digital análogo (D/A). Los eventos son sincronizados por el reloj en tiempo real en la computadora. Aunque la salida de la computadora se envía en intervalos de tiempos, el convertidor (D/A) debe producir una señal continua. Para poder producir dicha señal normalmente se conserva la señal de control constante hasta recibir otra señal de la computadora.

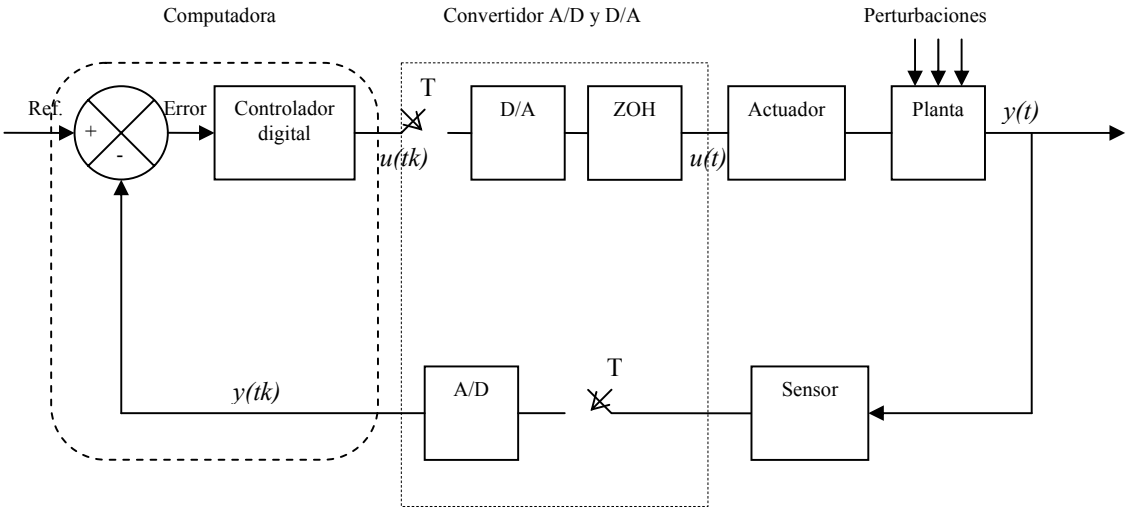


Figura 2-2: Lazo de control digital.

A continuación se describen los componentes de un lazo de control digital:

- *Convertidor análogo digital (A/D):* Es un dispositivo que convierte una señal análoga en una señal digital, el convertidor se utiliza como interfase entre un componente análogo y un componente digital. La conversión de una señal análoga en su correspondiente señal digital (número binario) es una aproximación, debido a que la señal análoga puede tomar un número infinito de valores.
- *Convertidor digital análogo (D/A):* Es un dispositivo que convierte una señal digital en una señal análoga. Este convertidor se utiliza como interfase entre un componente digital y un componente análogo.
- *Planta:* Es el proceso físico que se desea controlar. En el diseño de un controlador digital es de gran importancia la exactitud con la que se modela la planta.
- *Actuador:* Es el dispositivo que ejecuta las acciones necesarias para lograr los cambios deseados en la variable controlada.
- *Sensor:* Es el dispositivo que se encarga de medir el valor actual de la variable controlada, y normalmente cuenta con un transmisor para enviar la señal al controlador.
- *Controlador:* Se encarga de calcular el error, que es la referencia menos la variable controlada, y envía la manipulación necesaria para reducirlo. En este dispositivo reside el algoritmo de control.

- *Transductor:* Son dispositivos que convierten una señal de entrada en una señal de salida en otra forma, por ejemplo un dispositivo que convierte una señal de presión en una señal de voltaje. Se utilizan cuando el controlador, sensor y/o actuador no se pueden conectar directamente ya que utilizan distintos tipos de señales para su operación.
- *Retenedor de orden cero (ZOH):* El retenedor recibe una señal de entrada análoga y la mantiene constante por un periodo de tiempo específico, hasta que llegue la nueva señal.

En un lazo cerrado de control se tienen además las siguientes variables:

- *Variable controlada:* Es la variable que se desea controlar y mantener en un valor determinado. Normalmente la variable controlada es la salida del sistema de control.
- *Referencia o Set-point:* Es el valor al que se desea tener la variable controlada.
- *Manipulación:* Es la acción correctiva que envía el controlador al actuador para reducir el error.
- *Perturbaciones:* Son efectos que modifican el valor de la variable controlada, y por lo tanto alteran de forma negativa el desempeño del lazo de control.

2.2. Transformada *z*.

Una herramienta matemática que comúnmente se utiliza para el análisis y síntesis de sistemas de control discretos es la transformada *z*. El papel que desempeña la transformada *z* en sistemas discretos es similar al que desempeña la transformada de Laplace en sistemas continuos. Con la transformada *z* la solución de ecuaciones de diferencias lineales se logra mediante operaciones algebraicas. La transformada *z* de una función en el tiempo *x(t)* (donde *t* no es negativo), o de una secuencia de valores *(kT)*, (donde *k* es cero o un entero positivo y *T* es el periodo de muestreo), está definida por la siguiente ecuación:

$$X(z) = Z[x(t)] = Z[x(kT)] = \sum_{k=0}^{\infty} x(kT) z^{-k} \tag{2.1}$$

Para una secuencia de números *x(k)*, la transformada *z* esta definida por:

$$X(z) = Z[x(k)] = \sum_{k=0}^{\infty} x(k) z^{-k} \tag{2.2}$$

Las ecuaciones 2.1 y 2.2 se refieren a la transformada *z* unilateral. Las ecuaciones 2.3 y 2.4 se refieren a la transformada *z* bilateral, donde para *x(t)* $-\infty < t < \infty$, y para *x(k)* $k = 0, \pm 1, \pm 2, \cdots$

$$X(z) = Z[x(t)] = Z[x(kT)] = \sum_{k=-\infty}^{\infty} x(kT) z^{-k} \tag{2.3}$$

$$X(z) = Z[x(k)] = \sum_{k=-\infty}^{\infty} x(k) z^{-k} \tag{2.4}$$

Para la mayoría de las aplicaciones de ingeniería, la transformada *z* unilateral tendrá una forma cerrada conveniente dentro de su región de convergencia. La expansión de la ecuación 2.1 es:

$$X(z) = x(0) + x(T)z^{-1} + x(2T)z^{-2} + \cdots + x(kT)z^{-k} + \cdots \tag{2.5}$$

La ecuación 2.5 implica que la transformada *z* de cualquier función de tiempo continua *x(t)* puede ser escrita en forma de serie, donde z^{-k} indica la posición en tiempo en la que la amplitud *x(kT)* ocurre. Si *X(z)* se da en la forma de la ecuación 2.5, la transformada *z* inversa se puede obtener como una secuencia de la función *x(kT)* que corresponde a los valores de *x(t)* en los instantes de tiempo respectivos.

En la figura 2-3 se muestra un diagrama de la forma en que se utiliza la transformada *z* para realizar operaciones de forma más sencilla.

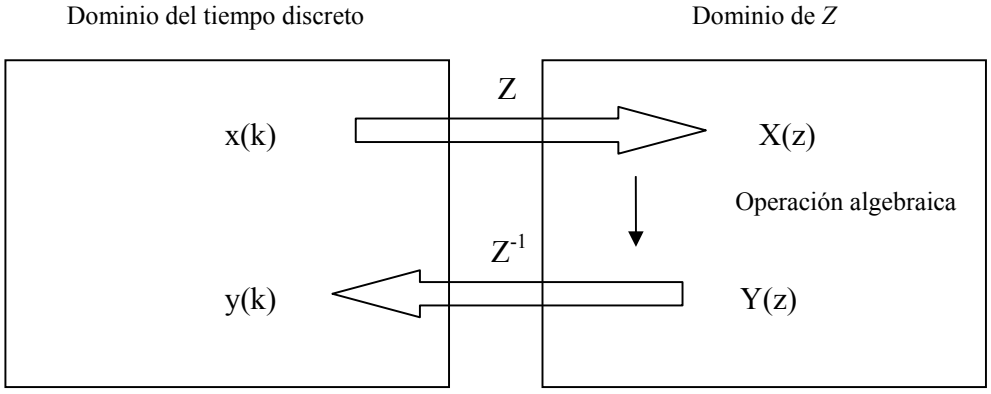


Figura 2-3: Transformada *z*.

Al igual que para la transformada de Laplace, se pueden encontrar tablas con las transformaciones *z* más comunes. En la tabla 2.1 se muestran algunas de ellas.

X(s)	*X(t)*	*x(kT) ó x(k)*	*X(z)*
		$\delta_o(k)$	

-	-	1, $k = 0$ 2, $k \neq 0$	1
-	-	$\delta_o(n-k)$ 1, $n = k$ 2, $n \neq k$	z^{-k}
$\frac{1}{s}$	$1(t)$	$1(k)$	$\frac{1}{1-z^{-1}}$
$\frac{1}{s+a}$	e^{-at}	e^{-akT}	$\frac{1}{1-e^{-aT}z^{-1}}$
$\frac{1}{s^2}$	T	kT	$\frac{Tz^{-1}}{(1-z^{-1})^2}$
$\frac{2}{s^3}$	T^2	$(kT)^2$	$\frac{T^2z^{-1}(1+z^{-1})}{(1-z^{-1})^3}$
$\frac{6}{s^4}$	T^3	$(kT)^3$	$\frac{T^3z^{-1}(1+4z^{-1}+z^{-2})}{(1-z^{-1})^4}$
$\frac{a}{s(s+a)}$	$1-e^{-at}$	$1-e^{-akT}$	$\frac{(1-e^{-aT})z^{-1}}{(1-z^{-1})(1-e^{-aT}z^{-1})}$
$\frac{b-a}{(s+a)(s+b)}$	$e^{-at}-e^{-bt}$	$e^{-akT}-e^{-bkT}$	$\frac{(e^{-aT}-e^{-bT})z^{-1}}{(1-e^{-aT}z^{-1})(1-e^{-bT}z^{-1})}$
$\frac{1}{(s+a)^2}$	Te^{-at}	kTe^{-akT}	$\frac{Te^{-aT}z^{-1}}{(1-e^{-aT}z^{-1})^2}$
$\frac{s}{(s+a)^2}$	$(1-at)e^{-at}$	$(1-akT)e^{-akT}$	$\frac{1-(1+aT)e^{-aT}z^{-1}}{(1-e^{-aT}z^{-1})^2}$
$\frac{2}{(s+a)^3}$	T^2e^{-at}	$(kT)^2e^{-akT}$	$\frac{T^2e^{-aT}(1+e^{-aT}z^{-1})z^{-1}}{(1-e^{-aT}z^{-1})^3}$
$\frac{\omega}{s^2+\omega^2}$	$Sen\ \omega t$	$sen\ \omega kT$	$\frac{z^{-1}sen\omega T}{1-2z^{-1}\cos\omega T+z^{-2}}$
$\frac{s}{s^2+\omega^2}$	$cos\ \omega t$	$cos\ \omega kT$	$\frac{1-z^{-1}\cos\omega T}{1-2z^{-1}\cos\omega T+z^{-2}}$
$\frac{\omega}{(s+a)^2+\omega^2}$	$e^{-at}sen\ \omega t$	$e^{-akT}sen\ \omega kT$	$\frac{e^{-aT}z^{-1}sen\omega T}{1-2e^{-aT}z^{-1}\cos\omega T+e^{-2aT}z^{-2}}$

$\frac{s+a}{(s+a)^2+\omega^2}$	$e^{-at}\cos\omega t$	$e^{-akT}\cos\omega kT$	$\frac{1-e^{-aT}z^{-1}\cos\omega T}{1-2e^{-aT}z^{-1}\cos\omega T+e^{-2aT}z^{-2}}$
		a^k	$\frac{1}{1-az^{-1}}$
		a^{k-1} $k=1,2,3\ldots$	$\frac{z^{-1}}{1-az^{-1}}$

Tabla 2.1: Transformaciones *z* comunes.

$x(t) = 0$, para $t<0$. $x(kT) = x(k) = 0$, para $k<0$.

La transformada *z* tiene las siguientes propiedades:

Multiplicación por una constante	$Z[ax(k)] = aX(z)$
Suma	$Z[x_1(k) \pm x_2(k)] = X_1(z) \pm X_2(z)$
Traslación en k	$Z[x(k-1)] = z^{-1}X(z)$ $Z[x(k+1)] = zX(z) - zX(0)$ $Z[x(k+n)] = z^n X(z) - z^n x(0) - z^{n-1}x(1) - \ldots - zx(n-1)$
Suma sobre k	$Z\left[\sum_{k=0}^{n} x(k)\right] = \frac{X(z)}{1-z^{-1}}$
Diferencias Vx(k)	$Z[x(k) - x(k-1)] = (1-z^{-1})X(z)$
Multiplicación por e^{-akT}	$Z\left[e^{-akT}x(k)\right] = X(ze^{aT})$
Multiplicación por kT	$Z[kTx(k)] = -Tz\frac{d}{dz}X(z)$
Teorema del valor inicial	$x(0) = \lim_{z\to\infty} X(z)$ *(si el límite existe)*
Teorema del valor final	$\lim_{k\to\infty} x(k) = \lim_{z\to\infty}(1-z^{-1})X(z)$ *(si polos* $(1-z^{-1})X(z)$ *dentro C.U.)*

Tabla 2.2: Propiedades de la Transformada *z*.

La notación para la transformada *z* inversa es Z^{-1}. La transformada *z* inversa es el proceso de pasar de la expresión $X(z)$ de una función, a su expresión $x(k)$ (secuencia de números). Otros tres métodos, aparte de por tablas (tabla 2.1), para obtener la transformada *z* inversa son:

- *Método de división directa.* Con este método se obtiene la transformada *z* inversa mediante la expansión de $X(z)$ en una serie de potencia infinita en z^{-1}. Este método se utiliza

generalmente cuando es muy complicado por otros métodos, o cuando se desea encontrar únicamente los primeros términos de *x(k)*.

$$X(z) = \sum_{k=0}^{\infty} x(k)z^{-k} = x(0) + x(1)z^{-1} + x(2)z^{-2} + \cdots + x(k)z^{-k} + \cdots \qquad (2.6)$$

Después de realizar la expansión, *x(k)* es el coeficiente del término z^{-k}, y de esta manera se pueden determinar los valores de *x(k)* para *k = 0,1,2,..*

- *Método de la integral de inversión.* La integral de inversión está dada por:

$$Z^{-1}\left[X(z)\right] = x(kT) = x(k) = \frac{1}{2\pi j}\oint_C X(z)z^{k-1}dz \qquad (2.7)$$

Donde C es un círculo con centro en el origen y de radio tal que todos los polos de $X(z)z^{k-1}$ están adentro de él.

- *Método de expansión en fracciones parciales.* Si *X(z)* tiene uno o más ceros en el origen (z = 0), entonces X(z)/z o X(z) es expandible en una suma de términos de primer y segundo orden mediante fracciones parciales, y después de la expansión, con la utilización de tablas se puede encontrar su función correspondiente en el tiempo.

2.3. Controladores PID digitales.

Los controladores PID cuentan con acciones proporcional, integral y derivativa para disminuir el error, es decir, para que la variable controlada sea igual a la referencia. La acción proporcional genera una salida del controlador proporcional al error. La acción integral ayuda a eliminar los errores de estado estable, ya que la acción proporcional por si sola no puede corregirlos. Finalmente la acción derivativa ayuda a corregir errores anticipadamente, ya que predice el error mediante la medición de la velocidad de cambio del error. Existen diversas estructuras de controladores PID´s, y algunas de las más comunes son las siguientes:

- *Estructura ideal del PID.*

$$m(t) = K_c\left(e(t) + \frac{1}{\tau_i}\int e(t)dt + \tau_d\frac{de(t)}{dt}\right) \qquad (2.8)$$

en Laplace se presenta como:

$$M(s) = K_c\left(1 + \frac{1}{\tau_i s} + \tau_d s\right)E(s) \qquad (2.9)$$

donde;

K_c = Ganancia proporcional

τ_i = Constante de tiempo integral

τ_i = Constante de tiempo integral

$E(s)$ = Error

$M(s)$ = Manipulación

- *Estructura clásica.*

$$M(s) = K_c\left(1+\frac{1}{\tau_i s}\right)\left(\frac{1+\tau_d s}{1+\tau_a s}\right)E(s) \tag{2.10}$$

τ_a es la constante de tiempo de un filtro, se recomienda $0.33\tau_d \geq \tau_a \geq 0.05\tau_d$

- *Estructura no interactiva.*

$$M(s) = \left(K_c + \frac{1}{\tau_i s}\right)E(s) - \left(\frac{\tau_d s}{1+\tau_a s}\right)Y(s) \tag{2.11}$$

$Y(s)$ es la salida del proceso

- *Estructura industrial.*

$$M(s) = \left(1+\frac{1}{\tau_i s}\right)\left(R(s) - \frac{1+\tau_d s}{1+\tau_a s}\right)Y(s) \tag{2.12}$$

Los controladores PID se deben sintonizar, es decir, ajustar sus parámetros para que tenga un desempeño satisfactorio. Existen varios métodos para realizar la sintonización. Algunos de ellos son: de prueba y error, de ganancia última, de respuesta al escalón, de criterios integrales, entre otros. En el presente trabajo se utiliza el método de criterios integrales para sintonizar el controlador PID. En este método se pretende minimizar la suma del error a través del tiempo, y para lograrlo se proponen los siguientes índices de desempeño:

- *Integral del valor absoluto del error (IAE).*

$$\int_0^\infty |e(t)|dt \tag{2.13}$$

- *Integral del cuadrado del error (ISE).*

$$\int_0^\infty e^2 dt \tag{2.14}$$

- *Integral del tiempo por el valor absoluto del error (ITAE).*

$$\int_0^\infty t|e(t)|dt \tag{2.15}$$

Existen tablas para el cálculo de los parámetros del controlador para cada uno de los criterios anteriores, para controlar la variable de proceso ante cambios de referencia o perturbaciones. Si se elige la sintonización ante cambios en referencia, la corrección de perturbaciones es muy lenta; y si se elige ante perturbaciones se presentan sobretiros de hasta el 50% en cambios de referencia.

Para realizar un controlador PID digital se debe de discretizar la función de transferencia, y para realizar dicho proceso se puede utilizar alguna de las siguientes transformaciones digitales:

- *Diferencias hacia atrás.*

$$s = \frac{1 - z^{-1}}{T} \tag{2.16}$$

- *Diferencias hacia delante.*

$$s = \frac{1 - z^{-1}}{z^{-1}T} \tag{2.17}$$

- *Aproximación de Tustin o transformación bilineal.*

$$s = \frac{2}{T}\frac{1 - z^{-1}}{1 + z^{-1}} \tag{2.18}$$

La transformación bilineal es la que presenta una mejor aproximación, sin embargo polos estables en Laplace pueden convertirse en polos con tendencia a la inestabilidad o timbre en *Z*, por tanto para garantizar la persistencia de la estabilidad la técnica de diferencias hacia atrás puede ser utilizada.

2.4. Sistemas difusos.

Los sistemas difusos combinan la información de expertos humanos (lenguaje natural) con mediciones y modelos matemáticos basados en leyes físicas. Los sistemas difusos transforman la base de conocimiento humano en una fórmula matemática. Los sistemas difusos son sistemas basados en conocimiento ó sistemas basados en reglas difusas del tipo IF-THEN (Si-entonces)
Si la velocidad del carro es alta, entonces aplicar menos fuerza al acelerador.

"Alta" y "menos" son conjuntos difusos (funciones de membresía) de las variables velocidad y fuerza al acelerador.

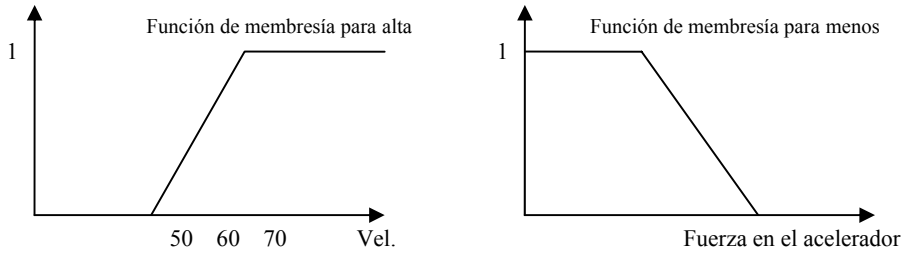


Figura 2-4: Ejemplo de funciones de membresía.

Los conjuntos difusos puede ser de diversas formas, tal y como se muestra en la figura 2-5.

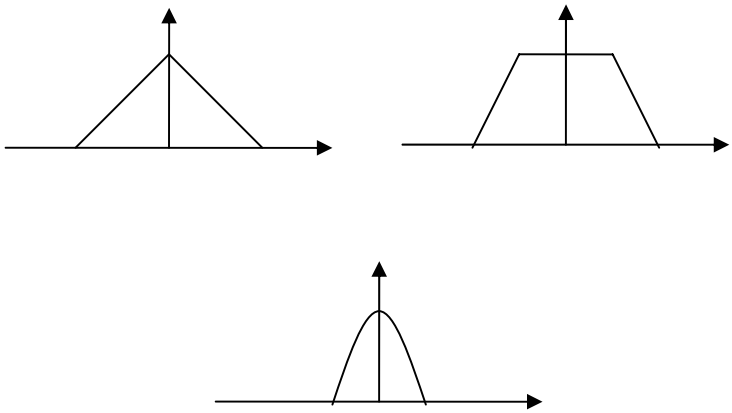


Figura 2-5: Conjuntos difusos.

Los sistemas difusos se pueden usar para modelar procesos ó fenómenos no lineales. Existen los siguientes tipos de sistemas difusos:

a) Sistemas difusos puros.

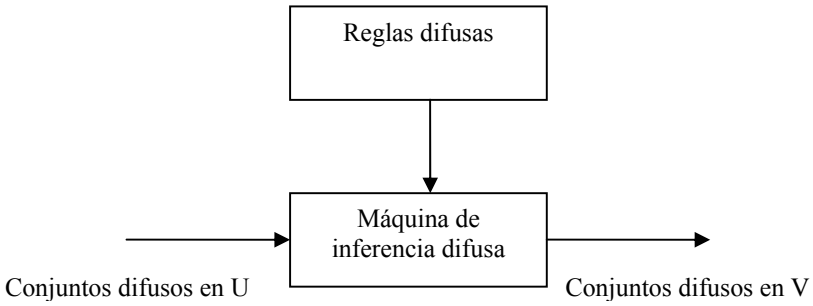


Figura 2-6: Sistema difuso puro.

La máquina de inferencia combina las reglas difusas en un mapeo de los conjuntos difusos de la entrada a los conjuntos difusos de la salida basado en principios de lógica difusa.

b) Sistemas tipo Takagi-Sugeno-Kant.

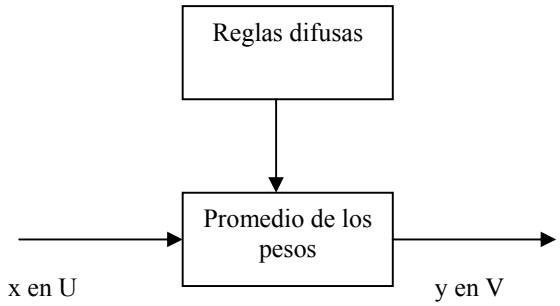


Figura 2-7: Sistema difuso del tipo Takagi-Sugeno-Kant.

c) Sistemas difusos con difusificador y desdifusificador.

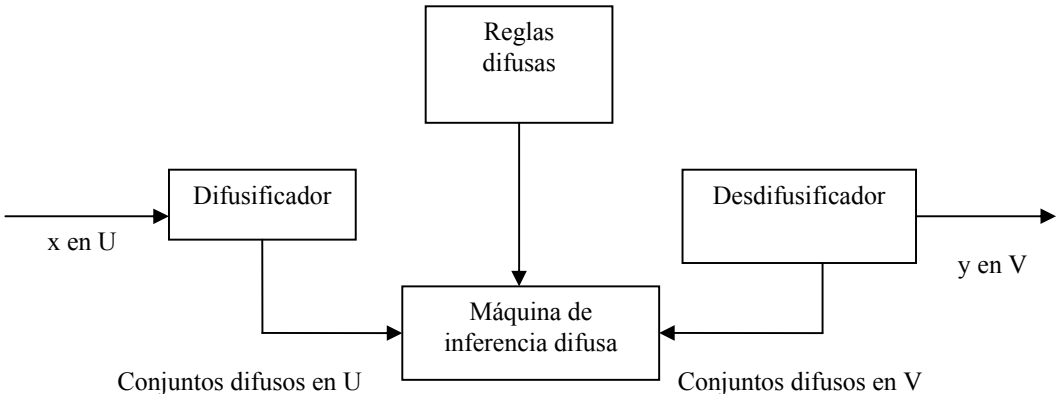


Figura 2-8: Sistema difuso con difusificador y desdifusificador.

Los sistemas difusos se han aplicado en diversas áreas como medicina, administración, manufactura, comunicaciones, procesamiento de señales, control automático, entre otros. En productos del consumidor han tenido mucho éxito usados en lavadoras, cámaras de video, transmisiones de automóviles. En control automático se han usado en hornos de cemento y control de velocidad de tren de pasajeros.

Un conjunto difuso en el universo U se caracteriza por la función de membresía $\mu_A(x)$ que toma el intervalo $[0,1]$. El conjunto difuso A se puede representar por:

$$A = \{(x, \mu_A(x)) | x \in U\} \tag{2.19}$$

Un conjunto difuso A también se puede denotar como:

Para X discreto:

$$A = \sum_{x_i \in X} \mu_A(x_i) / x_i \tag{2.20}$$

Para X continuo:

$$A = \int_X \mu_A(x_i) / x_i \tag{2.21}$$

Operaciones básicas de los conjuntos difusos son:

- *Complemento.*

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x) \tag{2.22}$$

- *Unión.*

$$\mu_{A \cup B}(x) = \max\left[\mu_A(x), \mu_B(x)\right] \tag{2.23}$$

- *Intersección.*

$$\mu_{A \cap B}(x) = \min\left[\mu_A(x), \mu_B(x)\right] \tag{2.24}$$

El soporte de un conjunto difuso A en un universo U, es un conjunto que contiene todos los elementos de U para los cuales el valor de la función de membresía en A es distinta de cero. Un singleton difuso es un conjunto difuso cuyo soporte es un solo punto en U. El punto Crossover de un conjunto difuso es el punto en U donde el valor de la función de membresía en A es 0.5. El peso de un conjunto difuso es el valor máximo obtenido por alguno de los puntos. En un conjunto difuso normal el peso es igual a uno.

Una variable lingüística es una variable cuyo valor son palabras o frases en un lenguaje natural o artificial. Una variable numérica toma valores numéricos: *Edad = 75.* Una variable lingüística toma valores lingüísticos: *Edad viejo.* Todos los valores lingüísticos forman un conjunto de términos. $T_{(edad)}$={joven, no joven, muy joven, medio joven, no medio joven, viejo, no viejo, muy viejo, más o menos viejo, no tan viejo…}. Una variable lingüística es caracterizada por (X, T, U, M) donde: X es el nombre de la variable lingüística (ej. edad), T es el conjunto de los valores lingüísticos que X puede tomar (ej. T={joven, medio joven, viejo}), U es el dominio actual en donde la variable lingüística X toma valores (ej. U=[0,100]), y M es una regla semántica que relaciona cada valor lingüístico con un conjunto difuso en U (ej. M relaciona joven, medio joven y viejo con las MF´s). Las variables lingüísticas son los elementos más fundamentales en la representación del conocimiento humano.

Los valores de una variable lingüística es un término compuesto por una concatenación de términos individuales. La clasificación de los términos individuales es:

- Término primario, etiquetas del término difuso, ej. joven, viejo…
- Complemento "no" y conexiones "y" y "o".
- Modificadores tales como "muy", "más o menos"…

Los modificadores más comunes son muy y más o menos que están de finidos por:

$$\mu_{muy} A(x) = [\mu A(x)]^2 \tag{2.25}$$

$$\mu_{más\ o\ menos} A(x) = [\mu A(x)]^{1/2} \tag{2.26}$$

If-then. El conocimiento humano es representado en términos de reglas difusas si-entonces. Un regla difusa si-entonces es una condición expresada como:
SI < proposición difusa >, entonces < proposición difusa >.

Existen dos tipos de proposiciones difusas: proposiciones difusas individuales y compuestas. Una proposición difusa individual es un solo estado, ej. *x es A, x es una variable lingüística y A es un conjunto difuso.* Una proposición difusa compuesta es una composición de proposiciones individuales usando los conectores "y", "o" y "no". Las proposiciones difusas compuestas deben ser entendidas como relaciones difusas y las MF´s de esas relaciones son calculadas usando la norma T (intersección), S (unión) y complemento. La intersección difusa se utiliza para el conector "y". La unión difusa se utiliza para el conector "o". Y el complemento difuso se utiliza para el conector "no".

Las reglas de inferencia describen las características de un sistema difuso mediante la relación difusa entrada-salida. Un sistema difuso de múltiples entradas y salidas se puede descomponer en una colección de sistemas entrada-salida simples. El número de reglas se incrementa exponencialmente con la dimensión de la entrada. Se dice que un conjunto de reglas difusas si-entonces es consistente si no hay reglas con la misma parte si, pero con distinta parte entonces.

Las fórmulas de las máquinas de inferencia más utilizadas para sistemas difusos y control difuso son:

- *Máquina de inferencia producto:*

$$\mu B'(y) = \max_{l=1}^{M} [\sup_{x \in U} (\mu A'(x) \prod_{i=1}^{n} \mu A_i^l(xi) \mu B^l(y)] \tag{2.27}$$

Combinación de regla individual base mediante unión. Se utiliza implicación producto de Mamdani, producto algebraico para norma t, y máximo para norma s.

- *Máquina de inferencia mínimo:*

$$\mu B'(y) = \max_{l=1}^{M} [\sup_{x \in U} \min(\mu A'(x), \mu A_1^l(x_1), ..., \mu A_n^l(x_n), \mu B^l(y))]\} \tag{2.28}$$

Combinación de regla individual base mediante unión. Se utiliza implicación mínimo de Mamdani, mínimo para norma t, y máximo para norma s.
Si el conjunto A' es un singleton, dado por;

$$\mu A'(x) = \begin{cases} 1 & si \quad x = x^* \\ 0 & \sin o \end{cases} \tag{2.29}$$

La máquina de inferencia producto se puede reducir:

$$\mu B'(y) = \max_{l=1}^{M}[\prod_{i=1}^{n} \mu A_i^l(xi^*)\mu B^l(y)] \tag{2.30}$$

La máquina de inferencia mínimo se puede reducir:

$$\mu B'(y) = \max_{l=1}^{M}[\min(\mu A_1^l(x_1^*),...,\mu A_n^l(x_n^*), \mu B^l(y))]\} \tag{2.31}$$

Máquina de inferencia de Lukasiewicz:

$$\mu B'(y) = \min_{l=1}^{M}\{\sup_{x\in U} \min[\mu A'(x), 1 - \min_{i=1}^{n}(\mu A_i^l(xi)) + \mu B^l(y))]\} \tag{2.32}$$

Máquina de inferencia de Zadeh:

$$\mu B'(y) = \min_{l=1}^{M}\{\sup_{x\in U} \min[\mu A'(x), \max(\min(\mu A_i^l(xi),...,\mu A_i^l(x_n), \mu B^l(y)), 1 - \min_{i=1}^{n}(\mu A_i^l(xi)))]\} \tag{2.33}$$

Máquina de inferencia de Dienes-Rescher:

$$\mu B'(y) = \min_{l=1}^{M}\{\sup_{x\in U} \min[\mu A'(x), \max(1 - \min_{i=1}^{n}(\mu A_i^l(xi)), \mu B^l(y))]\} \tag{2.34}$$

Algunos de los difusificadores que existen son: Singleton, Gaussian y Triangular.

- *Singleton:*

$$\mu A'(x) = \begin{cases} 1 & si \quad x = x^* \\ 0 & \sin o \end{cases} \tag{2.35}$$

- *Gaussiano:*

$$\mu A'(x) = e^{-\left(\frac{x_1 - x_1^*}{a_1}\right)^2} * \cdots * e^{-\left(\frac{x_n - x_n^*}{a_n}\right)^2} \tag{2.36}$$

- *Triangular:*

$$\mu A'(x) = \begin{cases} \left(1 - \frac{|x_1 - x_1^*|}{b_1}\right) * \cdots * \left(1 - \frac{|x_n - x_n^*|}{b_n}\right) & si \quad |x_i - x_i^*| \le b_i, \quad i = 1,2,...,n \\ 0 & \sin o \end{cases} \tag{2.37}$$

Los difusificadores singleton simplifican los cálculos de máquina de inferencia para cualquier tipo de función de membresía de las reglas difusas. Los difusificadores Gaussianos y Triangulares simplifican los cálculos si las MF´s de las reglas difusas son Gaussianas o triangulares. Los difusificadores Gaussianos y Triangulares eliminan el ruido en la entrada.

Conceptualmente la función de los desdifusificadores es especificar un punto en V que represente mejor al conjunto difuso B´. Para elegir un método de desdifusificación se consideran los siguientes aspectos: que y* represente a B´, simplicidad de cálculos y continuidad, es decir, que un pequeño cambio en B´ no resulte un gran cambio en y*. Algunos tipos de desdifusificadores son:

- *Centro de gravedad:*

$$y^* = \frac{\int_V y \mu B'(y) dy}{\int_V \mu B'(y) dy} \tag{2.38}$$

- *Centro promedio:*

$$y^* = \frac{\sum_{l=i}^{M} y^{-1} w_l}{\sum_{l=1}^{M} w_l} \tag{2.39}$$

- *Máximo:*

$$hgt(B') = \left\{ y \in V \,\middle|\, \mu B'(y) = \sup_{y \in V} \mu B'(y) \right\} \tag{2.40}$$

Donde *hgt(B)* es el conjunto de todos los puntos en *V* en donde $\mu B'$ alcanza su valor máximo. El desdifusificador que reúne mejores características es el centro promedio.

2.5. Redes de Petri.

Las Redes de Petri son una herramienta matemática de propósito general. Las Redes de Petri tienen dos características interesantes:

- Hacen posible modelar y visualizar tipos de comportamiento teniendo paralelismo, concurrencia, sincronización y recursos compartidos.

- Los resultados teóricos son abundantes, las propiedades de estas redes han sido y siguen siendo estudiadas extensivamente.

Los principales usuarios de las Redes de Petri son los científicos en computación y en control automático. Sin embargo la herramienta es lo suficientemente general, para modelar una gran variedad de tipos de fenómenos.

Las Redes de Petri tienen dos tipos de nodos, denominadas lugares y transiciones. Un lugar es representado por un círculo y una transición por una barra (autores recientes lo representan por una caja). Los lugares y las transiciones son conectados por arcos. El número de lugares es finito y distinto de cero. El número de transiciones también es finito y distinto de cero. Un arco es conectado directamente entre un lugar y una transición o una transición a un lugar. Los lugares se representan por una "P", y las transiciones por una "T".

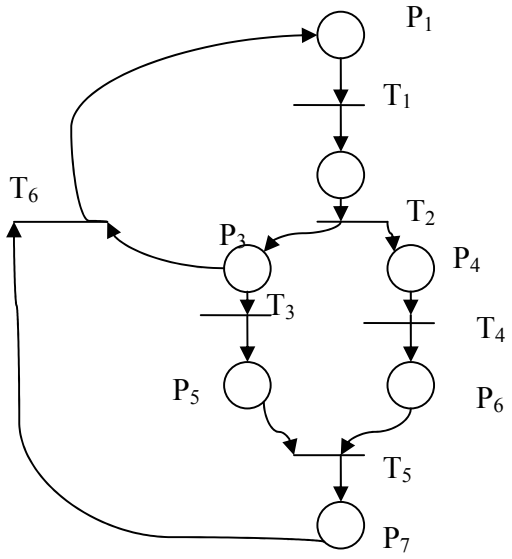


Figura 2-9: Red de Petri sin marcaje.

En la figura 2-9 se representa una Red de Petri con 7 lugares, 6 transiciones y 15 arcos. Se tiene el conjunto de lugares P={P_1, P_2, P_3, P_4, P_5, P_6, P_7} y el conjunto de transiciones T={T_1, T_2, T_3, T_4, T_5, T_6}. El lugar P_3 es la entrada de la transición T_3, y el lugar P_5 es su salida. Una transición sin lugar en la entrada se le denomina "transición fuente", y una transición sin lugar en la salida se le llama "transición sumidero".

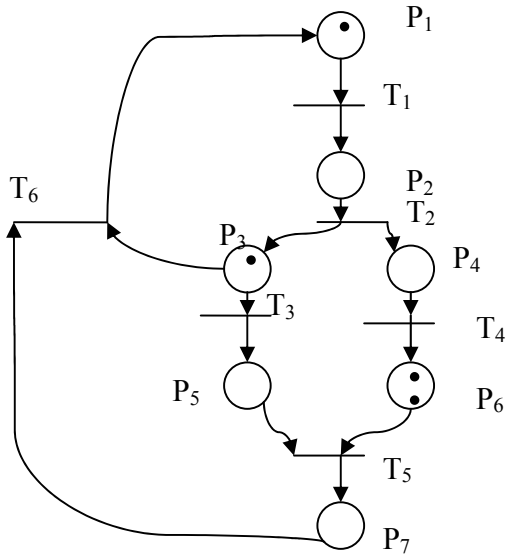


Figura 2-10: Red de Petri con marcaje.

En la figura 2-10 se presenta una Red de Petri con marcaje. Las marcas se representan con la letra "M". $m_1=m_3=1$, $m_6=2$, y $m_2=m_4=m_5=m_7=0$. El marcaje de la Red esta definido por $M=(m_1,m_2,m_3,m_4,m_5,m_6,m_7)$, por lo tanto $M=(1,0,1,0,0,2,0)$. El marcaje define el estado de una Red de Petri en un momento preciso. La evolución de los estados corresponde por lo tanto a una evolución del marcaje, causada por la activación de las transiciones. Una transición sólo puede ser activada si en el lugar de entrada existe por lo menos una marca. Una transición fuente siempre está activada.

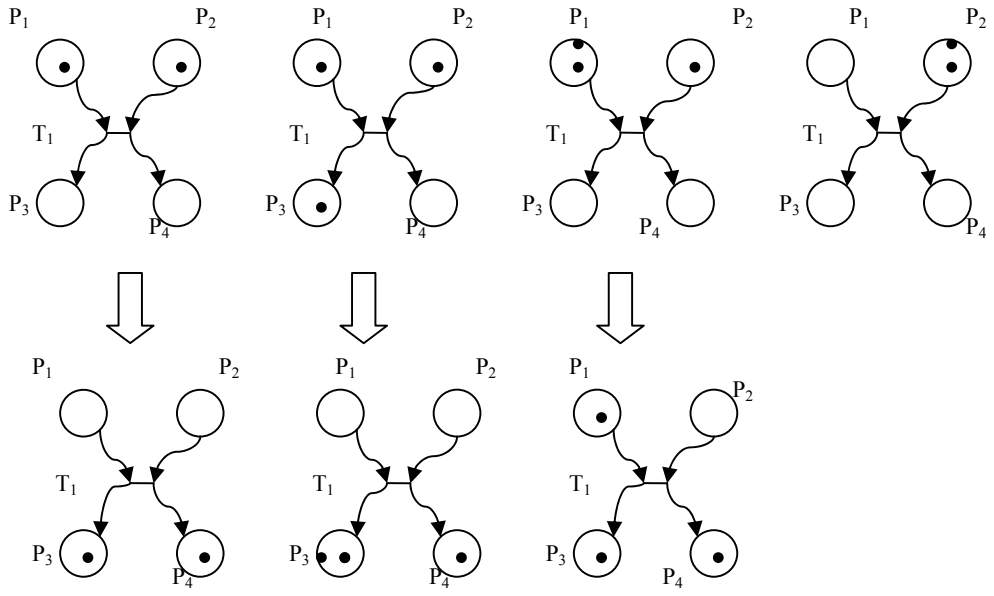


Figura 2-11: Ejemplos de activación de transiciones.

Las Redes de Petri pueden ser autónomas o no autónomas. Las no autónomas requieren de un evento externo para que se active la transición, y son sincronizadas y/o temporizadas.

Estructuras particulares de las Redes de Petri:

- *Estado gráfico:* ocurre si y sólo si cada transición tiene exactamente una entrada y una salida. Pueden ocurrir conflictos, pero no sincronización.
- *Evento gráfico:* ocurre si y sólo si cada lugar tiene exactamente una entrada y una salida. Puede estar sincronizado, pero no puede haber conflictos.
- *Libre de conflicto:* es cuando cada lugar tiene como máximo una salida.
- *De elección libre:* es cuando ocurre un conflicto, es decir, cuando un lugar es entrada de dos o más transiciones, y todas esas transiciones tienen una sola entrada. Es de elección libre extendida cuando todas las transiciones en conflicto tienen el mismo número de entradas, es decir, si hay dos transiciones en conflicto ambas transiciones tienen dos entradas.
- *Simple:* Ocurre cuando cada transición es afectada como máximo por un conflicto.
- *Puro:* Un par de lugares P_i y una transición T_j se denominan "lazo seguro" si P_i es entrada y salida de T_j. Una Red de Petri es pura cuando no tiene lazo seguro.

Las Redes de Petri generalizadas, son aquellas en las que los arcos están asociados con un peso que tiene que ser un número entero positivo.

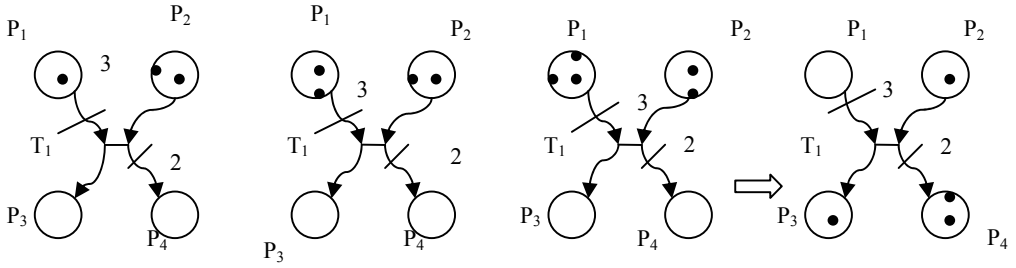


Figura 2-12: Arcos asociados con un peso.

Cuando no se marca el valor, el peso es igual a 1.

La capacidad finita de las Redes de Petri es una de las características que son asociadas con los lugares.

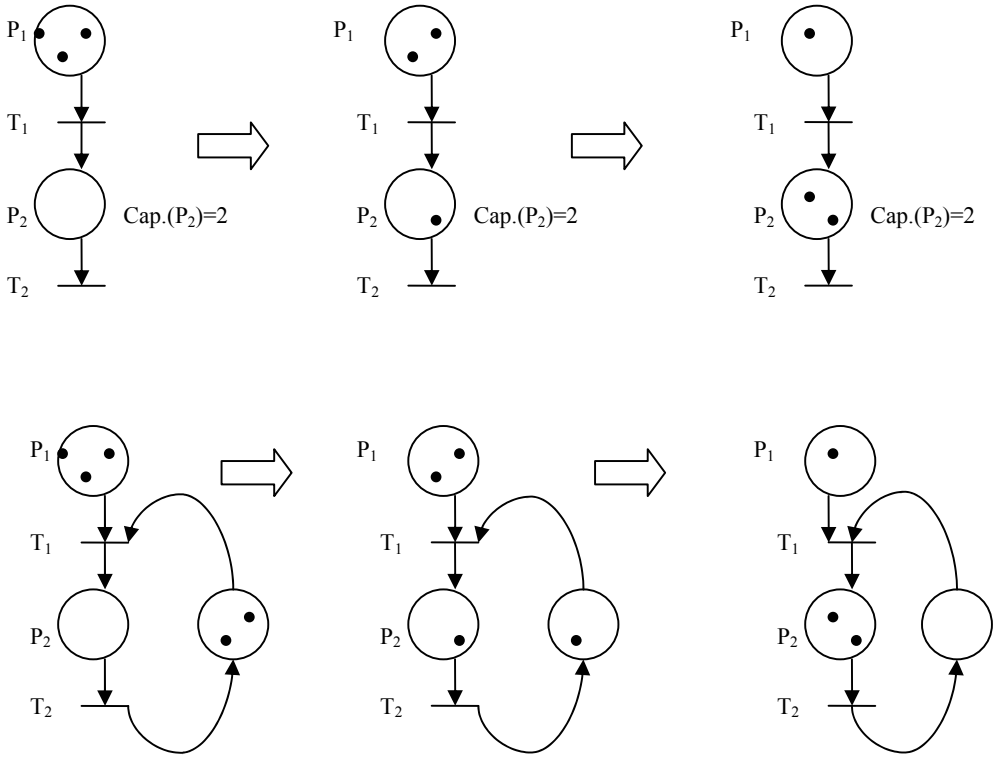


Figura 2-13: Capacidad finita de las Redes de Petri.

Los arcos inhibidores son arcos directamente conectados del lugar Pi a la transición Tj. Al final del arco se introduce un círculo. El inhibidor de arco entre Pi y Tj significa que la transición Tj sólo se activará si el lugar Pi no contiene marcas. La activación se realizará tomando una marca de todas las entradas a Tj excepto de Pi.

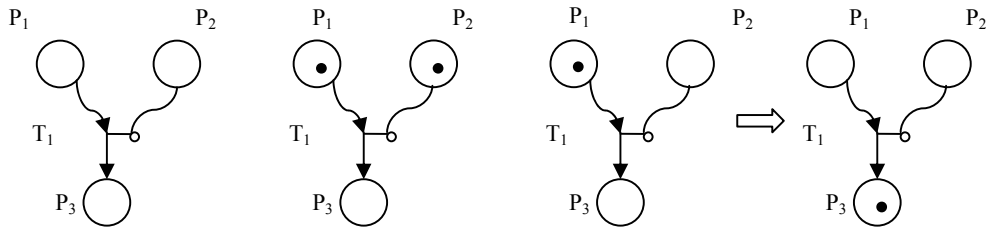


Figura 2-14: Arcos inhibidores.

El marcaje de las Redes de Petri en un momento dado es una columna vector quien está compuesta por el número de marcas en el lugar P_i en ese momento.

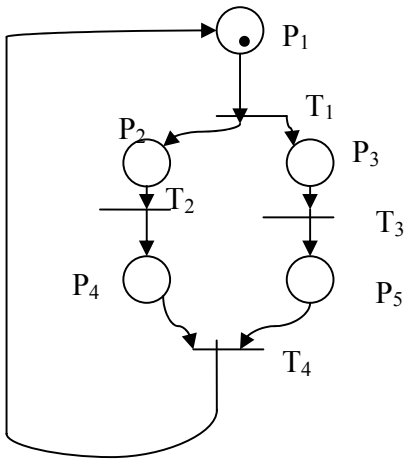


Figura 2-15: Marcaje de una Red de Petri.

Para la figura 2-15 se tiene:

$$M_0 = (1,0,0,0,0) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}^T = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$M_1 = (0,1,1,0,0)$$

$$M_2 = (0,0,1,1,0)$$

$$M_3 = (0,1,0,0,1)$$

Se dice que un lugar P_i es limitado por un marcaje inicial M_0 si el número de marcas no aumenta en el transcurso de los eventos. Las Redes no limitadas son las que si aumenta su número de marcas.

Se dice que una Red de Petri es segura o binaria para un marcaje M_0 si para todos los demás marcajes, cada lugar contiene al menos una marca.

Las Redes de Petri en las cuales las transiciones no se desactivan se les denominas "vivas". Algunas Redes de Petri llegan a un punto muerto, esto sucede cuando no se pueden activar más transiciones. El concepto de conflicto ocurre cuando un lugar es entrada de dos o más transiciones. Puede existir el conflicto efectivo o persistente y el conflicto no efectivo. El efectivo es cuando el número de marcas de un lugar es menor al número de salidas de ese lugar. El no efectivo es cuando el número de marcas en un lugar es mayor o igual a su número de salidas.
La activación múltiple ocurre cuando dos o más transiciones son activadas en un solo paso o simultáneamente.

Al empezar de un estado inicial de marcaje, el marcaje de la Red va evolucionando y si no tiene un punto muerto, el número de transiciones es ilimitado. Los sistemas invariantes permiten las propiedades certeras de los marcajes obtenidos y transiciones activadas para ser caracterizados.

El marcaje grafico es unir nodos que corresponden a los marcajes conseguidos y los arcos corresponden a la activación de las transiciones, obteniendo que pase de un marcaje a otro.

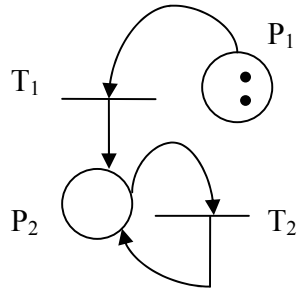
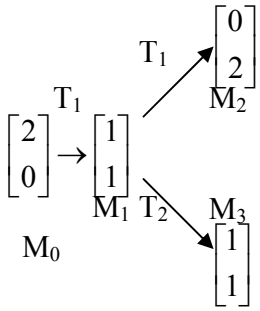


Figura 2-16: Marcaje gráfico.



La cobertura de árbol contiene un número finito de nodos. Una cobertura gráfica de los marcajes conseguidos, se obtiene uniendo juntos los nodos de cobertura de árbol que correspondan al mismo marcaje.

Una de las características de las Redes de Petri es su capacidad de representar gráficamente relaciones y visualización de conceptos, algunos de ellos son:

- *Paralelismo y concurrencia.*
- *Sincronización.*
- *Recurso compartido.*
- *Memorización.*
- *Lectura.*
- *Capacidad limitada.*

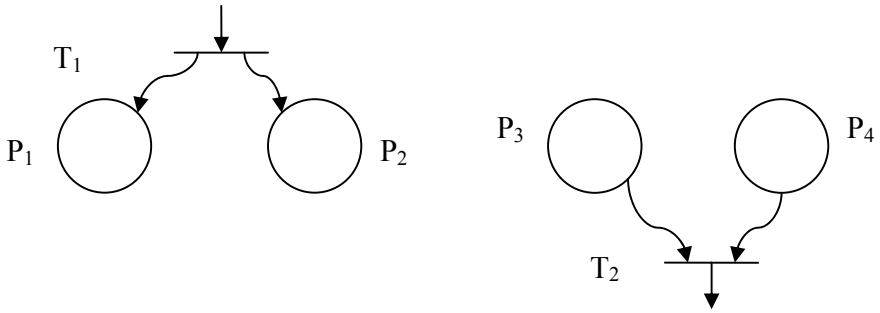


Figura 2-17: Paralelismo y concurrencia.

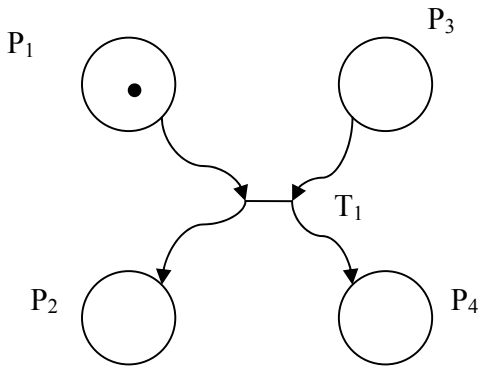


Figura 2-18: Sincronización.

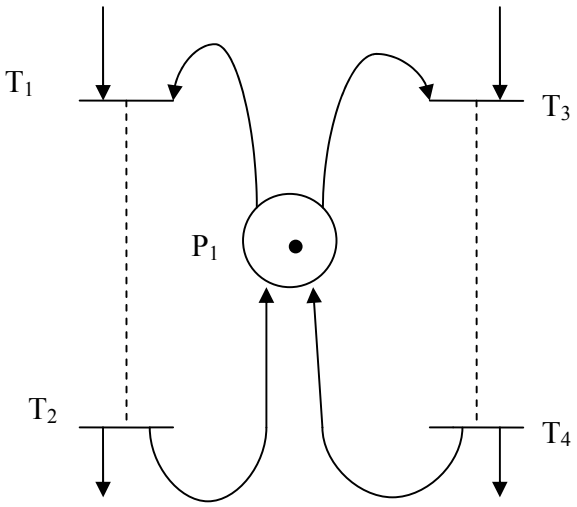


Figura 2-19: Recurso compartido.

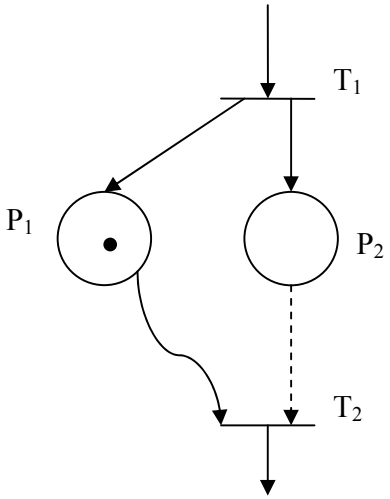


Figura 2-20: Memorización.

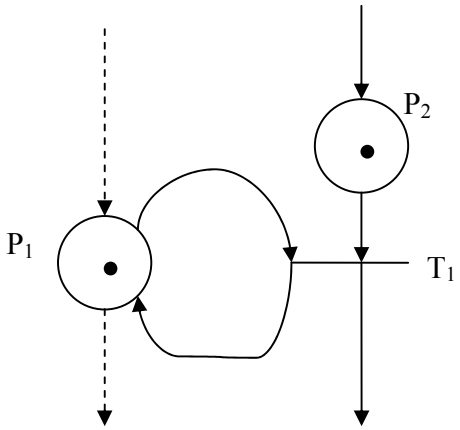


Figura 2-21: Lectura.

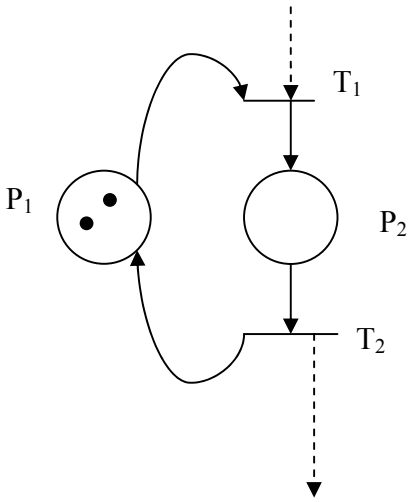


Figura 2-22: Capacidad limitada.

2.5.1. Redes de Petri no autónomas.

En las Redes de Petri no autónomas sabemos que transiciones pueden ser activadas, pero no sabemos cuando se activarán. En una Red de Petri sincronizada, un evento es asociado a cada transición, y la transición se realizará si la transición se puede activar, cuando el evento asociado ocurra. Una Red de Petri sincronizada es un triple <R, E, Sync> donde R es una Red con

marcaje, E es un conjunto de eventos externos, Sync es una función del conjunto T de las transiciones de R.

E= $\{E^1, E^2, \ldots\}$, es un conjunto de eventos externos. La notación E^i, (E super i) corresponde al nombre de evento externo número. La notación T_j (E sub j) corresponde al evento asociado con la transición T_j. Dos eventos externos nunca ocurren simultáneamente.

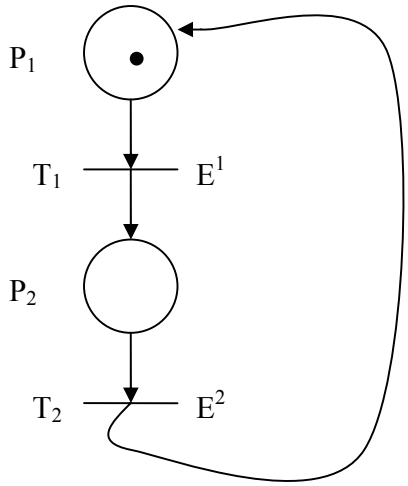


Figura 2-23: Red de Petri sincronizada.

El mismo evento puede ser asociado con varias transiciones de una Red sincronizada. Consideremos un nuevo evento quien no es un evento externo. Este siempre es el evento ocurrente, el cual se escribe como "e".

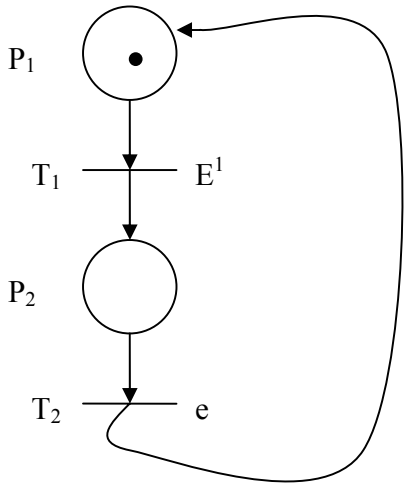


Figura 2-24: Evento ocurrente "e".

En la figura 2-24 el evento "e" está asociado con la transición T_2. Lo cual significa que en cuanto se pueda activar T_2, se activará inmediatamente. En el caso del ejemplo se dice que hay una ocurrencia iterativa en la activación del evento E^1, porque cada vez que se active el evento se activará la transición. Se dice que una Red es totalmente sincronizada si ninguna de sus transiciones es asociada con el elemento "e".

Redes de Petri temporizadas.

Una Red de Petri temporizada permite describir sistemas que son dependientes del tiempo. Existen dos métodos para modelar el tiempo: se puede asociarse el tiempo con los lugares (P-timed), o con las transiciones (T-timed).

- *P-timed.* Un tiempo d_i que puede valer cero, es asociado con cada lugar P_i. Podríamos considerar el caso en donde d_i es un valor constante, pero en general es una variable. Una P-timed PN es un par <R,Tempo> donde R es el marcaje de la Red y Tempo es una función del conjunto de lugares P con un conjunto de números racionales o cero. $(P_i) = d_i$ = temporización asociada con el lugar P. Las Redes temporizadas son definidas por lo regular por tiempos en números racionales, sin embargo es posible definir Redes de Petri temporizadas con números reales. Cuando una marca es depositada en el lugar P_i, esa marca debe quedarse en ese lugar por lo menos un tiempo d_i. Se dice que esa marca no está disponible en ese tiempo. Cuando el tiempo di transcurre, la marca se vuelve disponible.
- *T-timed.* Se puede mostrar que las Redes P-timed y T-timed son equivalentes. Es posible ir de un modelo al otro. Una Red de Petri T-timed es un par <R, Tempo> donde R es el

marcaje de la Red; y Tempo es una función del conjunto de transiciones T con conjunto de números racionales o cero. Tempo(T_j)=d_j=tiempo asociado con transiciones T_j. Una marca puede tener dos estados: éste puede estar reservado para la activación de la transición T_j o puede no estar reservado. En cualquier momento t, el marcaje presente M, es la suma de M^r y M^n, donde M^r es el marcaje de las marcas reservadas y M^n es el marcaje de las marcas no reservada.

Para una Red de Petri la velocidad de máximo funcionamiento puede ser definido como: tan pronto como se habilite una transición, las marcas requeridas para esta transición son reservadas. Y el funcionamiento de velocidad propio como: tan pronto como una marca sea depositada en un lugar, ésta es reservada para activación de una transición.

Capítulo 3

Controlador mediante Redes de Petri

3.1. Diseño del controlador mediante Redes de Petri.

Las Redes de Petri sirven para modelar una gran variedad de tipos de fenómenos. En esta sección se presentará la Red de Petri diseñada para modelar el funcionamiento del controlador implementado. El desarrollo del diseño del controlador se basó en la implementación de algoritmos computacionales tomando en cuenta el conocimiento del comportamiento de los procesos lineales y no lineales. El funcionamiento del controlador mediante Redes de Petri es similar al de controladores difusos en donde se aplican reglas *if-then*, y en donde se utiliza el conocimiento del proceso que se desea controlar. A diferencia de un FKBC y controladores PID en la implementación del controlador mediante Redes de Petri no se utilizó ningún modelo matemático específico, ya que lo que se deseaba era que no se tuviera que realizar ninguna modelación del proceso a controlar, y que se pudiera utilizar en una gran variedad de procesos. Por lo tanto el controlador mediante Redes de Petri se puede utilizar en procesos aproximados a un sistema de primer orden en un amplio rango de k, τ y θ. Aunque no es necesario que el usuario del controlador introduzca algún valor del proceso, en el momento de solicitar un cambio de referencia, el controlador hace un cálculo aproximado de θ, y una vez que el proceso se estabiliza después del cambio de referencia el controlador realiza un cálculo de su ganancia (k).

En el desarrollo del controlador después de implementar algún algoritmo computacional (regla *if-then*), se realizaban pruebas con diversos procesos, se revisaban los resultados obtenidos (sobretiro, tiempo de elevación y tiempo de establecimiento), y se verificaba si había mejorado el funcionamiento del controlador o no. Finalmente se dejaron de incorporar algoritmos computacionales hasta que el controlador desarrollado tuvo mejor desempeño (en varios procesos) que un controlador PID sintonizado para cada proceso. En la figura 3-1 se muestra un diagrama de flujo del desarrollo de la implementación del controlador mediante Redes de Petri.

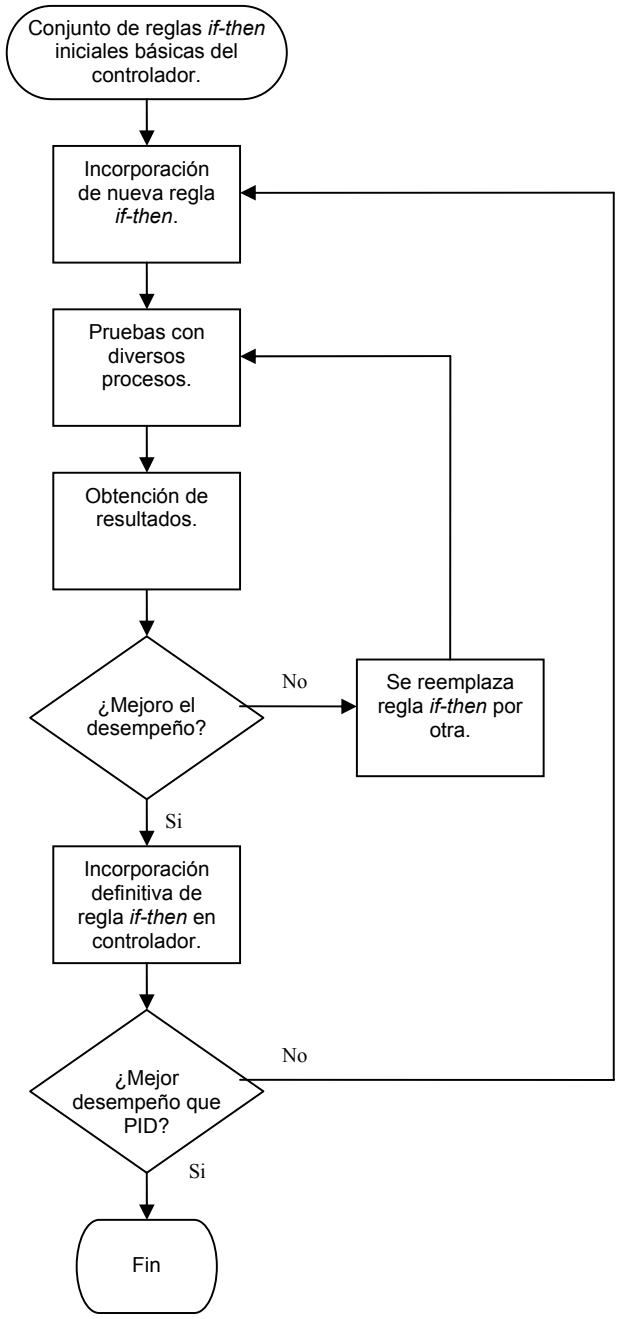


Figura 3-1: Desarrollo de la implementación del controlador mediante Redes de Petri.

Una vez que se lograron los resultados deseados por parte del controlador diseñado y ya no se incorporaron más reglas *if-then*, se implementó la Red de Petri que modela su comportamiento. En dicha Red de Petri los lugares representan las salidas del controlador (diversas manipulaciones, o simplemente auxiliares para conectar dos transiciones o almacenar marcas), y cada transición representa las entradas del controlador (diversas condiciones relacionadas con el comportamiento del proceso). Para que un lugar realice la operación que tiene asignada debe de tener por lo menos un token, y para que una transición se habilite se debe cumplir el evento externo asignado.

En figura 3-3 se muestra la Red de Petri del controlador. En esta Red el paso del token por los distintos lugares muestra que acción de control se está llevando acabo, ya que casi todos los lugares generan una manipulación correctiva cada vez que cuentan con una o más marcas. Los lugares que están asociados con una manipulación se señalan en la red con las letras "*GM*" (genera manipulación). Debido al diseño de la Red, los lugares que generan una manipulación únicamente podrán contar con una marca como máximo. El único lugar que puede tener más de una marca es el 3, que sirve para mostrar el tiempo muerto del proceso. Para elegir que ruta seguirán los tokens, se utilizan transiciones ligadas a un evento externo el cual depende del comportamiento del proceso, un ejemplo de un evento externo para está red podría ser: el evento E^r se cumple si el error es mayor que 10. Algunas transiciones no están asociadas a eventos externos (e), lo cual significa que en cuanto se habiliten se activará la transición. Las variables del proceso que se monitorean para saber cuando se cumple cada evento externo, son: referencia, manipulación y variable controlada. La información se almacena en vectores de tamaño 100, es decir, se cuenta con el valor actual y con 99 valores pasados.

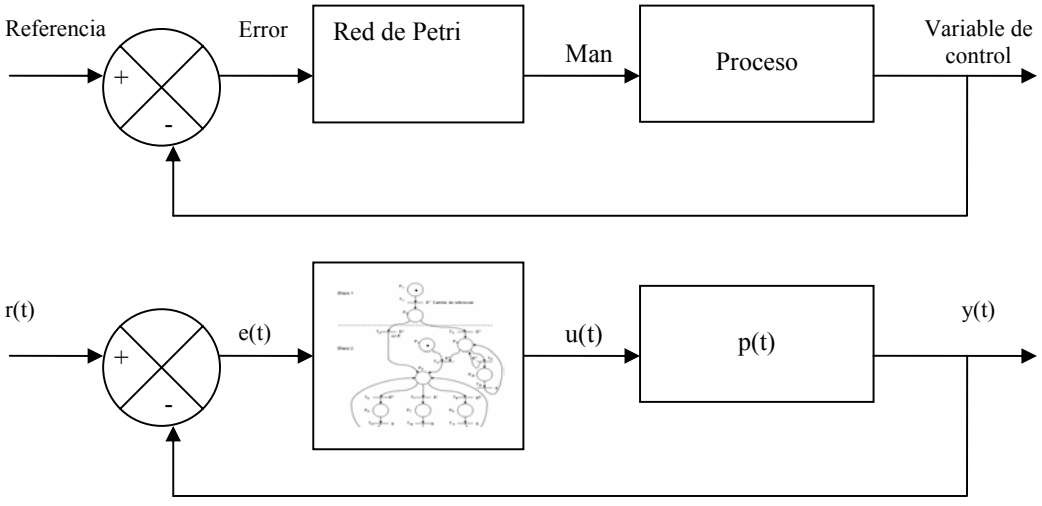
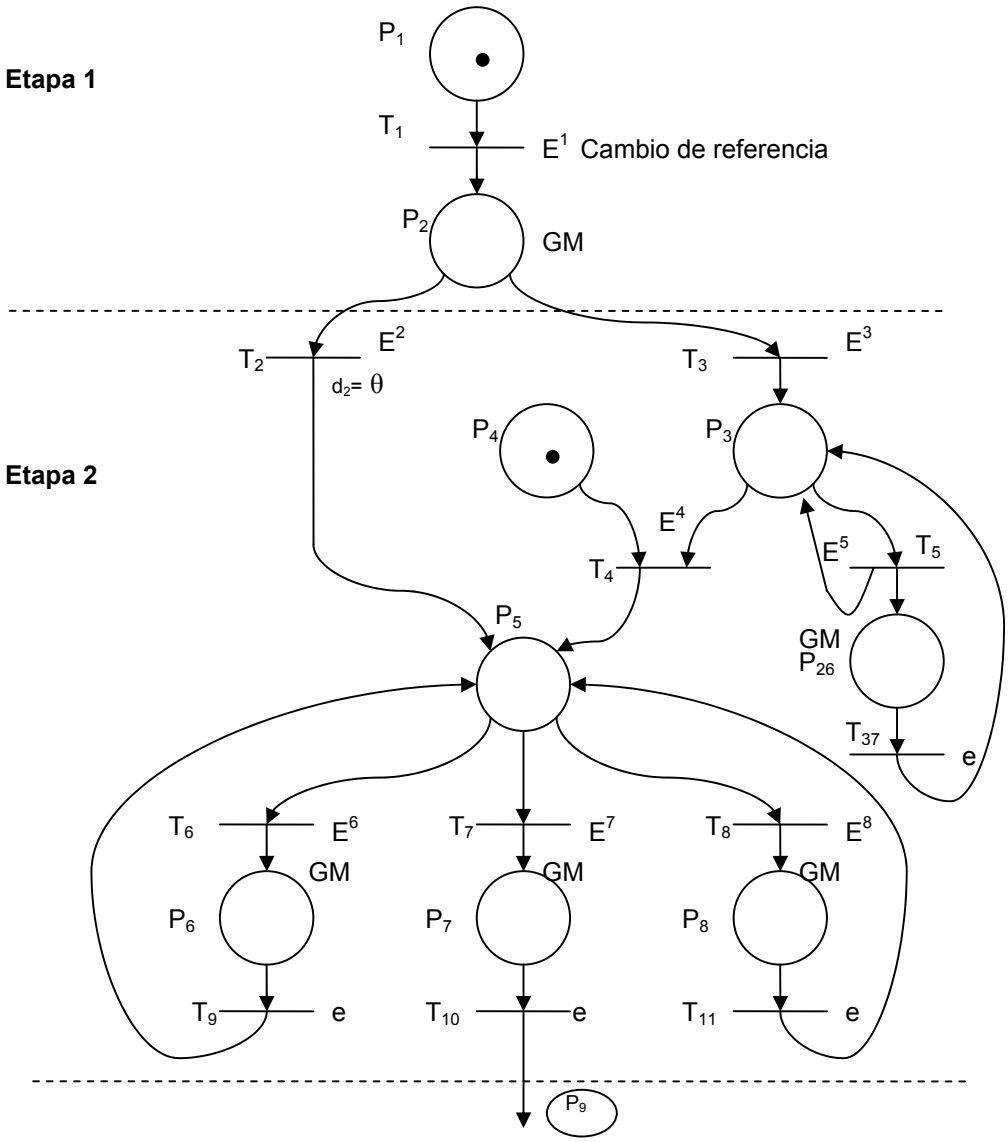
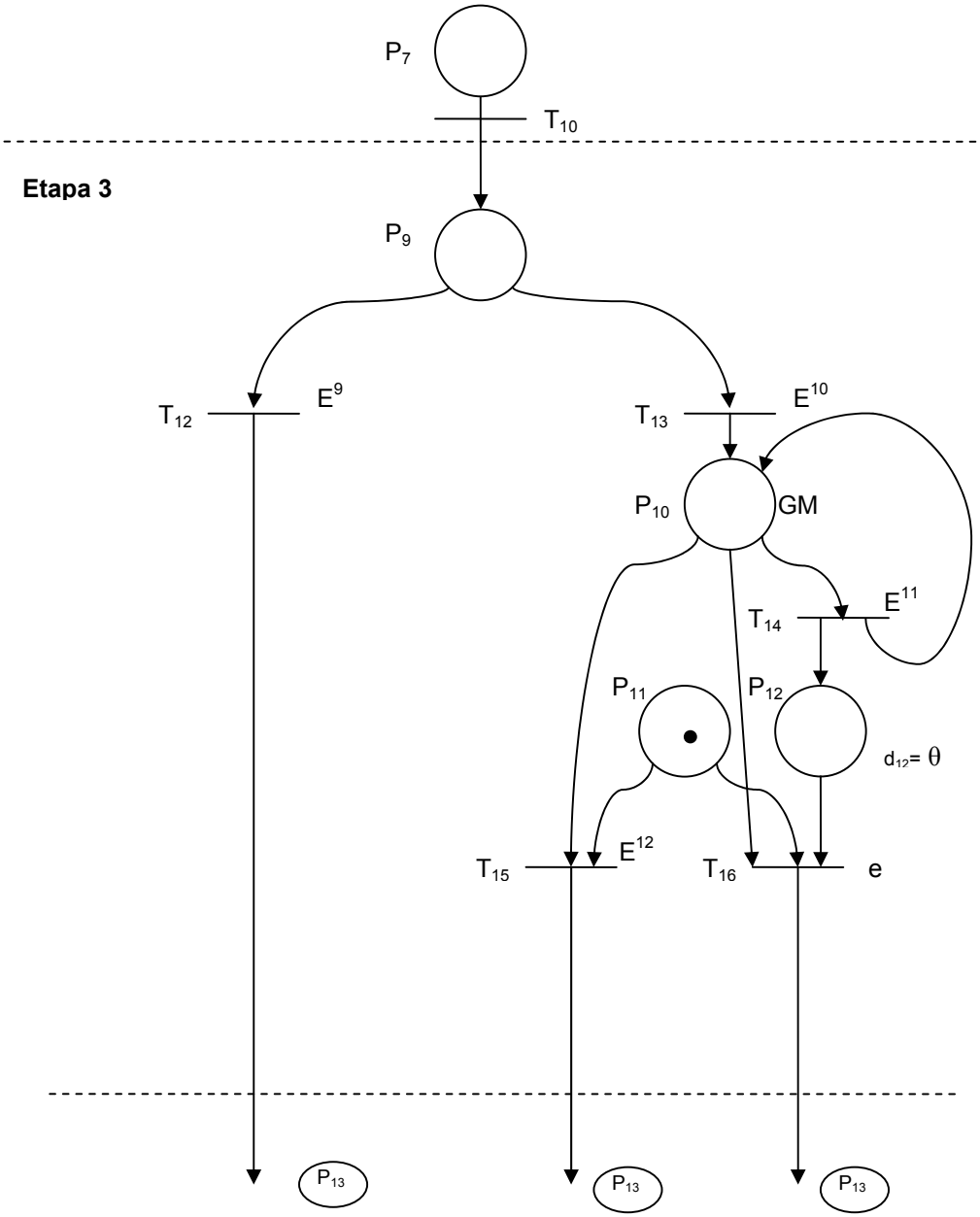


Figura 3-2: Diagrama de bloques del controlador de Redes de Petri.


Etapa 1
P1
T1
E1 Cambio de referencia
P2
GM
T2
E2
d2= θ
T3
E3
Etapa 2
P4
P3
E4
E5
T5
T4
P5
GM
P26
T37
e
T6
E6
T7
E7
T8
E8
GM
GM
GM
P6
P7
P8
T9
e
T10
e
T11
e
P9


P7
T10
Etapa 3
P9
T12
E9
T13
E10
P10
GM
E11
T14
P11
P12
d12= θ
T15
E12
T16
e
P13
P13
P13

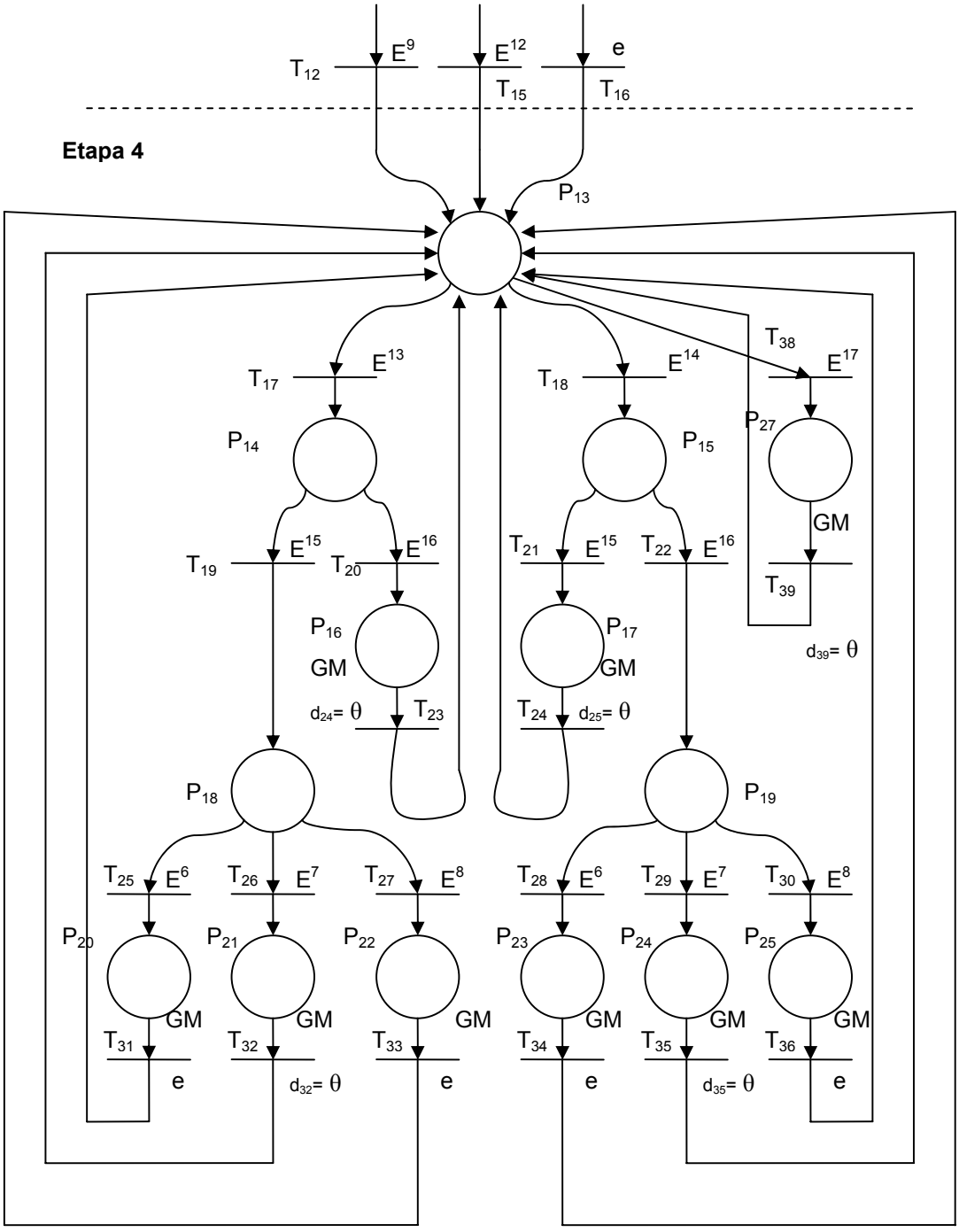


Figura 3-3: Red de Petri del controlador.

Cada vez que se realice un cambio de referencia, la Red se inicializará con el marcaje de la figura 3-3. A continuación se describirá cada uno de los eventos externos a los que están ligadas las transiciones, y la función de cada lugar.

- E^1 se cumple cuando existe un cambio de referencia.
- E^2 se cumple si ya se calculó previamente el tiempo muerto {θ}, el tiempo muerto se calcula cada vez que en modo simulación se cambia el proceso, o cada vez que se inicializa el programa.
- E^3 se cumple sino se ha calculado el tiempo muerto.
- E^4 se cumple si $(\Delta variable\ controlada\ [0] \geq 0.3$ *ó si* $\Delta variable\ controlada\ [0] \leq -0.3$, y
- E^5 se cumple si $(-0.3 < \Delta variable\ controlada\ [0] < 0.3)$ *y que el lugar 4 tenga al menos un token.*
- E^6 se cumple si $(1+\theta) < (Error[0] / \Delta variable\ controlada\ [0]) \leq (3+\theta)$.
- E^7 se cumple si $(Error[0] / \Delta variable\ controlada\ [0]) \leq (1+\theta)$.
- E^8 se cumple si $(3+\theta) < (Error[0] / \Delta variable\ controlada\ [0])$.
- E^9 se cumple si $\theta = 0$.
- E^{10} se cumple si $\theta \neq 0$.
- E^{11} se cumple si ($\Delta variable\ controlada\ es > 0$ *y el* $Error[0] < 0$, *ó si* $\Delta variable\ controlada\ es < 0$ *y el* $Error[0] > 0$) y que no esté habilitada la transición 16.
- E^{12} se cumple sino se cumple el evento eterno E^{11}.
- E^{13} se cumple si $\Delta variable\ controlada\ es > 0$.
- E^{14} se cumple si $\Delta variable\ controlada\ es < 0$.
- E^{15} se cumple si $Error[0] \geq 0$.
- E^{16} se cumple si $Error[0] < 0$.
- E^{17} se cumple si $\Delta variable\ controlada\ es = 0$.
- T_2, T_{23}, T_{24}, T_{32}, T_{35} *y* T_{39} están asociadas a un *tiempo* = θ, lo cual significa que cuando se activen dichas transiciones el token tardará θ en pasar al lugar 13.
- P_1 sirve para almacenar el token de inicio.
- P_2 genera una *manipulación* $[0]$ = *manipulación* $[-1] + \Delta referencia$.
- P_3 sirve para calcular el tiempo muerto. El número de tokens que contiene es el tiempo muerto del proceso. Debido a que el marcaje es un número entero, si θ es un número fraccionario éste se aproximará al número entero siguiente.
- P_4 sirve para limitar el paso a un solo token a las siguientes etapas.
- P_5 no genera ninguna manipulación, sirve para conectar la salida de las transiciones 2, 4, 9 y 11 con la entrada de las transiciones 6, 7 y 8.

- P_6 genera una *manipulación [0] = manipulación [-1].*
- P_7 genera *una manipulación [0] = manipulación antes de realizar el cambio de referencia + (Δreferencia/Ganancia del proceso {K_p}).* Se inicializa con una K_p determinada y cada vez que se realiza un cambio de referencia se calcula dicho valor. La ecuación para calcular K_p es: K_p = *(Variable controlada en estado estable después de realizar un cambio de referencia – variable controlada antes de realizar el cambio de referencia)/(manipulación a la que se tiene la variable controlada en estado estable después de realizar un cambio de referencia – manipulación que se tenía antes de realizar el cambio de referencia).*
- P_8 genera una *manipulación = manipulación [-1]+ 0.03*(Δmanipulacion[-1-θ]/Δvariable controlada[0])*ErrorC[0].* Donde *ErrorC[0]= Error[0]+0.75*(ΔError[0]*θ). ErrorC[0]* es el error aproximado que se tendrá después de que transcurra el tiempo muerto, lo cual sirve para generar manipulaciones tomando en cuanto que dicha manipulación afectará al proceso después de que transcurra el tiempo muerto, y en ese momento el error ya no será el mismo. *(Δvariable controlada[0]/ Δmanipulacion[-1-θ])* nos indica cuanto cambia la variable controlada ante un cambio de manipulación. Si la manipulación anterior fue cero *manipulación = manipulación [-1] + Error[0].*
- P_9 sirve para conectar la salida de la transición 10 con la entrada de las transiciones 12 y 13.
- P_{10} genera una *manipulación [0] = manipulación [-1] + (Error[0])/(15*K_p).*
- P_{11} sirve para limitar el paso a un solo token a las siguientes etapas, y para que únicamente se active la transición 15 ó 16, no ambas.
- P_{12} a este lugar esta asociado a un *tiempo = θ*, lo cual significa que cuando dicho lugar reciba un token, la transición 16 no estará habilitada hasta que transcurra θ.
- P_{13} sirve para conectar la salida de las transiciones 23, 24, 31, 32, 33, 34, 35, 36 y 39 con la entrada de las transiciones 17, 18 y 38.
- P_{14} sirve para conectar la salida de la transición 17 con la entrada de las transiciones 19 y 20.
- P_{15} sirve para conectar la salida de la transición 18 con la entrada de las transiciones 21 y 22.
- P_{16} y P_{17} generan una *manipulación [0] = manipulación [-1]+(2*ErrorC[0]/K_p).*
- P_{18} sirve para conectar la salida de la transición 19 con la entrada de las transiciones 25, 26 y 27.
- P_{19} sirve para conectar la salida de la transición 22 con la entrada de las transiciones 28, 29 y 30.
- P_{20} genera una *manipulación [0] = manipulación[-1]-((Δvariable controlada C* ErrorC[0])/((2+ θ)).* Donde *Δvariable controlada C= Δvariable controlada[0] + (0.75*(Δvariable controlada[0]- Δvariable controlada[-1])*θ). Δvariable controlada C* sirve para

generar manipulaciones tomando en cuanto que dicha manipulación afectará al proceso después de que transcurra el tiempo muerto, y en ese momento *Δvariable controlada* ya no será la misma.

- P_{21} genera una *manipulación [0] = manipulación[-1]-(Δvariable controlada C* ErrorC[0]).*
- P_{22} y P_{25} generan una *manipulación = manipulación [-1]+ ((0.025*(Δmanipulacion[-1-θ]/Δvariable controlada[0])*ErrorC[0])/(1+θ))*. El valor de esta manipulación está a limitada a valor absoluto máximo menor o igual a valor absoluto del error actual por dos.
- P_{23} genera una *manipulación [0] = manipulación[-1]+ ((Δvariable controlada C* ErrorC[0])/((2+ θ))*.
- P_{24} genera una *manipulación [0] = manipulación [-1]+(Δvariable controlada C* ErrorC[0]).*
- P_{26} genera una *manipulación [0] = manipulación [-1].*
- P_{27} genera una *manipulación [0] = manipulación [-1] + Error[0].*

3.2. Etapas de control.

El comportamiento del controlador se puede dividir en 4 etapas, las cuales son consecutivas y no existe retorno hasta que no se haga un nuevo cambio de referencia, es decir, cuando se hace un cambio de referencia se comienza con la etapa 1 y el proceso de control se va realizando por el paso del token por las siguientes etapas. Una vez que se llega a la etapa 4 y el proceso ya se encuentra en estado estable de nuevo, las acciones de control ante perturbaciones se realizan en esta etapa y no hay forma de regresar a la etapa 2 ó 3, únicamente se puede regresar a la etapa 1 si es que se realiza otro cambio de referencia.

3.2.1. Etapa 1.

La etapa uno del controlador consta de dos lugares y una transición. El lugar uno contiene un token el cual pasará al lugar dos cuando se active la transición uno (cuando exista un cambio de referencia), una vez que se encuentre el token en el lugar dos, el controlador enviará una *manipulación [0]= manipulación [-1] + Δreferencia.* Se eligió dicha manipulación, ya que en ese instante no se tiene todavía información del comportamiento del proceso si es que se acaba de iniciar el programa, y únicamente se cuenta con *Δreferencia.* Esta etapa del controlador finaliza cuando se activa la transición 2 ó 3.

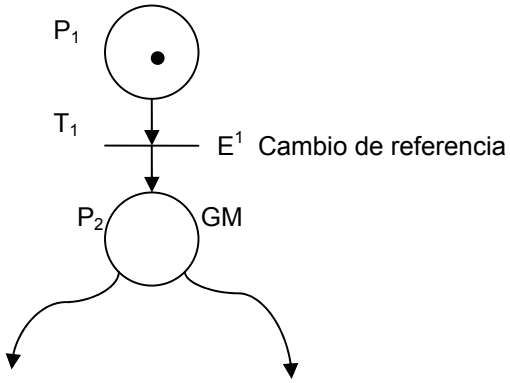


Figura 3-4: Etapa 1.

Cuando se acaba de iniciar el programa de control o se ha modificado el proceso en modo simulación, se recomienda que el primer cambio de referencia no sea muy pequeño, ya que a partir de la manipulación inicial, que es directamente proporcional al cambio de referencia, el controlador realiza un cálculo de tiempo muerto {θ} del proceso y si la manipulación inicial es muy pequeña tal vez *Δvariable controlada* no sea mayor a 0.3 ó menor a -0.3, que es el rango que se tiene para ruido o perturbaciones iniciales. Una vez que se realiza el primer cambio de referencia, el controlador ha calculado θ del proceso, y por tanto ya se podrán realizar cambios de referencia de la magnitud que se desee. Cuando se tiene el controlador en tiempo real, el cálculo de θ se realiza únicamente cuando se inicia el programa, por lo tanto si el proceso a controlar cambia, el controlador seguirá teniendo un buen desempeño siempre y cuando el nuevo proceso tenga el mismo θ.

3.2.2. Etapa 2.

La etapa 2 cuenta con 7 lugares y 11 transiciones. Las transiciones 2 y 3 sirven para decidir que hacer si ya se calculo θ o no previamente. La transición 2 tiene asociada un tiempo = θ, lo cual significa que si el evento 2 se cumple el token pasará al lugar 5 hasta que transcurra θ. Si es la primera vez que se calcula θ, las transiciones 4 y 5 sirven para decidir que hacer si ya transcurrió el tiempo muerto del proceso o no ante la primera manipulación. Para no confundir la respuesta del proceso a una manipulación, con ruido o perturbaciones se utiliza una banda de *variable controlada* ± *0.3*. El lugar 26 sirve para mantener constante la manipulación hasta que el proceso responda a la primera manipulación. Una vez que se active la transición 4, el lugar 3 tendrá un número de marcas que representa el valor aproximado de θ.

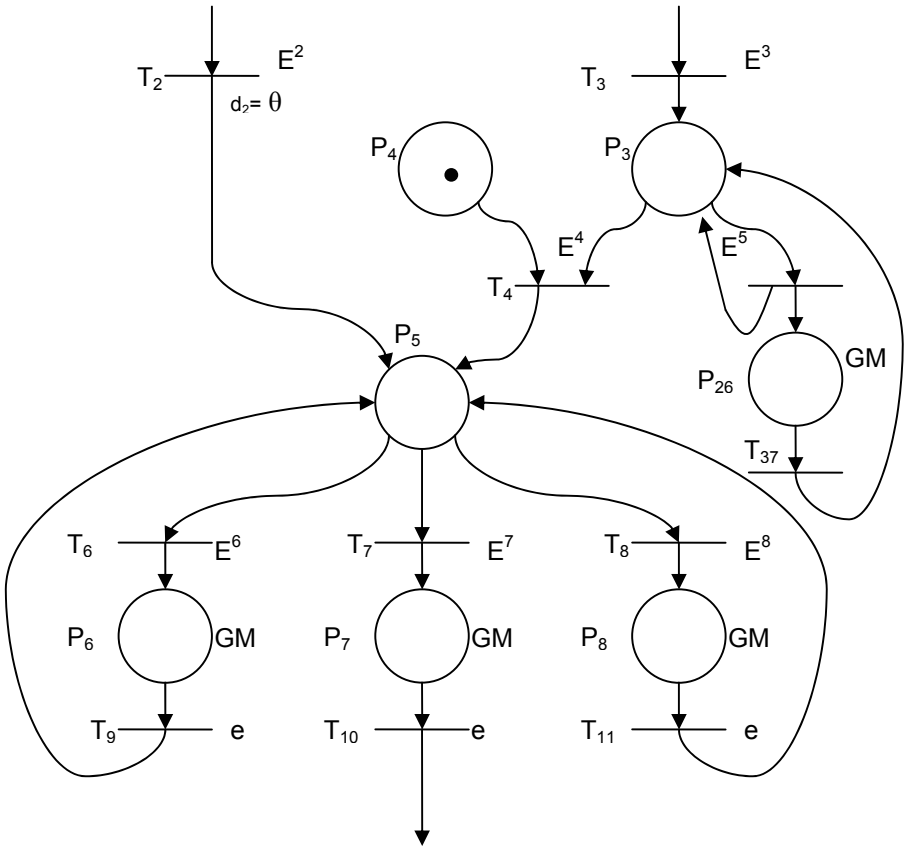


Figura 3-5: Etapa 2.

Un token llegará al lugar 5 cuando ya ha transcurrido θ después de que se realizó la manipulación inicial. Las transiciones 6, 7 y 8 sirven para elegir que Δmanipulación enviará el controlador. Para decidir se calcula en cuantos muestreos más la variable controlada será igual a la referencia, si *Δvariable controlada* continua constante. La ecuación para calcular el número de muestreos {*NM*} es: *Error[0]/ Δvariable controlada [0]*. Si *NM* es grande y *Δvariable controlada* es positivo, se necesita incrementar la manipulación para reducir el tiempo de elevación $\{t_r\}$. Si *NM* es grande y *Δvariable controlada* es negativo, se necesita disminuir la manipulación para reducir el tiempo de elevación $\{t_r\}$. Si *NM* es pequeño y *Δvariable controlada* es positivo, se necesita disminuir la manipulación para reducir el sobretiro $\{M_p\}$. Si *NM* es pequeño y *Δvariable controlada* es negativo, se necesita aumentar la manipulación para reducir el sobretiro $\{M_p\}$. Basados en las reglas *if-then* citadas, las transiciones 6, 7 y 8 se asocian con los eventos externos siguientes:

- E^6 se cumple si $(1+\theta)<NM \leq (3+\theta)$.
- E^7 se cumple si $NM \leq (1+\theta)$.
- E^8 se cumple si $(3+\theta)<(NM)$.

Y los lugares , 6, 7 y 8 se asocian con las siguientes manipulaciones:

- P_6 genera una *manipulación [0] = manipulación [-1]*.
- P_7 genera *una manipulación [0] = manipulación antes de realizar el cambio de referencia + (Δreferencia/Ganancia del proceso* $\{K_p\}$*)*. Si K_p es el valor correcto, la manipulación que se obtiene será muy similar a la que se obtendrá una vez que se estabilice la variable controlada.
- P_8 genera una *manipulación = manipulación [-1]+ 0.03*(Δmanipulacion[-1-θ]/Δvariable controlada[0])*ErrorC[0]*. Si la manipulación anterior fue cero, *manipulación = manipulación [-1] + Error[0]*.

El evento 6 representa cuando todavía falta para que la *Δvariable controlada* sea igual a la referencia. El evento 7 representa cuando falta muy poco para que la *Δvariable controlada* sea igual a la referencia. El evento 8 representa cuando falta mucho para que la *Δvariable controlada* sea igual a la referencia. El lugar 6 mantiene la manipulación constante; el lugar 7 reduce la manipulación si *Δvariable controlada* es positivo, y la aumenta si *Δvariable controlada* es negativo; y el lugar 8 aumenta la manipulación si *Δvariable controlada* es positivo, y la reduce si *Δvariable controlada* es negativo.

Para diseñar las ecuaciones que generan *Δmanipulación*; primero se revisó las variables que se tenían monitoreadas o se pudieran calcular, se eligieron las que están más relacionadas con el comportamiento del proceso; se verificó para cada variable si afecta directamente o inversamente proporcional a la manipulación necesaria para reducir el error, si afecta directamente se coloca multiplicando a las demás variables, y si afecta inversamente proporcional se coloca en la ecuación dividiendo a las demás variables; se realizaron pruebas; y finalmente se eligieron las ecuaciones con las que se tenía un mejor desempeño del controlador.

Las transiciones 9, 10, 11 y 37 no están ligadas a ningún evento externo, por lo tanto en cuanto se habilitan se activa la transición. Mientras el token pase por los lugares 6 y 8 se continuará en la etapa 2 del controlador, únicamente se avanzará a la etapa 3 cuando el token pase por el lugar 7, lo cual se logrará cuando falten dos o menos tiempos de muestreo más para que la variable controlada sea igual a la referencia, si se continua con el mismo *Δvariable controlada*.

3.2.3. Etapa 3.

La etapa 3 cuenta con 4 lugares y 5 transiciones. Esta etapa sirve únicamente para procesos con $\theta \neq 0$, ya que si $\theta=0$ el evento 9 se cumple y por lo tanto se avanza a la cuarta etapa. Las transiciones 9 y 10 sirven para elegir que hacer si el proceso tiene θ o no. Esta etapa fue una de las adecuaciones que se hicieron en el controlador para tener un mejor desempeño cuando el proceso tiene tiempo muerto. Otros de los elementos del controlador que están directamente relacionados con θ, es:

- El cálculo del error aproximado que se tendrá después de que transcurra el tiempo muerto. *ErrorC[0]= Error[0]+0.75*(ΔError[0]*θ).*
- El cálculo de *Δvariable controlada* aproximado después de que transcurra el tiempo muerto. *Δvariable controlada C = Δvariable controlada[0] + (0.75*(Δvariable controlada[0]- Δvariable controlada[-1])*θ).*

El lugar 10 genera una *manipulación [0] = manipulación [-1] + (Error[0])/(15*K_p).* La relación que tiene con K_p es inversamente proporcional a la manipulación de que se requiere(por lo tanto se coloca dividiendo a los demás elementos en la ecuación), ya que si la ganancia del proceso es grande |*Δmanipulación*| debe de ser pequeña, y si la ganancia del proceso es chica |*Δmanipulación*| debe de ser grande. El error siempre es directamente proporcional a *Δmanipulación* (por lo tanto se coloca multiplicando a los demás elementos en la ecuación), ya que si el |error| es grande |*Δmanipulación*| debe de ser grande, y si el |error| es chico |*Δmanipulación*| debe de ser pequeño. El "15" se incluyo en esta ecuación dividiendo para minimizar |*Δmanipulación*|.

El lugar 11 sirve para que se realice la transición 15 ó 16 una sola vez. El lugar 12 está asociado con un tiempo = θ, por lo tanto la transición 16 se habilitará hasta que transcurra θ después de que llegue un token al lugar 12. Los eventos 11 y 12 sirven para elegir que hacer si se cumple o no la siguiente condición: *Δvariable controlada es > 0 y el Error[0] < 0, ó Δvariable controlada es < 0 y el Error[0] > 0*, que es igual a que exista o no sobretiro. Lo que se obtiene con los lugares 10, 11 12, y transiciones 14, 15 y 16, es que se genere la manipulación asociada al lugar 10 hasta que transcurra θ del proceso o exista sobretiro, una vez que se de alguna de esas dos condiciones se pasará a la siguiente etapa.

La etapa 3 se incluyo para reducir el tiempo de elevación $\{t_r\}$en procesos con θ (cuando se tiene la K_p correcto), ya que si se conectaba la etapa 2 directamente con la etapa 4, algunos procesos presentaban t_r grande, es decir, el control de ciertos procesos era muy lento.

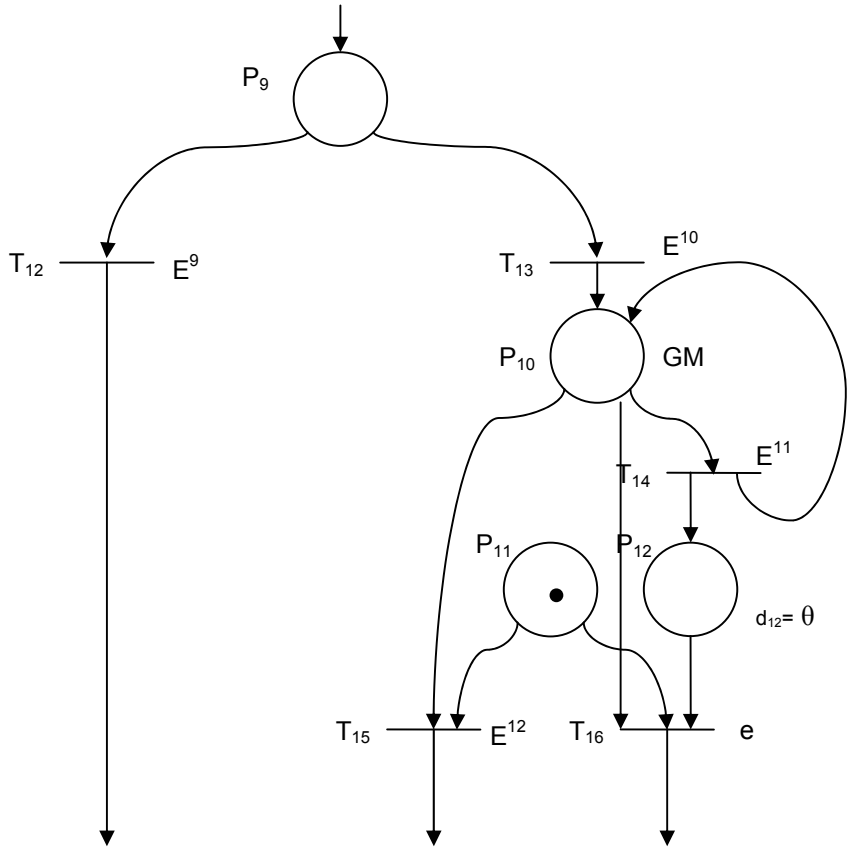


Figura 3-6: Etapa 3.

3.2.4. Etapa 4.

La etapa 4 contiene 14 lugares y 22 transiciones. Cuando se llega a esta etapa del control, después de realizar algún cambio de referencia, el error es muy pequeño comparado con el error inicial y $|\Delta variable\ controlada|$ también es un valor pequeño, lo cual significa que falta poco para que la *variable controlada* sea igual a la referencia y se estabilice. Esta etapa sirve para evitar el error de estado estable, y corregir perturbaciones.

Esta etapa comienza cuando llega un token al lugar 13, después se decide que hacer si *Δvariable controlada* es cero, positivo o negativo mediante la utilización de las transiciones 38, 17 y 18 que están ligadas a los eventos externos 17, 13 y 14.

- E^{17} se cumple si *Δvariable controlada es = 0.*
- E^{13} se cumple si *Δvariable controlada es > 0.*
- E^{14} se cumple si *Δvariable controlada es < 0*

Si se cumple el evento 13 ó 14 posteriormente se pueden tener dos opciones nuevamente para cada uno de los casos, y es: que el Error⌈0⌉ sea mayor o igual a 0, ó que el Error⌈0⌉ sea menor que cero. Mediante las transiciones 19 (ligada a E^{15}), 20 (ligada a E^{16}), 21 (ligada a E^{15}), y 22 (ligada a E^{16}) se generan esas decisiones.

- E^{15} se cumple si *Error⌈0⌉ ≥ 0.*
- E^{16} se cumple si *Error⌈0⌉ < 0.*

Por lo tanto si se cumple:

- La transición 19 significa que *Δvariable controlada* es positivo, y el Error⌈0⌉ es mayor o igual a 0.
- La transición 20 significa que *Δvariable controlada* es positivo, y el Error⌈0⌉ es menor que 0.
- La transición 21 significa que *Δvariable controlada* es negativo, y el Error⌈0⌉ es mayor o igual a 0.
- La transición 22 significa que *Δvariable controlada* es negativo, y el Error⌈0⌉ es menor que 0.

Si se cumple la transición 20 ó 21, quiere decir que existe sobretiro, por lo tanto las manipulaciones que generan los lugares 16 y 17 son iguales.

- P_{16} y P_{17} generan una *manipulación ⌈0⌉ = manipulación ⌈-1⌉+(2*ErrorC⌈0⌉/K_p).*

K_p actúa inversamente proporcional a la manipulación requerida para reducir el error (ya que si K_p es grande la manipulación es pequeña, y si K_p es chico la manipulación debe de ser grande), por lo tanto ese término se coloca en la ecuación dividiendo a los demás elementos. Como ya se ha mencionado anteriormente el Error es directamente proporcional a la manipulación requerida, por tanto multiplica a los demás elementos. El "2" se colocó en la ecuación para incrementar |*Δmanipulación*|.

Si se cumple la transición 19 ó 22, se decide entre varias opciones (iguales a las que se presentan en la etapa 2 del controlador), mediante las transiciones 25 (ligada a E^6), 26 (ligada a E^7), 27 (ligada a E^8), 28 (ligada a E^6), 29 (ligada a E^7) y 30 (ligada a E^8). Para cada uno de los casos posible se genera una manipulación mediante el paso del token por alguno de los siguientes lugares: 20, 21, 22, 23,24 ó 25. La *Δmanipulación* ligada a los lugares 20 y 23, 21 y 24,

y 22 y 25, es la misma pero con signo contrario, ya que son las mismas condiciones del proceso pero en sentido opuesto.

La manipulación que genera el lugar 22 y 25 es muy parecida a la que genera el lugar 8, ya que las condiciones son muy similares

- P_{22} y P_{25} generan una *manipulación = manipulación [-1]+ ((0.025*(Δmanipulacion[-1-θ]/Δvariable controlada[0])*ErrorC[0])/(1+θ))*. El valor de esta manipulación está a limitada a valor absoluto máximo menor o igual a valor absoluto del error actual por dos.

(Δvariable controlada[0]/ (Δmanipulacion[-1-θ]) nos dice como responde el proceso (después de haber transcurrido θ) a una manipulación. Este valor es inversamente proporcional a la manipulación necesaria para reducir el error (ya que si este valor es grande se necesita poca *|Δmanipulación|*, y si este valor es chico se requiere *|Δmanipulación|* grande), por lo tanto este término divide a los demás elementos de la ecuación. El "0.025" se incluyo para que *|Δmanipulación|* no sea muy grande. θ se incluyo para reducir *|Δmanipulación|*en procesos con tiempo muerto.

En la ecuación ligada a los lugares 21 y 24 únicamente se incluyó el *error* (calculado para procesos con θ) y *Δvariable controlada error* (calculado para procesos con θ), ambas variables son directamente proporcionales a la manipulación necesaria para reducir el error, por lo tanto se multiplican.

- P_{21} genera una *manipulación [0] = manipulación[-1]-(Δvariable controlada C* ErrorC[0])*.
- P_{24} genera una *manipulación [0] = manipulación[-1]+(Δvariable controlada C* ErrorC[0])*.

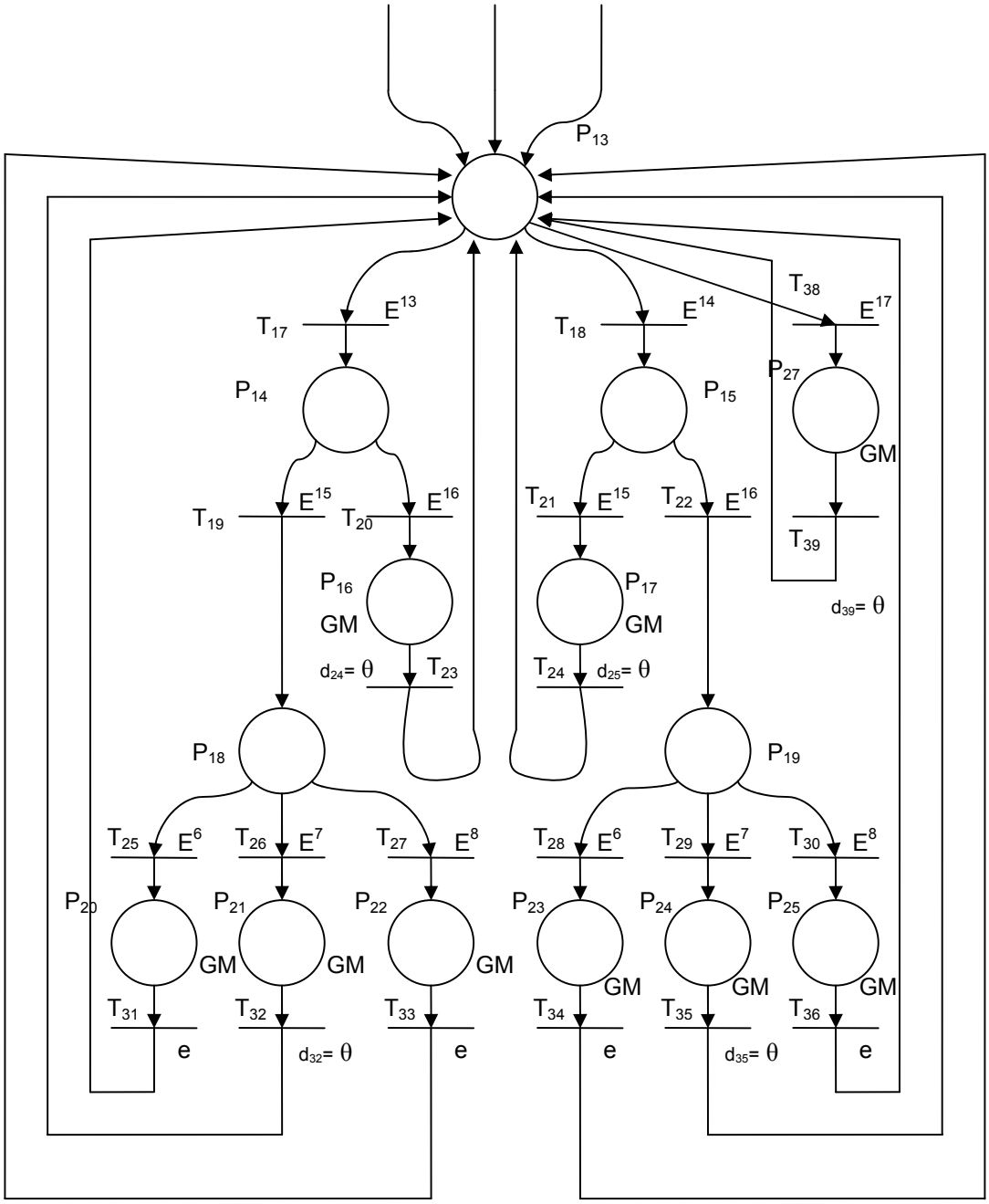


Figura 3-7: Etapa 4.

Las ecuaciones que están ligadas con el lugar 20 y 23 son similares a las de los lugares 22 y 25, pero además se les agregó el θ para reducir $|\Delta manipulación|$ en procesos con tiempo muerto. El "2" se incluyo para minimizar $|\Delta manipulación|$.

- P_{20} genera una *manipulación [0] = manipulación[-1] - ((Δvariable controlada C* ErrorC[0])/((2+ θ))*.
- P_{23} genera una *manipulación [0] = manipulación[-1]+ ((Δvariable controlada C* ErrorC[0])/((2+ θ))*.

El lugar 27 genera una *manipulación [0] = manipulación [-1] + Error[0]*, lo cual ocurre si *Δvariable controlada es = 0* . Este lugar se incorporó para eliminar el error de estado estable.

Las transiciones T_2, T_{23}, T_{24}, T_{32}, T_{35} *y* T_{38} están asociadas a un *tiempo = θ*, lo cual significa que cuando se activen dichas transiciones el token tardará θ en pasar al lugar 13, esto se hizo para que el controlador mantenga la misma manipulación y pueda verificar como responde el proceso antes de realizar otra acción correctiva. Esta acción se realiza en lugares que están asociados con *|Δmanipulación|*grandes. En los lugares que no tienen en la salida una transición asociada a un *tiempo = θ,* se incluyo en la manipulación que generan el tiempo muerto dividiendo a los demás elementos de la ecuación, para reducir *|Δmanipulación|*, ya que no se podrá verificar el efecto de esa manipulación en el proceso hasta que transcurra θ.

En cuanto el *|error|* es menor a 1 y *|Δvariable controlada|* es menor que 0.25 el controlador realiza un cálculo de la ganancia del proceso, mediante la siguiente ecuación:

K_p *= (Variable controlada en estado estable después de realizar un cambio de referencia – variable controlada antes de realizar el cambio de referencia)/(manipulación a la que se tiene la variable controlada en estado estable después de realizar un cambio de referencia – manipulación que se tenía antes de realizar el cambio de referencia).*

Con esta acción de autosintonización, el controlador mejora su desempeño después de que se realiza el primer cambio de referencia, si es que la ganancia del proceso no era igual a la calculada previamente o inicial del programa.

Capítulo 4

Implantación computacional

4.1. Panel principal del programa.

La implantación computacional del controlador desarrollado mediante Redes de Petri, se llevó a cabo utilizando el software de National Instruments LabWindows CVI 5.0, con este paquete computacional se pueden realizar interfaces hombre máquina (HMI) mediante la programación en lenguaje C. Este programa facilita la programación de objetos visuales ya que cuenta con gráficas, botones, paneles, íconos, selectores, entre otros.

Al correr el programa se mostrará en pantalla el panel de la figura 4-1, el cual brinda información del proceso en forma gráfica y numérica, y en donde el usuario puede realizar diversas acciones.

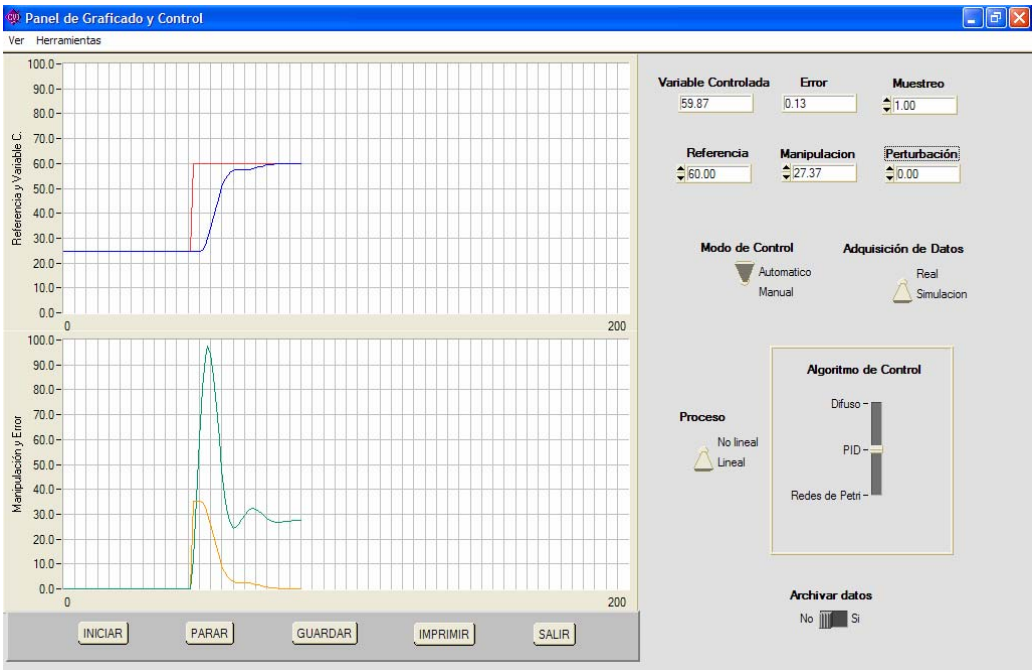


Figura 4-1: Panel principal.

A continuación se describirá todos los elementos con los que cuentan los distintos paneles:

- *Gráfica superior*: Muestra el valor de la variable controlada a través del tiempo, así como la referencia. La variable controlada se grafica con una línea de color azul, y la referencia con una línea de color rojo. El eje vertical de la gráfica tiene un rango de 0 a 100, con divisiones de valor 10, y el eje horizontal muestra 200 segundos en donde cada cuadro representa 4 segundos.
- *Gráfica inferior*: Muestra el valor de la manipulación a través del tiempo, así como la error. La manipulación se grafica con una línea de color verde, y el error con una línea de color rojo. Al igual que la gráfica superior, el eje vertical de la gráfica tiene un rango de 0 a 100, con divisiones de valor 10, y el eje horizontal muestra 200 segundos en donde cada cuadro representa 4 segundos.
- *Botón Iniciar:* Con este botón se inicia el graficado y el cálculo de todas las variables que se muestran en el panel principal cuando se acaba de correr el programa, y además sirve para continuar con el graficado y cálculos cuando se ha presionado previamente el *botón Parar*.
- *Botón Parar:* Con este botón se detiene el graficado, y los valores numéricos que se muestran en el panel principal se mantienen en el mismo valor, de igual forma se detiene la simulación. Para que el programa continúe con el graficado de variables y realizando cálculos, es necesario presionar nuevamente el botón iniciar.
- *Botón Guardar:* Con este botón se guardan los valores de la variable controlada a partir de que se posicionó el selector "Archivar datos" en "Sí". Los valores se guardan en un vector de tamaño máximo 900. Al oprimir este botón se desplegara la siguiente ventana:

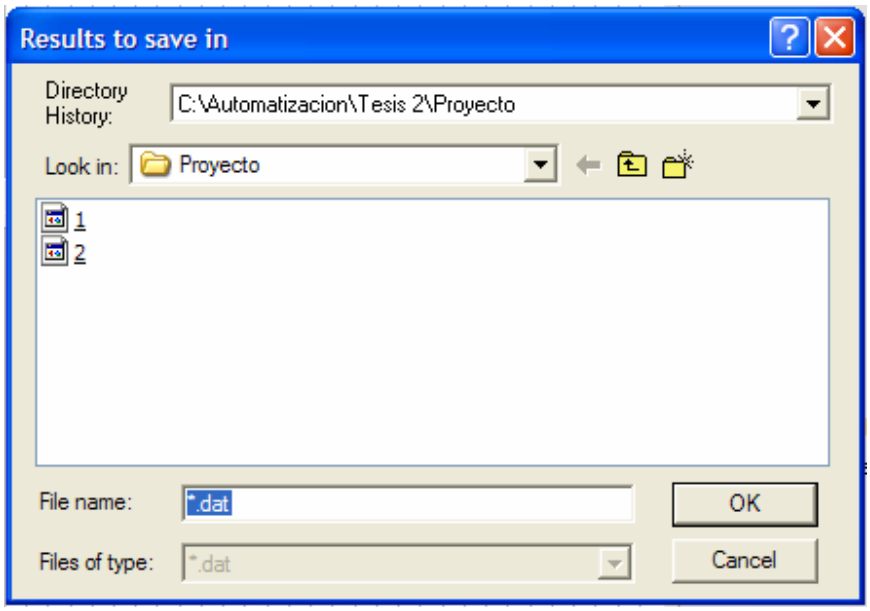


Figura 4-2: Ventana para guardar datos.

En esta ventana se le da nombre al archivo que contiene los datos, y se puede elegir en que carpeta queremos guardarlo.

- *Botón Imprimir*: Este botón sirve para mandar imprimir la gráfica que muestra la variable controlada y referencia. Al oprimir este botón se desplegará la siguiente ventana:

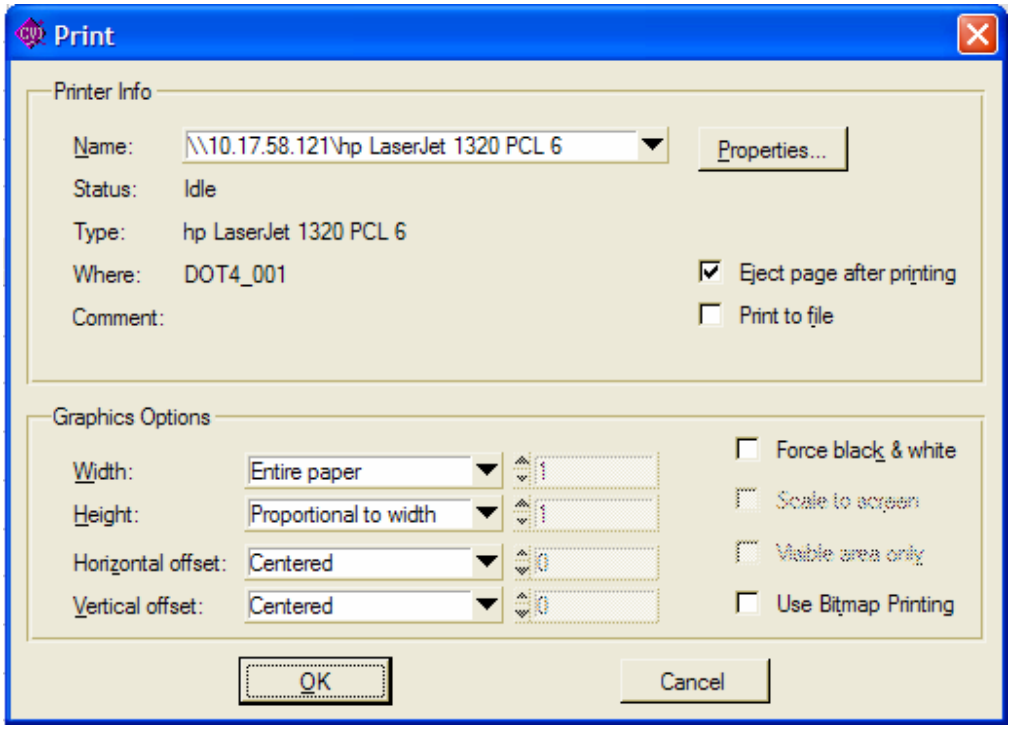


Figura 4-3: Ventana de impresión.

En esta ventana se puede seleccionar la impresora a la cual se desea mandar la impresión.

- *Botón Salir:* Al oprimir este botón se terminará con la ejecución del programa, y desaparecerá el panel principal de la pantalla.
- *Selector Archivar datos.* Al posicionar este selector en "Sí", el programa comenzará a almacenar los distintos valores de la variable controlada a través del tiempo, en un vector, el cual se guardará en un archivo (*.dat)* en cuanto se presione el botón "Guardar".
- *Selector Adquisición de Datos.* Con este selector se puede elegir si el proceso que se desea controlar es un proceso real o un proceso simulado. El valor predeterminado es proceso simulado. Si se selecciona proceso real, la computadora se comunicará con el proceso a través del puerto serial.
- *Selector Modo de Control.* Con él se puede elegir si el control sobre el proceso se desea realizar de forma manual o automática. Si se selecciona de forma manual; el valor de la referencia será igual al de la variable controlada (para generar transferencias sin brincos) no se podrá

modificar, y el valor de la manipulación si se podrá modificar. Si se selecciona de forma automática; el valor de la referencia se podrá modificar, pero el de la manipulación no, ya que el controlador generará las manipulaciones adecuadas para eliminar el error.

- *Selector de Algoritmo de Control.* Con él se puede elegir que tipo de controlador se desea utilizar cuando el control sobre el proceso se realiza de forma automática. El programa cuenta con tres controladores; PID, difuso y mediante Redes de Petri.
- *Selector de Proceso.* Con este selector se puede elegir si el proceso que deseamos simular sea lineal o no lineal. El proceso lineal es un sistema de primer orden que se puede modificar mediante la modificación de τ, K_p, y θ. El proceso no lineal simula el llenado automático de un tanque de agua cónico, que no se puede modificar.
- *Indicador numérico de Variable Controlada.* Este indicador muestra el valor actual de la variable controlada. Su rango es de –infinito a +infinito.
- *Indicador numérico del Error.* Muestra el valor actual del error. Su rango es de –infinito a +infinito. La ecuación para calcular el error es: *Error = Referencia – Variable Controlada.*
- *Indicador numérico de Tiempo de Muestreo.* Muestra y permite modificar el tiempo de muestreo. Su valor predeterminado es un segundo.
- *Indicador numérico de Referencia.* Muestra el valor actual de la referencia o Set point, y permite modificarlo cuando el modo de control es automático. Su rango de operación es de 0 a 100.
- *Indicador numérico de Manipulación.* Muestra el valor actual de la manipulación, y permite modificarla cuando el modo de control es manual. Su rango de operación es de 0 a 100.
- *Indicador numérico de Perturbación.* Permite introducir perturbaciones a procesos simulados. Su rango de operación es de -100 a 100, y su valor predeterminado es cero.
- *Barra de herramientas.* En la parte superior derecha del panel principal se localiza la barra de herramientas en ella se puede seleccionar: *Autor*, *Cambiar proceso* y *Modificar PID.*
  - Si se selecciona *Autor* se desplegará en pantalla el panel de la figura 4-4.

Figura 4-4: Panel "Autor".

Para regresar al panel principal se tiene que presionar el botón "Regresar".

- Si se selecciona *Cambiar proceso* se desplegará en pantalla el panel de la figura 4-5.

Figura 4-5: Panel "Parámetros del Proceso".

En este panel se puede modificar la ganancia (K_p), la constante de tiempo (τ) y el tiempo muerto (θ) del proceso simulado. Además se puede introducir ruido y cambiar el valor de la variable controlada de estado estable, es decir, su valor con *manipulación* = 0. Para regresar al panel principal es necesario presionar el botón "Salir". En cuanto se regresa a la pantalla principal comenzará la simulación del nuevo sistema.

- Si se selecciona *Modificar PID* se desplegará en pantalla el panel de la figura 4-6.

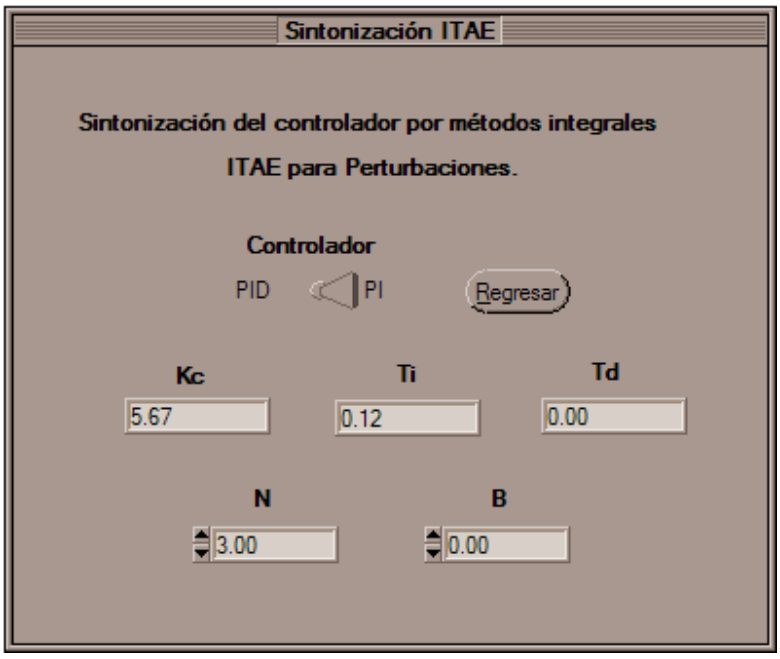


Figura 4-6: Panel de Sintonización del controlador PID.

En este panel se puede elegir entre un controlador PI y PID, y se puede cambiar su variables. El rango de valores de "N" es de 3 a 20, y el de "B" es de 0 a 1. Cada vez que se abra este panel, sino se modifica las variables del controlador, el controlador se sintonizará automáticamente para proceso que se está simulando actualmente. Para regresar al panel principal se tiene que presionar el botón "Regresar".

4.2. Procesos simulados.

En el programa se puede simular un proceso lineal o un proceso no lineal. El proceso lineal es un sistema de primer orden, el cual se puede modificar en el panel de control. El proceso no lineal es el llenado automático de un tanque de agua cónico, el cual no se puede modificar en el panel de control. A continuación se mostrará las ecuaciones que se utilizaron para simular ambos procesos.

4.2.1. Sistema lineal.

El proceso lineal es un sistema de primer orden cuya función de transferencia es de la forma siguiente:

$$Gp(s) = \frac{k}{\tau s + 1} e^{-\theta s} \tag{4.1}$$

Para realizar la digitalización se utiliza la ecuación 4.2.

$$HGp(z) = \frac{b1z^{-1-N} + b2z^{-n-2}}{1 + a1z^{-1}} = \frac{Y(z)}{U(z)} \tag{4.2}$$

donde;

$$Y(z) = Salida\ del\ proceso \tag{4.3}$$

$$U(z) = Entrada\ del\ proceso \tag{4.4}$$

$$T = tiempo\ de\ muestreo \tag{4.5}$$

$$N = piso(\theta/T) \tag{4.6}$$

$$f = \theta - NT \tag{4.7}$$

$$m = 1 - f/T \tag{4.8}$$

$$a1 = -e^{-T/\tau} \tag{4.9}$$

$$b1 = k_p(1 - e^{-mT/\tau}) \tag{4.10}$$

$$b2 = k_p(e^{-mT/\tau} - e^{-T/\tau}) \tag{4.11}$$

Si la ecuación 4.2 se transforma en una ecuación de diferencias obtenemos:

$$b1U_{k-1-N} + b2U_{k-2-N} = Y_k + a1Y_{k-1} \tag{4.12}$$

Si despejamos de la ecuación 4.12 Y_k, obtenemos:

$$Y_k = -a1Y_{k-1} + b1U_{k-1-N} + b2U_{k-2-N} \tag{4.13}$$

Se incluyen valores iniciales:

$$Y_k - Yss = -a1(Y_{k-1} - Yss) + b1(U_{k-1-N} - Uss) + b2(U_{k-2-N} - Uss) \tag{4.14}$$

Finalmente se despeja Y_k:

$$Y_k = -a1Y_{k-1} + b1U_{k-1-N} + b2U_{k-2-N} + Yss(1 + a1) - Uss(b1 + b2) \tag{4.15}$$

4.2.2. Sistema no lineal.

El proceso no lineal es el llenado automático de un tanque de agua cónico, el cual tiene las siguientes dimensiones.

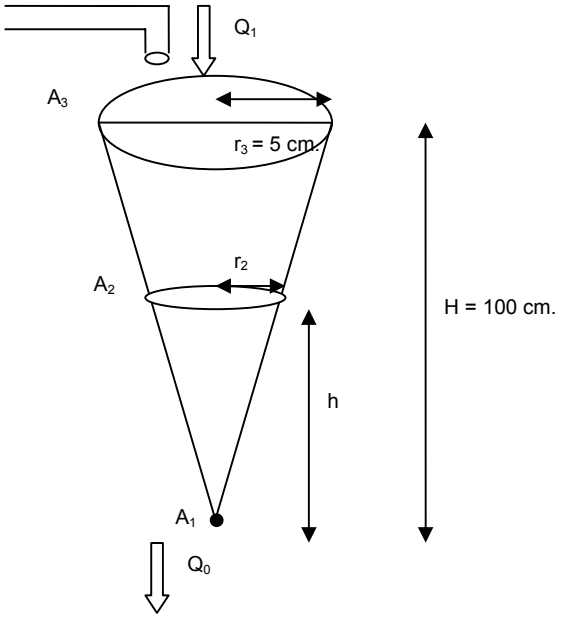


Figura 4-7: Tanque cónico.

El radio del orificio en la parte inferior del tanque (r_1) es igual a 0.3 cm. La ecuación diferencial del tanque de la figura 4-7 es:

$$\frac{dh}{dt} = \frac{Q_1 - Q_0}{A_2} \tag{4.16}$$

$$Q_0 = A_1 * \sqrt{2gh} \tag{4.17}$$

$Q_1 = 0.5cm^3$ por cada 1% de apertura de la válvula. (4.18)

$$A_2 = \pi r_2^2 \tag{4.19}$$

$$r_2 = 0.05h \tag{4.20}$$

$$A_2 = 0.0025\pi h^2 \tag{4.21}$$

$$\frac{dh}{dt} = \frac{Q_1 - A_1\sqrt{2gh}}{0.0025\pi h^2} \tag{4.22}$$

$$\frac{dh}{dt} = \frac{127.32395Q_1}{h^2} - \frac{159.37879}{\mathrm{h}^{3/2}} \tag{4.23}$$

Para generar la ecuación de diferencias a partir de la ecuación 4.23, se utiliza la fórmula de Runge-Kutta de cuarto orden con T = 1segundo.

$$k_1 = \frac{63.661977\mathrm{U}_{\mathrm{k\text{-}1}}}{\mathrm{Y}_{\mathrm{k\text{-}1}}^2} - \frac{159.37879}{Y_{k-1}^{3/2}} \tag{4.24}$$

$$k_2 = \frac{63.661977(\mathrm{U}_{\mathrm{k\text{-}1}} + 0.5)}{\left(\mathrm{Y}_{\mathrm{k\text{-}1}} + 0.5k_1\right)^2} - \frac{159.37879}{\left(Y_{k-1} + 0.5k_1\right)^{3/2}} \tag{4.25}$$

$$k_3 = \frac{63.661977(\mathrm{U}_{\mathrm{k\text{-}1}} + 0.5)}{\left(\mathrm{Y}_{\mathrm{k\text{-}1}} + 0.5k_2\right)^2} - \frac{159.37879}{\left(Y_{k-1} + 0.5k_2\right)^{3/2}} \tag{4.26}$$

$$k_4 = \frac{63.661977(\mathrm{U}_{\mathrm{k\text{-}1}} + 1)}{\left(\mathrm{Y}_{\mathrm{k\text{-}1}} + k_3\right)^2} - \frac{159.37879}{\left(Y_{k-1} + 0.5k_3\right)^{3/2}} \tag{4.27}$$

$$Y_k = Y_{k-1} + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \tag{4.28}$$

Las ecuaciones 4.24, 4.25, 4.26, 4.27 y 4.28 son las que se incluyen en el programa para realizar la simulación del proceso no lineal.

4.3. Proceso real.

El proceso real que se utilizó para la realización de pruebas es una estación para el control de la temperatura de salida de un sistema de secado industrial, está estación tiene la opción de ser controlada por una computadora mediante la comunicación por el puerto serial. La figura 4-8 muestra un diagrama esquemático del proceso de temperatura, entre sus características más importantes se encuentran:

- La fuente de alimentación es de 127 VCA.
- La señal de temperatura es medida por un sensor de temperatura con señal de 10 mV/°C, el cual está montado a la salida del flujo de aire, la señal puede transmitirse en un rango de 0 a 10V ó 4 a 20 mA.
- La manipulación que recibe del puerto serial de la computadora es procesada por un microcontrolador de 10 bits, el cual se encarga de llevar el tiempo de muestreo, así como controlar el ángulo de disparo para regular el voltaje suministrado a la resistencia que calienta el flujo de aire de salida.
- El flujo de aire de salida tiene un control de velocidad de la turbina que lo genera, la velocidad está graduada en 4 niveles.

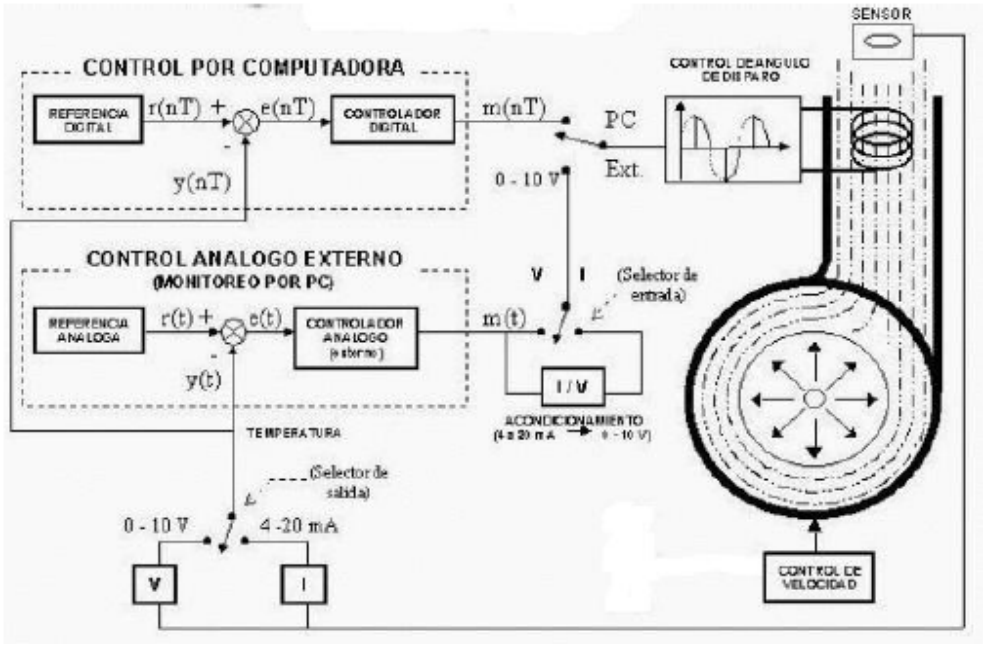


Figura 4-8: Diagrama esquemático del proceso de temperatura.

4.4. Controladores.

Como ya se ha mencionado anteriormente el programa cuenta con tres controladores: controlador PID, controlador difuso y controlador mediante Redes de Petri. En el capítulo dos se explicó el funcionamiento del controlador mediante Redes de Petri. A continuación se presentará las ecuaciones y algoritmos que se utilizaron para la implementación de los controladores mencionados.

4.4.1. PID.

El controlador PID se puede sintonizar automáticamente al proceso simulado, a través del panel mostrado en la figura 4-6. El criterio que se utiliza para la sintonización es ITAE para perturbaciones. Se eligió este criterio ya que con él se obtiene un buen desempeño del controlador ante perturbaciones, las cuales son muy comunes en los procesos reales.

- Para PI.

$$Kc = \frac{0.859}{K}\left(\frac{\theta}{\tau}\right)^{-0.977} \tag{4.29}$$

$$\tau i = 1/\left(\frac{\tau}{0.674}\left(\frac{\theta}{\tau}\right)^{0.680} \right) \tag{4.30}$$

- Para PID.

$$Kc = \frac{1.357}{K}\left(\frac{\theta}{\tau}\right)^{-0.947} \tag{4.31}$$

$$\tau i = 1/\left(\frac{\tau}{0.842}\left(\frac{\theta}{\tau}\right)^{0.738} \right) \tag{4.32}$$

$$\tau d = 0.381\tau\left(\frac{\theta}{\tau}\right)^{0.995} \tag{4.33}$$

Debido a que el proceso de muestreo introduce un retraso natural entre una acción (manipulación) y la observación de su efecto. Se utiliza la siguiente ecuación.

$$\theta_{calculado} = \theta_{estimado} + 3T/4 \tag{4.34}$$

La estructura del controlador que se utiliza es PID ISA con filtro en el derivativo. Se eligió está estructura porque con ella se tiene poco sobretiro ante cambios de referencia cuando se sintoniza para perturbaciones.

$$U(s) = Kc[bR(s) - Y(s) + \mathrm{Re}set * E(s)/s - \frac{\tau ds}{\frac{\tau ds}{N}Y(s)}] \tag{4.35}$$

$$U(s) = Kc * b * R(s) - Kc * Y(s) + Kc * \mathrm{Re}set * E(s)/s - Kc * \frac{\tau ds}{\frac{\tau ds}{N}}Y(s) \tag{4.36}$$

Debido a que la ecuación del controlador PID se tiene en tiempo continuo, ésta se tiene que discretizar, y transformar en una ecuación de diferencias para poderse incluir en el programa computacional.

La manipulación que genera un controlador PID es la suma de la manipulación proporcional más la manipulación integral más la manipulación derivativa.

$$Manipulación = ManP + ManI + ManD \tag{4.37}$$

$$ManP(s) = Kc[bR(s) - Y(s)] \tag{4.38}$$

$$ManI(s) = Kc[\mathrm{Re}set * E(s)/s] \tag{4.39}$$

$$ManD(s) = Kc[-\frac{\tau ds}{\frac{\tau ds}{N}}]Y(s) \tag{4.40}$$

- *Manipulación Proporcional.*

$$ManP(s) = Kc[bR(s) - Y(s)] \tag{4.41}$$

$$ManP(z) = Kc[bR(z) - Y(z)] \tag{4.42}$$

$$ManP_K = Kc[bR_K - Y_K] \tag{4.43}$$

$$ManP_K = -ManP_K + Kc[b(R_K - R_{K-1}) - (Y_K - Y_{K-1})] \tag{4.44}$$

La ecuación 4.43 está en modo posición, y la ecuación 4.44 en modo velocidad.

- *Manipulación integral.*

$$ManI(s) = Kc[\mathrm{Re}set * E(s)/s] \tag{4.45}$$

Utilizando la transformación bilineal obtenemos:

$$ManI(z) = Kc[\frac{\mathrm{Re}set * E(z)}{\frac{2}{T}\frac{1-z^{-1}}{1+z^{+1}}}] \tag{4.46}$$

Finalmente se obtiene la ecuación de diferencias en modo velocidad:

$$ManI_K = ManI_{K-1} + Kc[\frac{\mathrm{Re}set * T * (E_K + E_{K-1})}{2}] \tag{4.47}$$

- *Manipulación derivativa.*

$$ManD(s) = Kc[-\frac{\tau ds}{\frac{\tau ds}{N}}]Y(s) \tag{4.48}$$

Utilizando la transformación bilineal obtenemos:

$$ManD(z) = \frac{-Kc * \tau d\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}}{\frac{\tau d}{N}\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}+1}Y(z) \tag{4.49}$$

Finalmente se obtiene la ecuación de diferencias:

$$ManD_K = ManD_{K-1} + \frac{2\tau d - NT}{2\tau d + NT}(ManD_{K-1} + ManD_{K-2}) - \frac{2Kc\tau dN}{2\tau d + NT}(Y_K - 2Y_{K-1} + Y_{K-2}) \tag{4.50}$$

Las ecuaciones que están en el programa para realizar las acciones de control son: la 4.37, 4.44, 4.47 y 4.50. Las ecuaciones 4.29, 4.30, 4.31, 4.32, 4.33, y 4.34, también están incluidas, para sintonizar automáticamente el controlador.

4.4.2. Controlador Difuso.

La estructura de un controlador difuso (FKBC Fuzzy knoledge based controller) se muestra en la figura 4-8. La función de cada uno de sus elementos es la siguiente:

- *Modulo de difusificación.* El modulo de difusificación (FM) ejecuta las siguientes acciones: (FM-F1) realiza una normalización de las variables de estado del proceso. Y (FM-F2) convierte los valores normalizados en un conjunto difuso.
- *Conocimiento base.* El conocimiento base de un FKBC esta compuesto por una base de datos y reglas. La base de datos provee información al módulo de difusificación para que funcione apropiadamente, y las reglas brindan información al módulo de desdifusificación para que funcione adecuadamente.
- *Máquina de inferencia.* Las maquinas de inferencia más utilizadas son la de Mamdani y Gödel. La máquina de inferencia genera un conjunto de salidas difusas de control, a partir de una entrada de conjuntos difusos.
- *Modulo de desdifusificación.* Las funciones del modulo de desdifusificación (DM) son: (DM-F1) convertir los conjuntos difusos de salida en un valor numérico. Y (DM-F2) realizar una desnormalización de los valores numéricos, de tal forma que se genera la salida de control en el dominio físico.

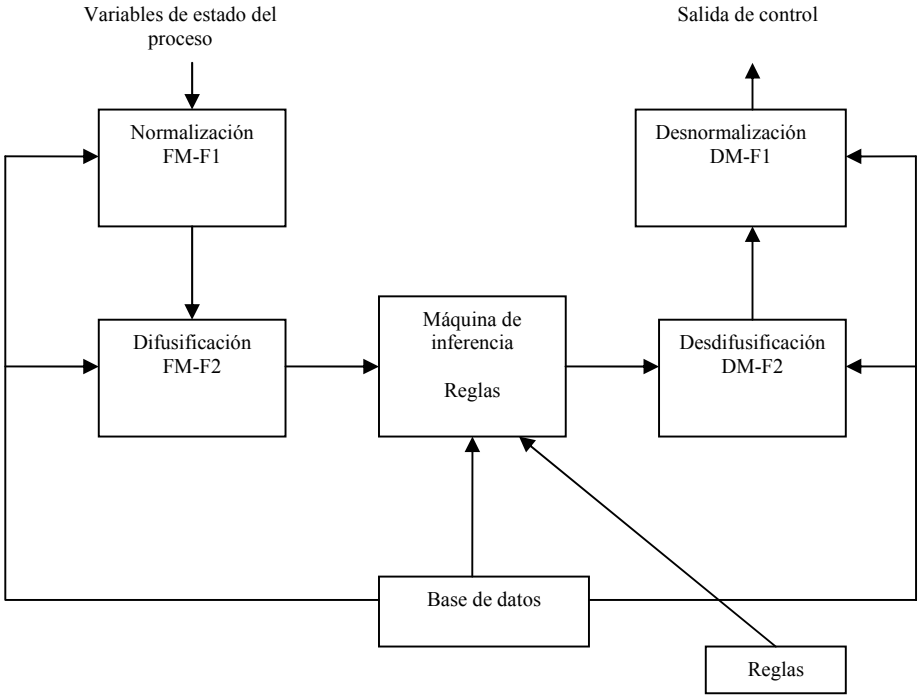


Figura 4-9: Estructura de un FKBC.

El controlador difuso no puede ser sintonizado desde el panel de control. El controlador difuso desarrollado se sintonizó para dos procesos, uno real y uno simulado. El cambio de algoritmo de control se realiza automáticamente al cambiar entre proceso real y simulado, y viceversa.

Proceso Real.

El proceso real es la estación descrita en la sección 4.3, en la velocidad 1. La función de transferencia del proceso es la siguiente.

$$\frac{1.3}{21s+1}e^{-3.5s} \tag{4.51}$$

La temperatura de estado estable del proceso es 20.7° centígrados.

El FKBC diseñado es PI. La ecuación para un controlador PI convencional es la siguiente

$$u = K_P \cdot e + K_I \cdot \int e dt \tag{4.52}$$

Donde K_p y K_I son los coeficientes de ganancia proporciona e integral. La ecuación 4.52 se puede transformar en:

$$\dot{u} = K_P \cdot \dot{e} + K_I \cdot e \tag{4.53}$$

Un FKBC PI consiste en reglas de la forma: Si e es < > y Δe es < >, entonces Δu es < >. Donde e es el error, Δe es el cambio del error, u es la manipulación, y Δu es el cambio de manipulación.

$$e(k) = y_{sp} - y(k) \tag{4.54}$$

$$\Delta e(k) = e(k) - e(k-1) \tag{4.55}$$

$$\Delta e(k) = -(y(k) - y(k-1)) \tag{4.56}$$

$$\Delta e(k) = y(k-1) - y(k) \tag{4.57}$$

y_{sp} es la referencia, y es la variable controlada y k es el tiempo de muestreo.

Las entradas del FKBC son el error (e) y el cambio del error (Δe), y la salida es el cambio de manipulación (Δu). Las figuras 4-10 y 4-11 muestran los conjuntos difusos de entrada del KFBC.

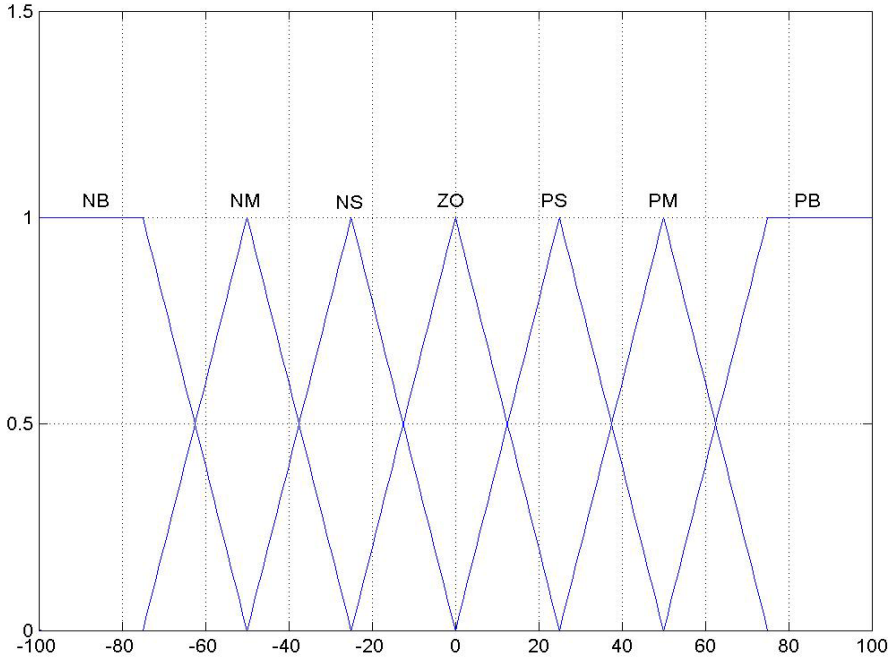


Figura 4-10: Conjuntos difusos del error.

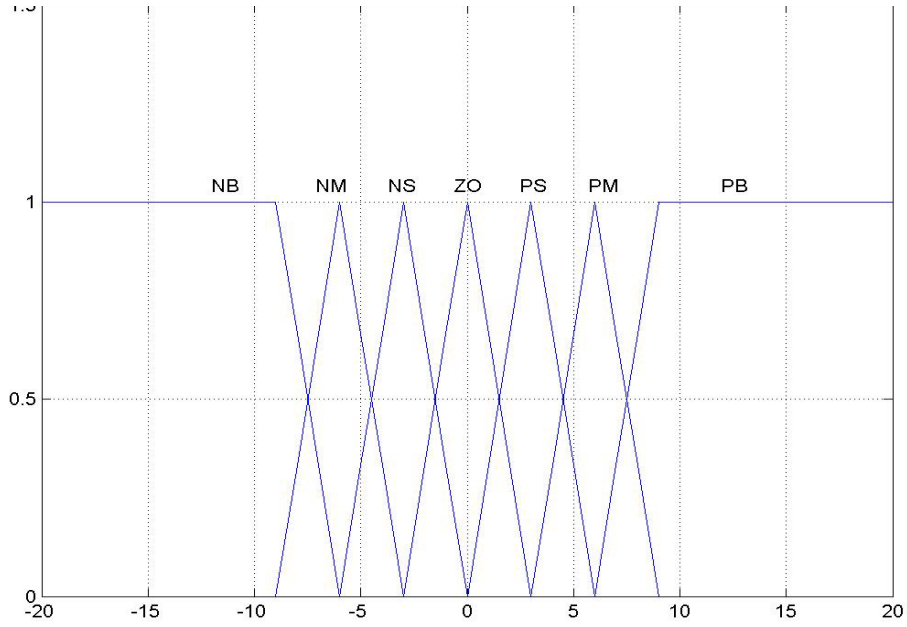


Figura 4-11: Conjuntos difusos del cambio del error.

NB es negativo grande, *NM* es negativo mediano, *NS* es negativo pequeño, *ZO* es cero, *PS* es positivo pequeño, *PM* es positivo mediano y *PB* es positivo grande. El cambio de manipulación (Δu) es un singleton, tal y como se muestre en la figura 4-12.

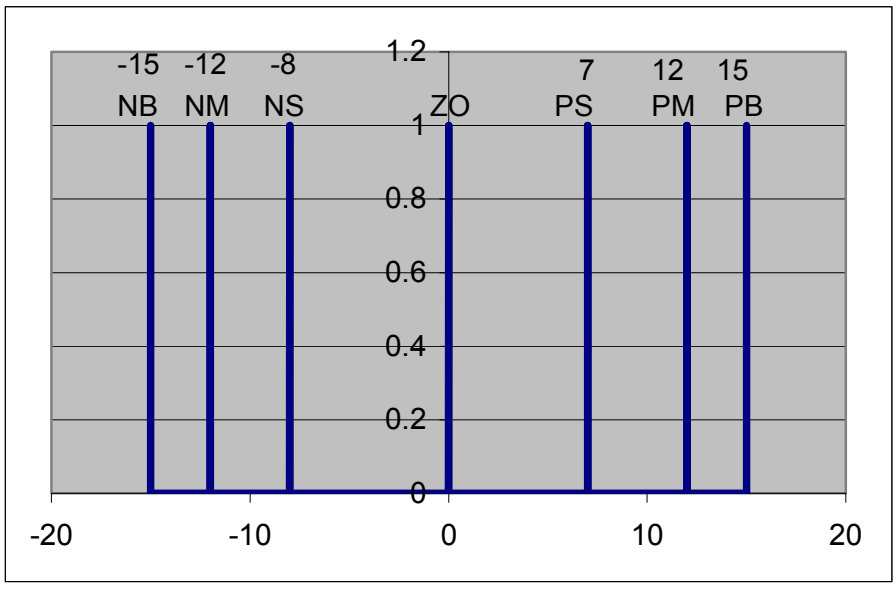


Figura 4-12: Cambio de manipulación.

Existen dos entradas (*e* y *Δe*) cada una con siete conjuntos difusos, por lo tanto el número de reglas es igual a 49, ya que es el número de combinaciones que se puede dar entre los conjuntos

de entrada. El comportamiento del cambio de manipulación (Δu) que envía el controlador difuso está dado por la tabla 4.1.

e\Δe	NB	NM	NS	ZO	PS	PM	PB
NB	NB	NB	NB	NB	NM	NS	ZO
NM	NB	NB	NB	NM	NS	ZO	PS
NS	NB	NB	NM	NS	ZO	PS	PM
ZO	NB	NM	NS	ZO	PS	PM	PB
PS	NM	NS	ZO	PS	PM	PB	PB
PM	NS	ZO	PS	PM	PB	PB	PB
PB	ZO	PS	PM	PB	PB	PB	PB

Tabla 4.1: Comportamiento de cambio de manipulaciones.

De la tabla 4.1, se pueden obtener las 49 reglas difusas, las cuales son:

1. Si *e* es NB y Δe es NB entonces Δu es NB.
2. Si *e* es NB y Δe es NM entonces Δu es NB.
3. Si *e* es NB y Δe es NS entonces Δu es NB.
4. Si *e* es NB y Δe es ZO entonces Δu es NB.
5. Si *e* es NB y Δe es PS entonces Δu es NM.
6. Si *e* es NB y Δe es PM entonces Δu es NS.
7. Si *e* es NB y Δe es PB entonces Δu es ZO.
8. Si *e* es NM y Δe es NB entonces Δu es NB.
9. Si *e* es NM y Δe es NM entonces Δu es NB.
10. Si *e* es NM y Δe es NS entonces Δu es NB.
11. Si *e* es NM y Δe es ZO entonces Δu es NM.
12. Si *e* es NM y Δe es PS entonces Δu es NS.
13. Si *e* es NM y Δe es PM entonces Δu es ZO.
14. Si *e* es NM y Δe es PB entonces Δu es PS.
15. Si *e* es NS y Δe es NB entonces Δu es NB.
16. Si *e* es NS y Δe es NM entonces Δu es NB.
17. Si *e* es NS y Δe es NS entonces Δu es NM.
18. Si *e* es NS y Δe es ZO entonces Δu es NS.
19. Si *e* es NS y Δe es PS entonces Δu es ZO.

20. Si *e* es NS y *Δe* es PM entonces Δu es PS.

21. Si *e* es NS y *Δe* es PB entonces Δu es PM.

22. Si *e* es ZO y *Δe* es NB entonces Δu es NB.

23. Si *e* es ZO y *Δe* es NM entonces Δu es NM.

24. Si *e* es ZO y *Δe* es NS entonces Δu es NS.

25. Si *e* es ZO y *Δe* es ZO entonces Δu es ZO.

26. Si *e* es ZO y *Δe* es PS entonces Δu es PS.

27. Si *e* es ZO y *Δe* es PM entonces Δu es PM.

28. Si *e* es ZO y *Δe* es PB entonces Δu es PB.

29. Si *e* es PS y *Δe* es NB entonces Δu es NM.

30. Si *e* es PS y *Δe* es NM entonces Δu es NS.

31. Si *e* es PS y *Δe* es NS entonces Δu es ZO.

32. Si *e* es PS y *Δe* es ZO entonces Δu es PS.

33. Si *e* es PS y *Δe* es PS entonces Δu es PM.

34. Si *e* es PS y *Δe* es PM entonces Δu es PB.

35. Si *e* es PS y *Δe* es PB entonces Δu es PB.

36. Si *e* es PM y *Δe* es NB entonces Δu es NS.

37. Si *e* es PM y *Δe* es NM entonces Δu es ZO.

38. Si *e* es PM y *Δe* es NS entonces Δu es PS.

39. Si *e* es PM y *Δe* es ZO entonces Δu es PM.

40. Si *e* es PM y *Δe* es PS entonces Δu es PB.

41. Si *e* es PM y *Δe* es PM entonces Δu es PB.

42. Si *e* es PM y *Δe* es PB entonces Δu es PB.

43. Si *e* es PB y *Δe* es NB entonces Δu es ZO.

44. Si *e* es PB y *Δe* es NM entonces Δu es PS.

45. Si *e* es PB y *Δe* es NS entonces Δu es PM.

46. Si *e* es PB y *Δe* es ZO entonces Δu es PB.

47. Si *e* es PB y *Δe* es PS entonces Δu es PB.

48. Si *e* es PB y *Δe* es PM entonces Δu es PB.

49. Si *e* es PB y *Δe* es PB entonces Δu es PB.

La máquina de inferencia que se utiliza es la máquina de inferencia producto la cual utiliza: unión para la combinación de reglas individuales, implicación producto de Mamdani, producto algebraico para normas "t", y máximo para normas "s".

$$\mu B'(y) = \max_{l=1}^{M}\left[\sup_{X\in U}\left(\mu A'(x)\prod_{i=1}^{n}\ \mu A_1^l(x_1)\mu B^l(y)\right)\right] \tag{4.58}$$

El desdifusificador empleado es centro promedio.

$$y^* = \frac{\sum_{l=1}^{M} y^l w_l}{\sum_{l=1}^{M} w_1} \tag{4.59}$$

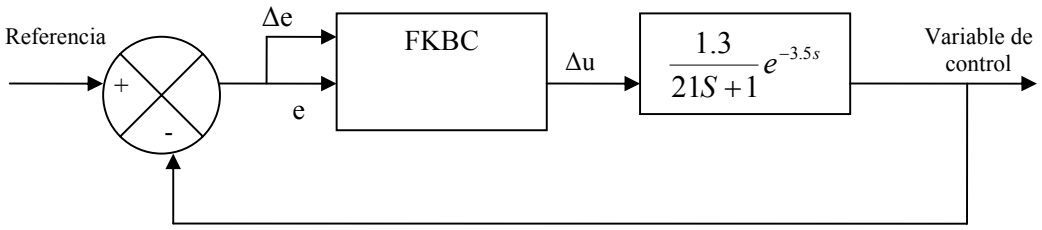


Figura 4-13: Diagrama de bloques del sistema de control.

Proceso Simulado.

El proceso simulado tiene la siguiente función de transferencia.

$$\frac{2}{20s+1}e^{-2s} \tag{4.51}$$

La temperatura de estado estable del proceso es 20° centígrados. En la sintonización del proceso simulado se utilizaron las mismas ecuaciones y conjuntos difusos que para el proceso real, excepto el conjunto difuso de cambio de manipulación. El cambio de manipulación (Δu) utilizado es un singleton, tal y como se muestre en la figura 4-14.

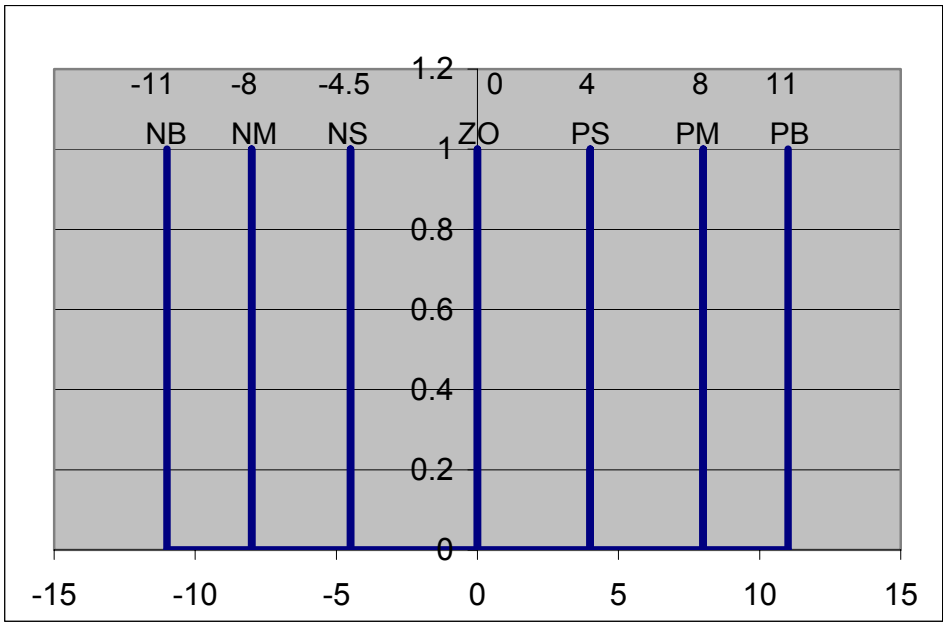


Figura 4-14: Cambio de manipulación para proceso simulado.

4.4.3. Controlador mediante Redes de Petri.

Como ya se mencionó en el capítulo 3, algunos lugares representan la magnitud de la manipulación que envía el controlador cuando tienen un token o más. Los lugares que generan alguna manipulación son el 2, 6, 7 , 8, 10, 16, 17, 20, 21, 22, 23, 24, 25, 26 y 27:

- P_2 genera una *manipulación [0]= manipulación [-1] + Δreferencia.*
- P_6 genera una *manipulación [0] = manipulación [-1].*
- P_7 genera *una manipulación [0] = manipulación antes de realizar el cambio de referencia + (Δreferencia/Ganancia del proceso {K_p}).*
- P_8 genera una *manipulación = manipulación [-1]+ 0.03*(Δmanipulacion[-1-θ]/Δvariable controlada[0])*ErrorC[0].* Si la manipulación anterior fue cero *manipulación = manipulación [-1] + Error[0].*
- P_{10} genera una *manipulación [0] = manipulación [-1] + (Error[0])/(15*K_p).*
- P_{16} y P_{17} generan una *manipulación [0] = manipulación [-1]+(2*ErrorC[0]/K_p).*
- P_{20} genera una *manipulación [0] = manipulación[-1]-((Δvariable controlada C* ErrorC[0])/((2+ θ)).*
- P_{21} genera una *manipulación [0] = manipulación[-1]-(Δvariable controlada C* ErrorC[0]).*

- P_{22} y P_{25} generan una *manipulación = manipulación [-1]+ ((0.025*(Δmanipulacion[-1-θ]/Δvariable controlada[0])*ErrorC[0])/(1+θ))*. El valor de esta manipulación está a limitada a valor absoluto máximo menor o igual a valor absoluto del error actual por dos.
- P_{23} genera una *manipulación [0] = manipulación[-1]+ ((Δvariable controlada C* ErrorC[0])/((2+ θ))*.
- P_{24} genera una *manipulación [0] = manipulación [-1]+(Δvariable controlada C* ErrorC[0])*.
- P_{26} genera una *manipulación [0] = manipulación [-1]*.
- P_{27} genera una *manipulación [0] = manipulación [-1] + Error[0]*.

Todas las ecuaciones anteriores están incluidas en el programa computacional.

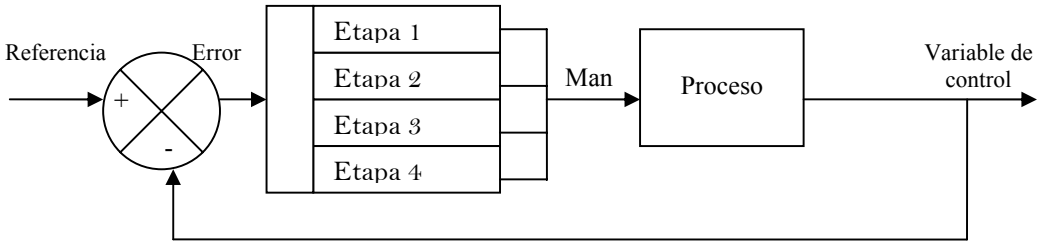


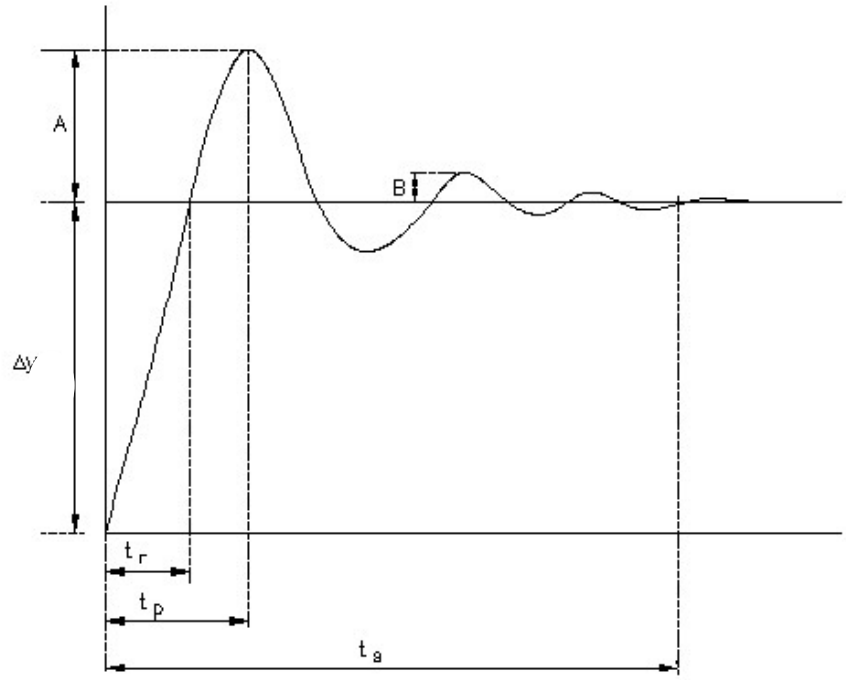
Figura 4-15: Diagrama de bloques del sistema de control de Redes de Petri .

Capítulo 5

Pruebas de validación del controlador

5.1. Simulación.

Para comprobar el funcionamiento apropiado del controlador mediante Redes de Petri se realizaron pruebas de control con distintos procesos simulados. Se utilizaron cuatro procesos lineales de primer orden y un proceso no lineal, el cual se describe detalladamente en la sección 4.2.2. El desempeño del controlador mediante Redes de Petri se compara con el desempeño de un controlador PID ISA y un controlador difuso, a través de gráficas que muestran la variable controlada después de realizar un cambio de referencia. Adicionalmente se calcularon índices de desempeño de los controladores ante un cambio de referencia, tales como: tiempo de establecimiento (t_s), sobretiro (M_p) y tiempo de elevación (t_r). La figura 5-1 muestra la forma en que se puede calcular t_s, M_p y t_r. Todas las pruebas se realizaron con un tiempo de muestreo igual a 1 segundo.



t_r	Tiempo de elevación	Tiempo para alcanzar por primera vez el valor final
t_p	Tiempo de pico	Tiempo para alcanzar el primer pico
t_s	Tiempo de establecimiento	Tiempo para que la magnitud de las oscilaciones sea menor o igual que el 2% de Δy_∞
m_p	Sobretiro	$m_p = \frac{A}{\Delta y_\infty}$
r	Razón de decaimiento	$r = \frac{B}{A}$

Figura 5-1: Curva de variable controlada ante un cambio de referencia.

5.1.1. Procesos lineales.

Se realizaron pruebas con 4 procesos lineales, los cuales su valor de estado estable es igual a 20, y sus funciones de transferencia son las siguientes:

- *Proceso 1.*

$$\frac{2}{20s+1}e^{-2s} \tag{5.1}$$

- *Proceso 2.*

$$\frac{4}{20s+1}e^{-2s} \tag{5.2}$$

- *Proceso 3.*

$$\frac{2}{60s+1}e^{-2s} \tag{5.3}$$

- *Proceso 4.*

$$\frac{2}{20s+1}e^{-4s} \tag{5.4}$$

Los controladores PID y difuso se sintonizaron únicamente para el proceso uno, ya que lo que se desea, es mostrar la robustez de cada uno de ellos ante cambios de los parámetros del proceso. En los procesos 2, 3 4 se vario únicamente un parámetro del proceso 1 para poder observar de que manera afecta la variación de cada parámetro en los controladores. En el proceso 2 se cambió la ganancia, en el proceso 3 sτ, y en el proceso 4 el tiempo muerto. El controlador mediante Redes de Petri como ya se ha mencionado anteriormente no se tiene que sintonizar, al realizar un cambio de referencia se realiza un cálculo automático de θ, y una vez que se estabiliza la variable controlada después de un cambio de referencia se realiza un cálculo aproximado de *K (ganancia del proceso).* A continuación se presentan las gráficas donde se muestra el desempeño de los controladores para cada uno de los procesos.

- *Proceso 1.*

  En la figura 5-2 se muestra un cambio de referencia de 30 a 60. Donde;

  ___ Controlador mediante Redes de Petri.

  - - - Controlador difuso.

  -.-.- Controlador PID.

  En la figura 5-2 se puede observar que los tres controladores tienen un buen desempeño, lo cual se debe a que el controlador PID y difuso están sintonizados para ese proceso. La tabla 5.1 muestra los índices de desempeño para cada controlador.

	FKBC	PID	Petri Nets
T_s	48 seg.	45 seg.	45 seg.
T_r	27 seg.	45 seg.	24 seg.
M_p	0.69%	0%	0.85%

Tabla 5.1: Desempeño de controladores en proceso 1.

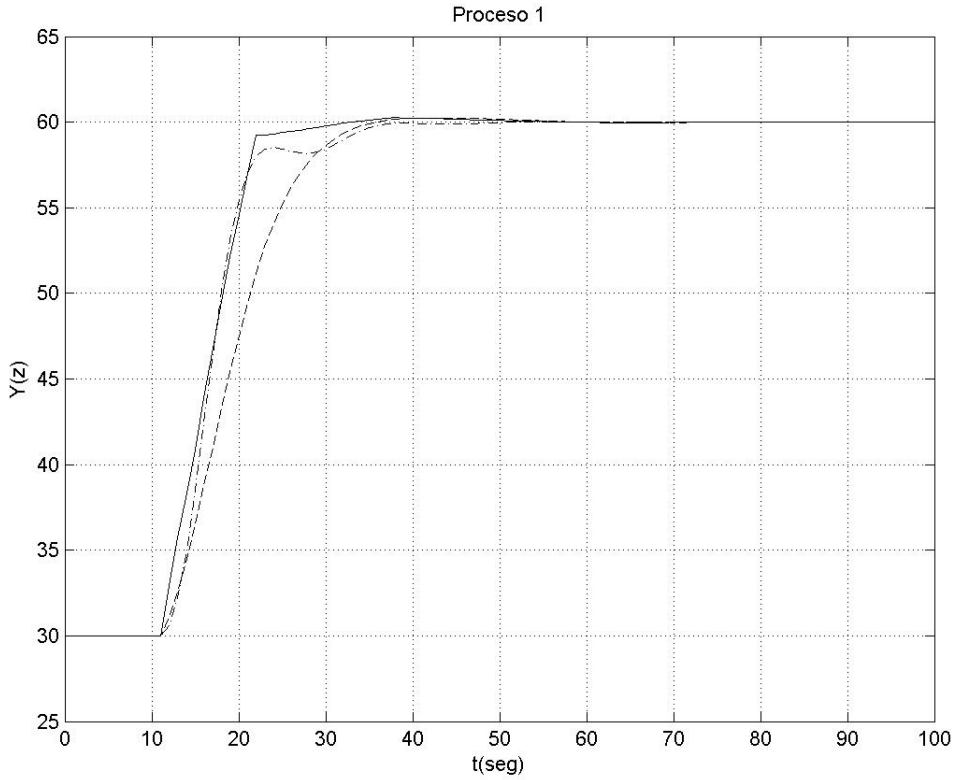


Figura 5-2: Cambio de referencia en proceso 1.

La figura 5-3 muestra como responde cada controlador ante perturbaciones. La magnitud de la perturbación que se simula es 10.

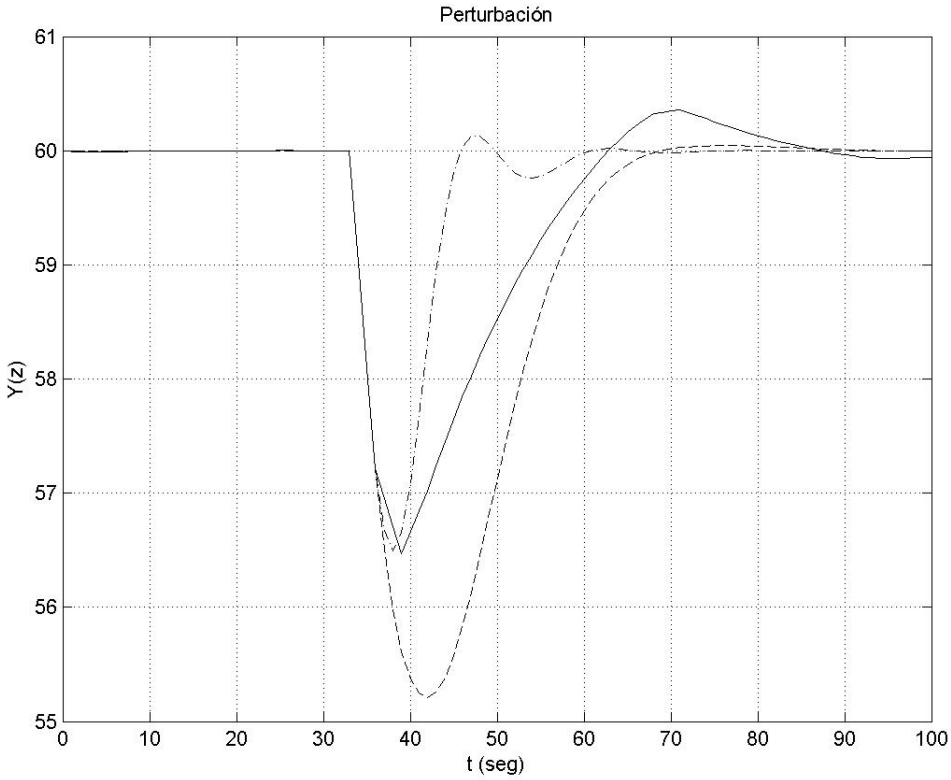


Figura 5-3: Control de perturbaciones.

___ Controlador mediante Redes de Petri.

- - - Controlador difuso.

-.-.- Controlador PID.

De la figura 5-3 se puede observar que el controlador PID es el que mejor corrige las perturbaciones.

- *Proceso 2.*

En la figura 5-4 se muestra un cambio de referencia de 30 a 60. Donde;

___ Controlador mediante Redes de Petri.

- - - Controlador difuso.

-.-.- Controlador PID.

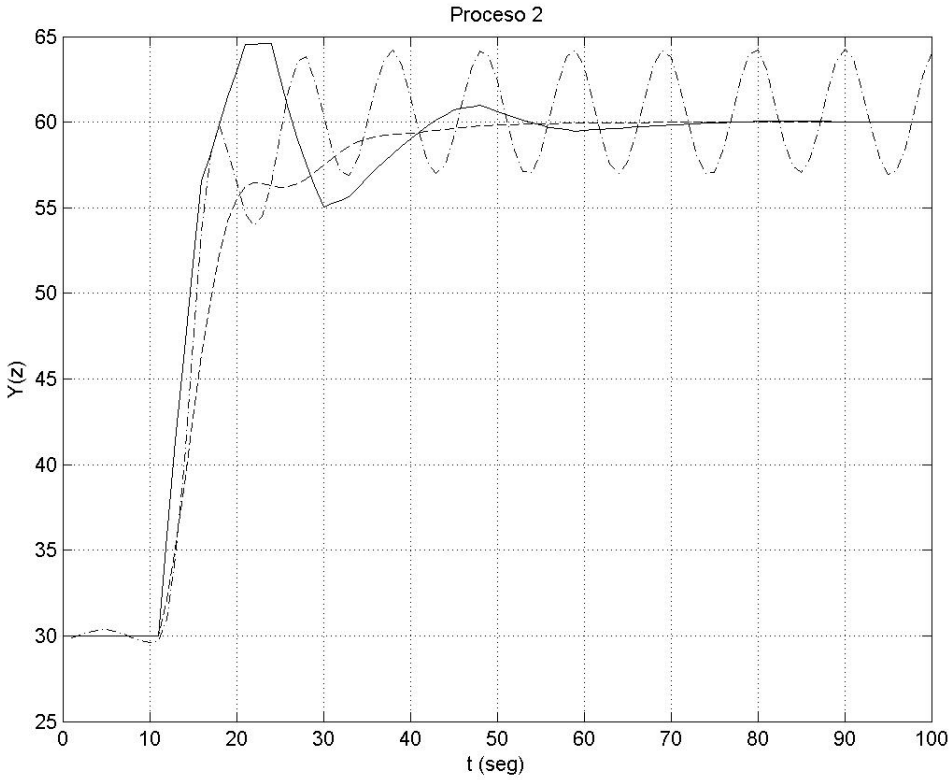


Figura 5-4: Cambio de referencia en proceso 2.

En la figura se puede observar que el controlador PID se volvió oscilatorio, que el controlador mediante Redes de Petri tiene mucho sobretiro y que el controlador difuso es el que tuvo mejor desempeño.

El controlador mediante Redes de Petri calcula cada vez que se realiza un cambio de referencia la ganancia del proceso, en este caso como la ganancia del proceso pasado era "2" su desempeño no fue optimo, pero si se continua con ese mismo proceso su desempeño mejorará considerablemente para los siguientes cambios de referencia. En la figura 5-5 se muestra como mejora el desempeño del controlador mediante Redes de Petri para ese mismo proceso en los siguientes cambios de referencia. En la tabla 5.2 se muestran los índices de desempeño de cada controlador.

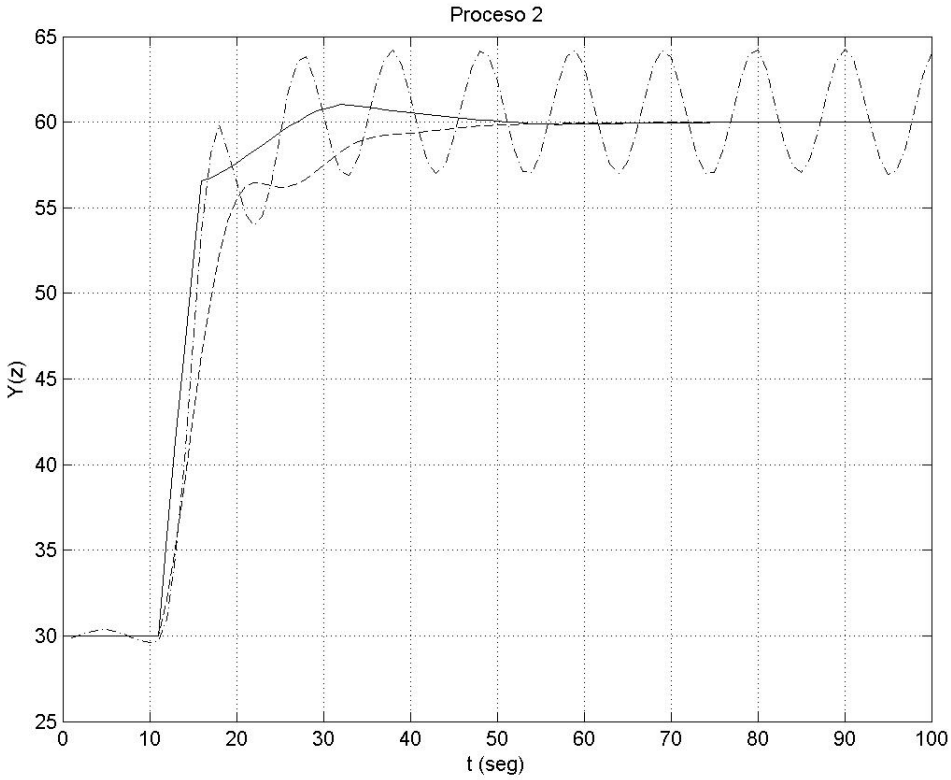


Figura 5-5: Autosintonización de controlador mediante Redes de Petri.

	FKBC	PID	Petri Nets 1	Petri Nets 2
T_s	55 seg.	oscilatorio	67 seg.	60 seg.
T_r	55 seg.	17 seg.	10 seg.	18 seg.
M_p	0.00%	oscilatorio	15.34%	3.47%

Tabla 5.2: Desempeño de controladores en proceso 2.

Con las pruebas que se realizaron con el proceso 2 se puede observar de que manera afecta a los controladores variaciones en la ganancia del proceso, ya que la ganancia de este proceso es el doble de la ganancia del proceso para el que se sintonizó el controlador PID y FKBC, y tanto θ como τ se mantuvieron con el mismo valor.

- *Proceso 3.*

  En la figura 5-6 se muestra un cambio de referencia de 30 a 60. Donde;

  ___ Controlador mediante Redes de Petri.

  - - - Controlador difuso.

  -.-.- Controlador PID.

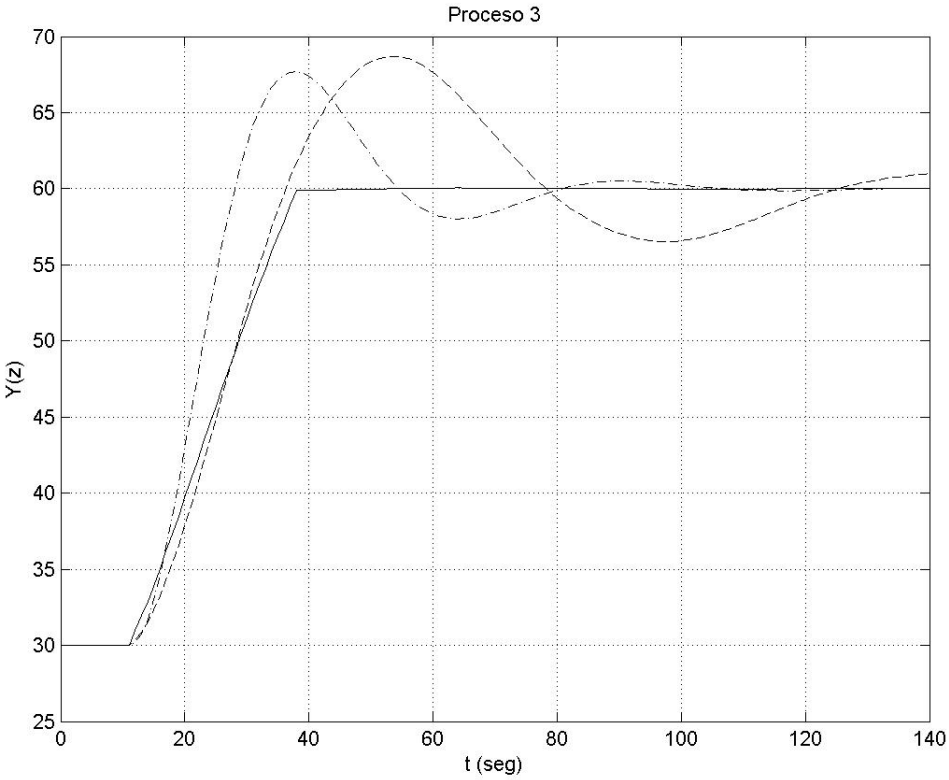


Figura 5-6: Cambio de referencia en proceso 3.

De la figura 5-6 se puede observar que el incremento de τ afectó considerablemente el desempeño del controlador difuso y en menor proporción al controlador PID. Por otra parte, también se puede notar que el desempeño del controlador mediante Redes de Petri no resultó afectado.

	FKBC	PID	Petri Nets
T_s	oscilatorio	121 seg.	62 seg.
T_r	28 seg.	20 seg.	44 seg.
M_p	oscilatorio	26%	0.18%

Tabla 5.3: Desempeño de controladores en proceso 3.

- *Proceso 4.*

Al igual que en los proceso anteriores se realizó una prueba con los tres controladores para un cambio de referencia de 30 a 60. El proceso 4 es igual al proceso 1 únicamente se incremento su tiempo muerto de 2 a 4 segundos.

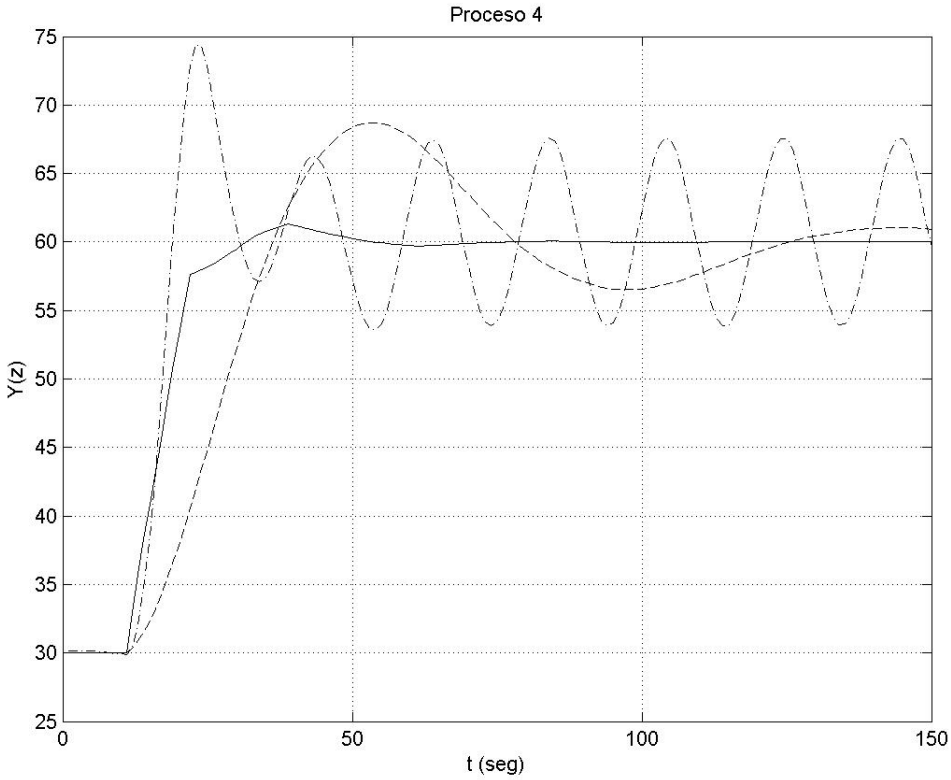


Figura 5-7: Cambio de referencia en proceso 4.

___ Controlador mediante Redes de Petri.

- - - Controlador difuso.

-.-.- Controlador PID.

El incremento de θ ocasionó que el controlador PID no puediera controlar el proceso, y que el controlador difuso tarde mucho tiempo en controlarlo. De igual forma que en el proceso anterior el mejor desempeño lo tuvo el controlador mediante Redes de Petri.

	FKBC	PID	Petri Nets
T_s	246 seg.	oscilatorio	80 seg.
T_r	30 seg.	12 seg.	25 seg.
M_p	28.97%	oscilatorio	4.39%

Tabla 5.4: Desempeño de controladores en proceso 4.

5.1.2. Proceso no lineal.

De igual forma que en los procesos lineales, se realizaron pruebas con un proceso no lineal (descrito en la sección 4.2.2.) con los tres controladores, y se comparó su desempeño. Cabe mencionar que los controladores PID y difuso están sintonizados para el proceso no lineal

aproximado a un sistema lineal de primer orden para una prueba escalón de 10% a 15%, cuya función de transferencia es la siguiente:

$$\frac{3.856}{34s+1}$$

En la figura 5-8 se muestra un cambio de referencia de 10 a 30, donde;

___ Controlador mediante Redes de Petri.

- - - Controlador difuso.

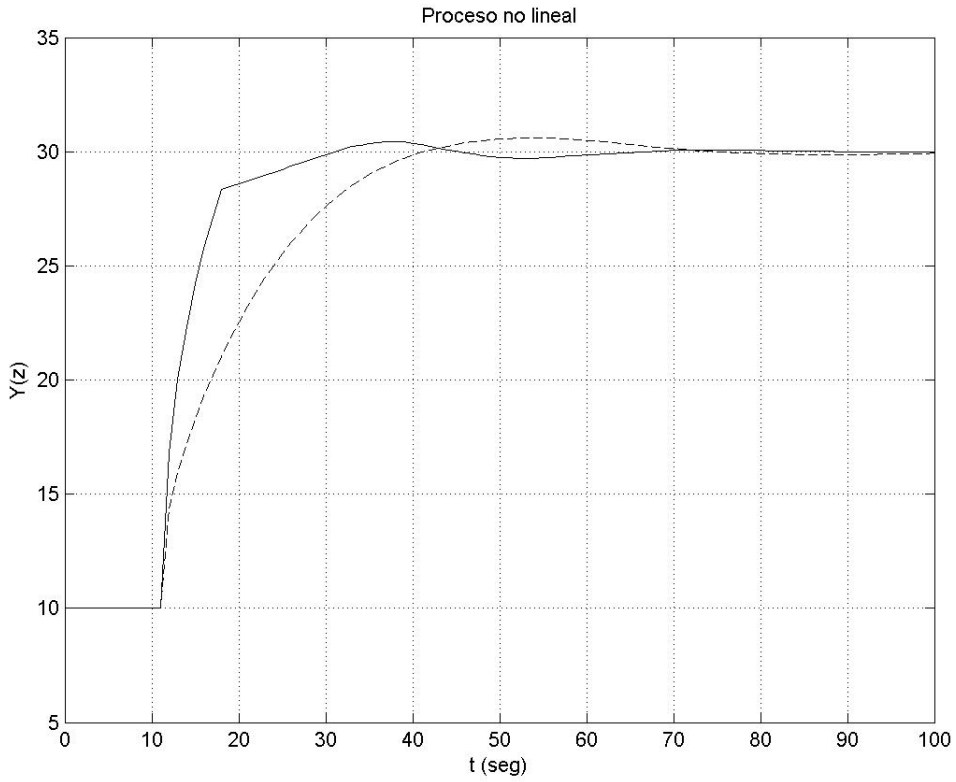


Figura 5-8: Cambio de referencia uno en proceso no lineal.

El controlador PID no se muestra porque en cuanto se accionaba el control automático utilizando este controlador la manipulación se volvía oscilatoria, y por lo tanto la variable controlada se inestabilizaba.

La tabla 5.5 muestra los índices de desempeño para cada controlador.

	FKBC	Petri Nets
t_s	92 seg.	54 seg.
t_r	31 seg.	21 seg.
M_p	3.03%	2.23%

Tabla 5.5: Desempeño de controladores en proceso no lineal en cambio de referencia 1.

Es importante destacar que por ser un proceso no lineal, el comportamiento del controlador difuso no es igual cuando se trabaja en referencias más altas. Entre más arriba este el nivel de agua, más lento se vuelve el proceso. A continuación se presentan los resultados obtenidos en otros cambios de referencia.

En la figura 5-9 se muestra un cambio de referencia de 30 a 60, donde;

___ Controlador mediante Redes de Petri.

- - - Controlador difuso.

-.-.- Controlador PID.

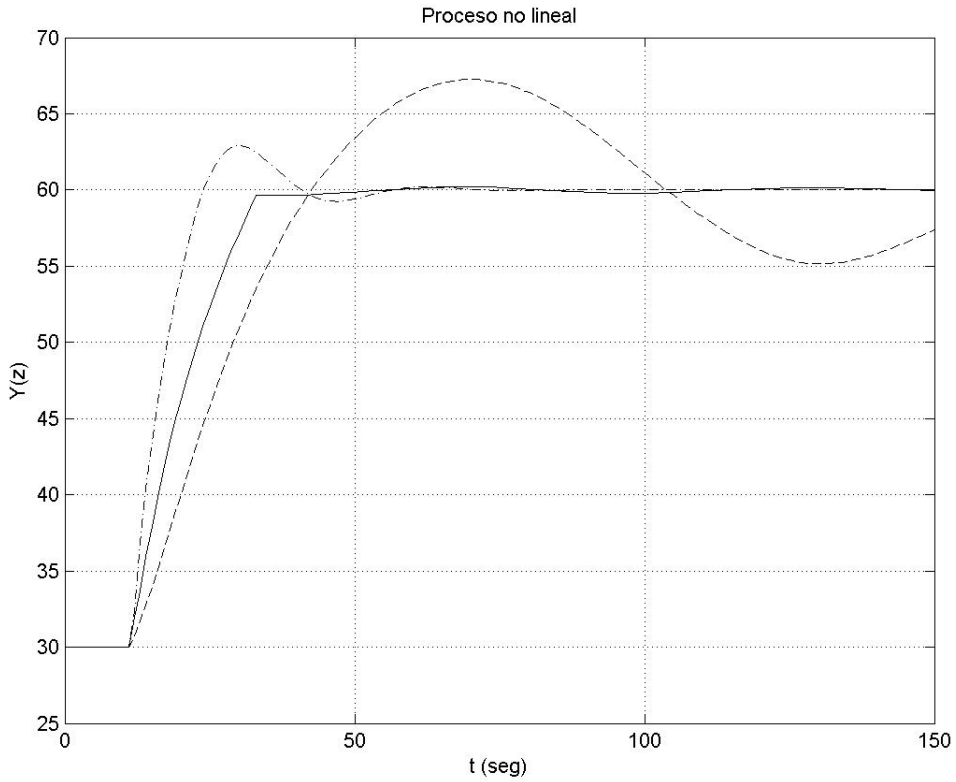


Figura 5-9: Cambio de referencia dos en proceso no lineal.

La tabla 5.6 muestra los índices de desempeño para cada controlador.

	FKBC	PID	Petri Nets
T_s	oscilatorio	85 seg.	77 seg.
T_r	32 seg.	15 seg.	45 seg.
M_p	oscilatorio	9.7%	0.72%

Tabla 5.6: Desempeño de controladores en proceso no lineal en cambio de referencia 2.

En la figura 5-10 se muestra un cambio de referencia de 60 a 80, donde;

___ Controlador mediante Redes de Petri.

- - - Controlador difuso.

-.-.- Controlador PID.

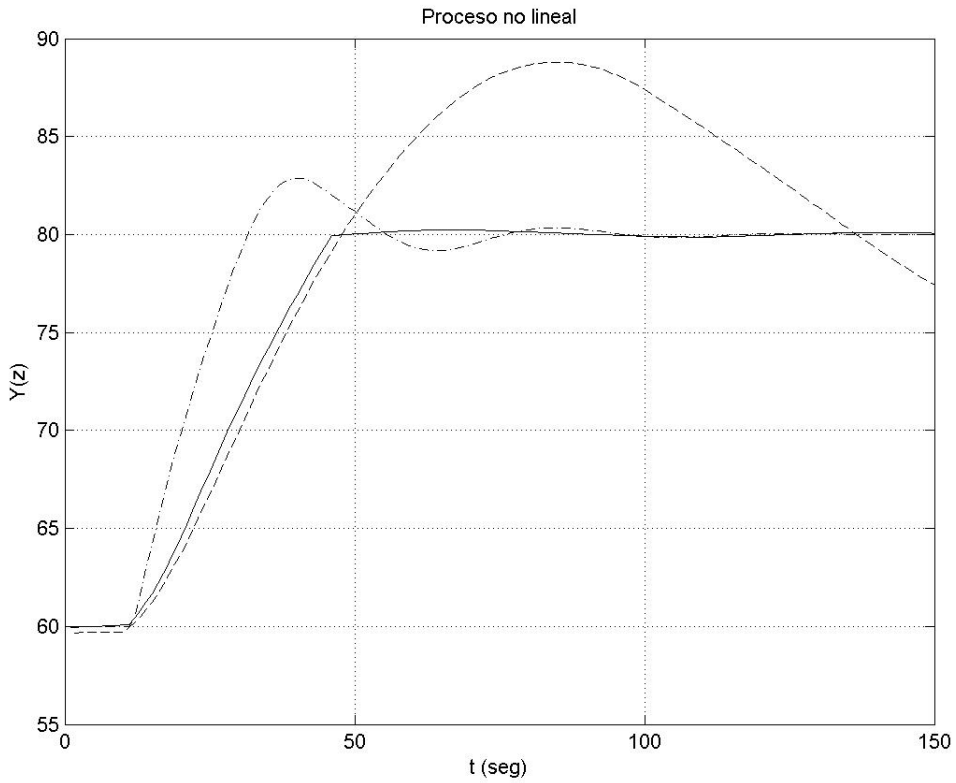


Figura 5-10: Cambio de referencia tres en proceso no lineal.

La tabla 5.7 muestra los índices de desempeño para cada controlador.

	FKBC	PID	Petri Nets
T_s	oscilatorio	115 seg.	83 seg.
T_r	37 seg.	22 seg.	38 seg.
M_p	oscilatorio	14.38%	1.16%

Tabla 5.7: Desempeño de controladores en proceso no lineal en cambio de referencia 3.

5.2. Tiempo real.

Para realizar pruebas en tiempo real, se utilizó la estación descrita en la sección 4.3. Dicha estación de trabajo cuenta con 4 velocidades, de los cuales se eligieron la uno y tres para realizar pruebas. Debido a que (de igual forma que en simulación) se buscaba comparar el desempeño del controlador mediante Redes de Petri con el controlador PID y difuso, se eligió la velocidad uno para sintonizar los controladores PID y difuso. La función de transferencia de la estación en la velocidad uno es:

$$\frac{1.3}{21s+1}e^{-3.5s} \qquad (5.5)$$

Las pruebas que a continuación se describen se realizaron con $T = 1$ *seg*. La primer prueba que se realizó fue con la velocidad 1 de la estación, la cual fue un cambio de referencia de 30 a 60° C.

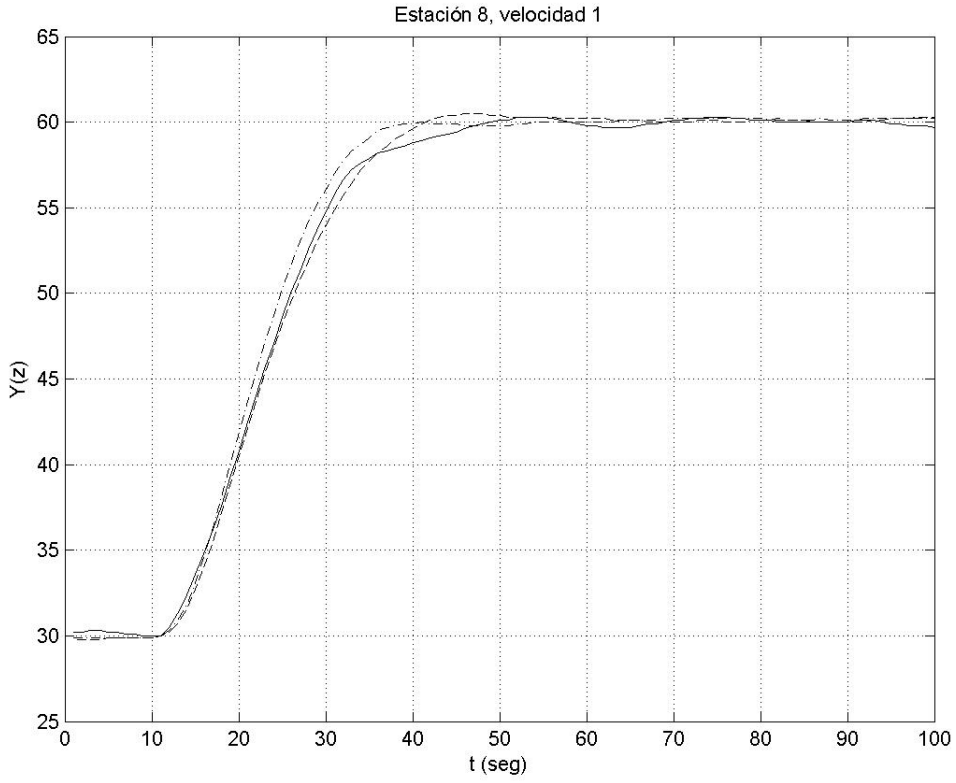


Figura 5-11: Cambio de referencia en estación 8 velocidad 1.

___ Controlador mediante Redes de Petri.

- - - Controlador difuso.

-.-.- Controlador PID.

	FKBC	PID	Petri Nets
T_s	54 seg.	38 seg.	48 seg.
T_r	34 seg.	33 seg.	41 seg.
M_p	1.66%	0.66%	1.00%

Tabla 5.8: Desempeño de controladores en estación 8 velocidad 1.

Se realizó otra prueba en tiempo real, la cual fue en la misma estación, pero ahora en la velocidad 3. Se prevé que los controladores PID y difuso no tenga la misma eficiencia en su desempeño que en la prueba pasada, ya que éstos están sintonizados para la velocidad 1 de la estación. Por otra parte debido a que la ganancia del proceso a la velocidad 1 y 3 no es la misma, el desempeño del controlador mediante Redes de Petri resulta afectado, pero nada más para el primer cambio de referencia que se realiza. En la figura 5-12 se muestra el funcionamiento del controlado mediante Redes de Petri después de que ya cálculo la nueva ganancia del proceso.

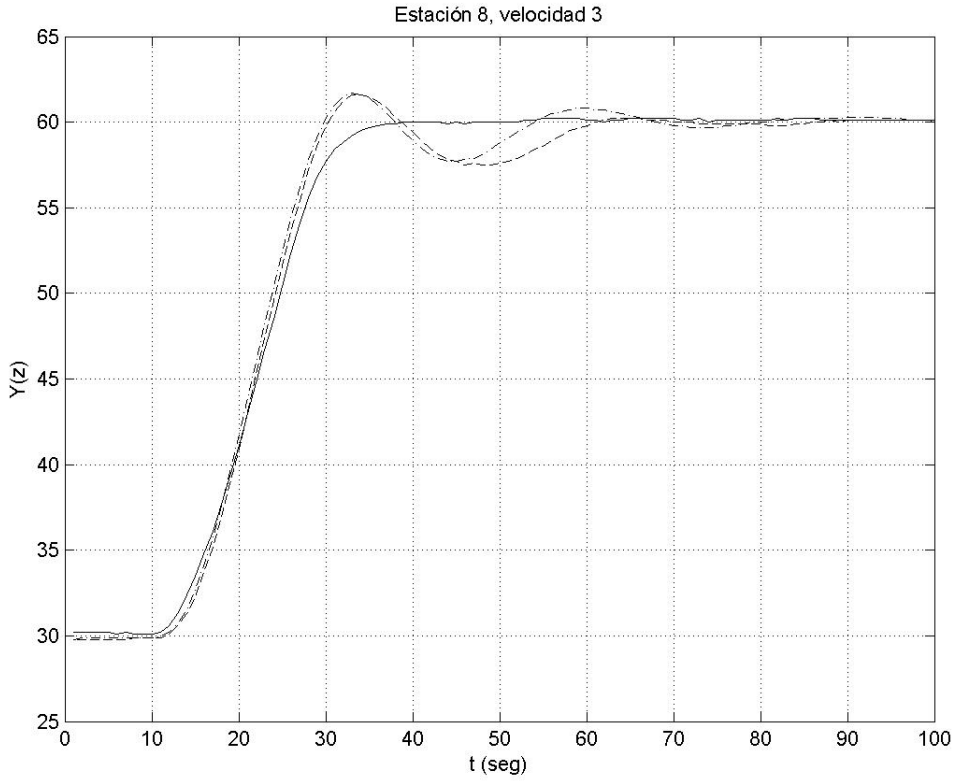


Figura 5-12: Cambio de referencia en estación 8 velocidad 3.

___ Controlador mediante Redes de Petri.

- - - Controlador difuso.

-.-.- Controlador PID.

	FKBC	PID	Petri Nets
T_s	78 seg.	88 seg.	51 seg.
T_r	23 seg.	22 seg.	31 seg.
M_p	5.33%	5.66%	0.66%

Tabla 5.9: Desempeño de controladores en estación 8 velocidad 3.

Capítulo 6

Conclusiones

6.1. Conclusiones de las pruebas realizadas.

Con las pruebas realizadas se pudo constatar que el controlador mediante Redes Petri presenta un mejor desempeño que los controladores PID y difusos, cuando se varían los parámetros del proceso a controlar. Una de las grandes ventajas que presenta este controlador, es que el usuario no necesita un modelo matemático del proceso ni introducir coeficientes para que el controlador tenga un buen desempeño, ya que en ciertas ocasiones es difícil modelar matemáticamente un proceso (lo cual normalmente se requiere para la sintonización de los controladores PID´s) o realizar pruebas para sintonizar el controlador (necesario en los controladores difusos).

En las pruebas realizadas además se pudo observar que los controladores PID son sumamente sensibles a variaciones en la ganancia del proceso y a que se incremente su tiempo muerto, por su parte los controladores difusos son menos sensibles en esos parámetros que los controladores PID, pero más sensibles a variaciones de τ. Al controlador mediante Redes de Petri también le afectan las variaciones del tiempo muerto y ganancia, pero el tiempo muerto se calcula automáticamente al iniciar el programa y realizar un cambio de referencia, y la ganancia se aproxima una vez que se estabiliza el proceso después de un cambio de referencia. Con este ajuste automático se logra que el controlador de Petri se pueda utilizar en distintos procesos, y si su ganancia es distinta, únicamente en el primer cambio de referencia no se presentará un desempeño adecuado, el cual puede ser que presente mucho sobretiro si la ganancia del proceso anterior es menor a la del proceso actual, o tiempo de elevación muy grande si es el proceso anterior tenía una ganancia mayor. Las variaciones de τ le afectan menos de lo que le afectan a un controlador PID o difuso. Para procesos con τ muy pequeña es necesario únicamente disminuir el tiempo de muestreo.

En las pruebas realizadas en tiempo real se pudo observar que el comportamiento de la estación "8" del laboratorio de control presenta no linealidades, las cuales pudieron ser controladas por los tres controladores PID, difuso y de Petri.

6.2. Trabajos futuros.

Debido a que el controlador mediante Redes de Petri está compuesto por reglas del tipo *if-else*, fácilmente se le pueden agregar nuevas reglas para trata de mejorar su desempeño, siguiendo la metodología mostrada en el capítulo 3. En trabajos futuros se podría agregar:

- Más pruebas en procesos no lineales simulados, ya que en el presente trabajo únicamente se realizaron pruebas con un solo proceso de este tipo.
- Realizar pruebas para mostrar rangos de operación del controlador mediante Redes de Petri, es decir, cuanta variación de los parámetros (τ, θ, y K_p) de un proceso puede controlar.
- Un aspecto en el que es superado el controlador mediante Redes de Petri por controladores PID (sintonizados para perturbaciones), es el control de perturbaciones, por lo tanto trabajos futuros podrían enfocarse a mejorar su desempeño ante perturbaciones.
- Pruebas en otros procesos reales, ya que los procesos reales presentan comportamientos no lineales que son difíciles de simular. Y el realizar pruebas con otros procesos reales ayudaría a identificar comportamientos que no se han tomado en cuento en el diseño del controlador.
- Verificar la estabilidad del controlador de Petri haciendo uso de la teoría de Redes de Petri.
- Encontrar un controlador de Redes de Petri optimo, utilizando las reglas de reducción de estados y análisis de marcaje.
- La validación teórica total del controlador propuesto.

Bibliografía

[1] Aström K. J. and B. Wittenmark. *Computer Controlled Systems Theory and Design*. Third edition, Prentice Hall 1997.

[2] David R. and H. Alla. *Petri Nets and Grafcet Tools for Modelling Discrete Event Systems*. First edition, Prentice Hall 1992.

[3] Driankov D., H. Hellendoorn and M. Reinfrank. *An Introduction to Fuzzy Control*. Second edition, Springer, 1996.

[4] Groover M. *Automation, Production Systems, and Computer Integrated Manufacturing*. Second edition, Prentice Hall 2001.

[5] Kuo B. *Sistemas de Control Automático*. Séptima edición, Prentice Hall 1996.

[6] Ogata K. *Discrete Time Control Systems*. Second edition, Prentice Hall 1995.

[7] Ogata K. *Ingeniería de Control Moderna*. Cuarta edición, Prentice Hall 2003.

[8] Rosas L. *Metodología para la Automatización del Diseño de Controladores RST Robustos*. Tesis ITESM Campus Monterrey 2002.

[9] Schmelkes C. *Manual para la Presentación de Anteproyectos e Informes de Investigación*. Segunda edición. Oxford 2003.

[10] Wang L. *A Course in Fuzzy Systems and Control*. Prentice Hall, 1997.

[11] Zill Dennis. *Ecuaciones Diferenciales con Aplicaciones de Modelado*. Sexta edición, International Thomson Editores 1999.