CONTROL EN CASCADA DE UN PROCESO INDUSTRIAL EN TIEMPO REAL. CASO DE ESTUDIO: PLANTA DE TANQUES INTERACTUANTES

ERMILSO DIAZ BENACHI
YONNY FERNANDO CABEZAS ORTIZ

Monografía de trabajo de grado

Director
I. E. Juan Fernando Flórez Marulanda

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE ELECTRÓNICA, INSTRUMENTACIÓN Y CONTROL
POPAYÁN
2008

CONTROL EN CASCADA DE UN PROCESO INDUSTRIAL EN TIEMPO REAL. CASO DE ESTUDIO: PLANTA DE TANQUES INTERACTUANTES

Monografía presentada como requisito parcial para optar por el título de Ingenieros en Automática Industrial

ERMILSO DIAZ BENACHI

YONNY FERNANDO CABEZAS ORTIZ

Director
Juan Fernando Flórez Marulanda
Ingeniero

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE ELECTRÓNICA, INSTRUMENTACIÓN Y CONTROL
POPAYÁN
2008

Nota de Aceptación

_____

_____

_____

Director _____
I.E. Juan Fernando Flórez Marulanda

Jurado _____
Dr. Carlos Alberto Gaviria López

Jurado _____
Mag. Héctor Fabio Jaramillo Ordoñez

Fecha de sustentación: Popayán, Noviembre de 2008

AGRADECIMIENTOS

Los autores del presente trabajo, manifiestan sus agradecimientos a su director, I.E. Juan Fernando Flórez Marulanda, al grupo de Aplicación de Tecnologías Inteligentes (ATI), al Ingeniero Víctor Alfaro docente de la Universidad de Costa Rica, al Ing. Roberto Buncher de la SUPSI, al Ing. Paolo Mantegazza del DIAMP, a la Universidad del Cauca, profesores, amigos y compañeros, quienes contribuyeron con el desarrollo de este trabajo.

“No es verdaderamente exitoso quien no es feliz; lo importante no es alcanzar determinada cumbre señalada por los demás, sino disfrutar el viaje al destino preferido por cada cual”

Albert Einstein

Esto no hubiera sido posible sin ustedes:

A mis padres, Alba y Wilibardo,
A mis hermanas Mabel, Claudia y Anita
A Carol por estar a mi lado
A mi familia, amigos, profesores,
A la Universidad del Cauca

YONNY

“La grandeza de un hombre no se mide por el terreno que ocupan sus pies, sino por el horizonte que descubren sus ojos”

José Martí

Dedico este triunfo a las personas que amo:

A mi mamá por brindarme su amor que no tiene límites,
A don Eliecer por la confianza que siempre ha depositado en mí,
A mi hermano Fabián, por su amistad y cariño,
A Mónica por su apoyo y compresión.

ERMILSO

CONTENIDO

Pág.

LISTA DE TABLAS

LISTA DE ILUSTRACIONES

LISTA DE ANEXOS

RESUMEN

En este documento se describe el desarrollo de un sistema de control en cascada utilizando un sistema operativo de tiempo real soportado sobre una plataforma de cómputo convencional y aplicado a un caso de estudio, la planta de tanques interactuantes del Laboratorio de Control de Procesos del Programa Ingeniería en Automática Industrial.

Con el fin de fijar las bases del sistema de control en cascada en el capítulo 1 se describen dos de los esquemas de control más comunes, en el capítulo 2 se da un énfasis a la ley de control PID (Proporcional-Integral-Derivativo), en el capítulo 3 se expone los conceptos más relevantes de los sistemas de control en cascada. En el capítulo 4 se explican las cualidades de los sistemas de tiempo real y de los sistemas operativos con características de tiempo real (SOTR en adelante) con el fin de generar bases de conocimiento para la elección de uno de ellos en el que se pueda implementar el sistema de control. En el capítulo 5 se describe la selección del software elegido, el diseño y la implementación del sistema de control, en un sistema de tiempo real para la planta de tanques interactuantes. En el capítulo 6 se presenta la integración del sistema y se muestran los resultados obtenidos al probar el sistema de control. Finalmente se consignan las experiencias sobresalientes del desarrollo del proyecto y las expectativas para trabajos futuros.

Palabras claves: Control en cascada, tiempo real, PID Industrial, SOTR.

INTRODUCCIÓN

La estabilidad de las variables de proceso en un sistema de control y sus valores correctos contribuyen a la optimización de la energía y obtener un mayor rendimiento. Un acertado esquema de control automático de esas variables se convierte, sin duda, en uno de los factores fundamentales en la consecución de tales beneficios. Un complemento ideal al control automático del proceso es el monitoreo y supervisión de las señales de dicha planta, de tal forma que se cuente con la información de proceso en el momento que sea requerida.

En cuanto a sistemas de control y plantas industriales el programa de Ingeniería en Automática Industrial de la Universidad del Cauca cuenta con un Laboratorio de Control de Procesos, en este laboratorio los estudiantes del programa realizan prácticas de control de velocidad y posición de motores de DC, de control de presión, de control de nivel y caudal, de control de temperatura y control de nivel por PLC (Controlador Lógico Programable). El propósito de estas prácticas es que los estudiantes identifiquen las características dinámicas de cada uno de los procesos, configuren y sintonicen estructuras de control PID y analicen su comportamiento. Sin embargo para lograr lo anterior no todas las plantas cuentan con el soporte computacional adecuado para facilitar la adquisición y análisis de los datos en línea. La Planta de control de nivel, que es el caso de estudio del presente proyecto, cuenta con la herramienta computacional *Labview* para la adquisición de datos, sin embargo este programa exige recursos que obligan a que el computador (PC) sea de buenas prestaciones, por otra parte las interfaces hombre-máquina (HMI) construidas en él presentan dos serias limitaciones: primero el tiempo desplegado en las gráficas del nivel, flujo y esfuerzo de control en el actuador no es uniforme, segundo no cuenta con la facilidad de capturar la curva desplegada y analizarla en línea, condición muy útil cuando se desea sintonizar un control PID a partir de la respuesta al escalón de la planta. Estas particularidades propias de la actual herramienta en *Labview*, hacen que el aprendizaje en la práctica no se vea facilitado con el uso de estos recursos software.

Por los motivos expuestos anteriormente, se implementa un sistema que permita al usuario enfrentarse con diferentes esquemas de control, adicionalmente por medio de interfaces sencillas permitirle a éste interactuar con las señales y variables del proceso. Es por eso que el presente proyecto, *Control en Cascada de un proceso industrial en tiempo real, caso de estudio Planta de Tanques Interactuantes,* presenta una solución a los inconvenientes encontrados anteriormente.

El sistema combina la implementación de un sistema de control en tiempo real soportado en una plataforma de cómputo convencional, empleando estrategias PID de realización serie y paralela de tipo industrial que incluyan técnicas para evitar fenómenos como el *windup* y *bump transfer,* principalmente aplicando un esquema de control en cascada para el caso de estudio. De esta manera el estudiante puede experimentar con diferentes realizaciones del control PID y entender algunos de los fenómenos que se presentan en él, adicionalmente que el estudiante se relacione con esquemas de control alternativos al *feedback,* como lo es el control en cascada dentro del mismo proceso. Una de las ventajas de la implementación, es que el estudiante podrá realizar medición en línea de las variables y señales del proceso, de esta manera tomar datos de tiempo y amplitud de las variables, lo que permitirá hacia futuro hacer identificación en línea del proceso.

Este proyecto reúne diversos elementos tecnológicos bastante innovadores como el diseño de un sistema de control en PC aplicado a un proceso industrial bajo un sistema operativo de tiempo real y el diseño de un controlador PID industrial con estrategia AWBT (*Antiwindup – Bumpless Transfer*) de estructura tanto serie como paralelo, sin embargo la principal contribución del trabajo consiste en el estudio de un esquema de control en cascada aplicado a un proceso industrial y, en forma adicional, el análisis de los mecanismos formales de sintonización PID que se aplican bajo este enfoque.

1. ESQUEMAS DE CONTROL EN LOS PROCESOS INDUSTRIALES

El objeto de todo proceso será la obtención de un producto final, con unas características determinadas de forma que cumpla con las especificaciones y niveles de calidad del mercado, que es cada día más exigente. Esta constancia en las propiedades del producto sólo será posible gracias a un control exhaustivo de las condiciones de operación, ya que tanto la alimentación al proceso como las condiciones del entorno varían en el tiempo. Es por eso que existen esquemas de control, los cuales permiten que las operaciones de los procesos sean más fiables y a la vez presenten características deseables entre las que se destacan:

- Mantener el sistema estable, independiente de disturbios y desajustes.
- Conseguir las condiciones de operación de forma rápida y continua.
- Trabajar correctamente bajo un conjunto de condiciones operativas.
- Manejar las restricciones de equipo y proceso de forma precisa.

La implementación de un adecuado esquema de control, que se adapte a las necesidades del sistema, significará una sensible mejora de la operación. Algunos de los beneficios obtenidos serán: incremento de la productividad, mejora de los rendimientos y calidad, control medioambiental, seguridad operativa, optimización de la operación del proceso y utilización del equipo, ahorro de energía y fácil acceso a los datos del proceso (Mavainsa, 2006).

Los esquemas de control son aplicables a cualquier tipo de proceso, pueden ser basados en sistemas realimentados o *feedback,* en esquemas *feedforward*, híbridos (feedforward+feedback) por ejemplo cascada, de relación, relés de cómputo, control de razón hasta control avanzado, como se comenta en (Smith y Corripio, 1991).

A continuación se explican dos de los más representativos y usados en el ambiente industrial actualmente.

1.1. CONTROL POR RETROALIMENTACIÓN O FEEDBACK

Es uno de los esquemas más usados en la industria de procesos, esta técnica fue utilizada industrialmente por primera vez, por *James Watt* hace aproximadamente 200 años, para regular la velocidad de una máquina de vapor con carga variable. En este tipo de control se toma la variable controlada y se retroalimenta al controlador para que este pueda tomar una decisión. La operación del sistema de control por retroalimentación es básicamente una operación de ensayo y error, es decir, cuando el controlador detecta que la variable controlada aumenta por arriba del punto de control, indica al actuador que realice una tarea, pero éste cumple con la orden más allá de lo necesario, en consecuencia la variable controlada desciende por abajo del valor deseado; al notar esto, el controlador envía nuevamente una señal al actuador para corregir este descenso. El ensayo y error continúa hasta que la variable controlada alcanza el valor de referencia. El control por retroalimentación es una técnica bastante simple que compensa las perturbaciones, claro está, una vez los disturbios se propaguen en el proceso y se manifiesten en un cambio en la variable controlada. Cualquier disturbio puede perturbar a la variable controlada desviándola del punto de control, el controlador para compensar dicha perturbación cambia su salida con el fin de tratar de mantener a la variable controlada (VC) en el punto de control (Smith y Corripio, 1991).

La desventaja del control por realimentación es que no detecta que tipo de disturbio entra al proceso, únicamente puede compensar la perturbación que se genere hasta que la variable controlada se desvía del punto de control, es decir el disturbio se debe propagar dentro de todo el sistema antes de que lo pueda compensar el control por realimentación.

1.2. CONTROL POR PRE-ALIMENTACIÓN O FEEDFORWARD

En el control *feedback* tradicional el controlador corrige desviaciones en el punto de operación solamente hasta que la perturbación en la VC afecta la operación del proceso. Para mejorar el desempeño existe la posibilidad de medir los disturbios que ingresan al proceso, de tal manera que el controlador actúe sobre la planta aún antes de que dichas disturbios alejen la VC del valor de consigna, el disturbio medido se alimenta a un

controlador denominado *feedforward* o de pre-alimentación el cual genera una acción de control para tratar de mantener a la variable controlada cerca de la consigna. Para obtener resultados eficientes con este tipo de control se deben realizar una adecuada selección de las variables más influyentes en la dinámica del proceso para ser controladas (Smith y Corripio, 1991).

Entre las desventajas de este esquema de control es que se pueden presentar efectos indeseables por errores de modelamiento del proceso sobre el desempeño del controlador. Estos errores surgen debido a que, para diseñar la función de transferencia del control *feedforward*, se requiere conocer los modelos matemáticos, o funciones de transferencia del proceso a controlar y de los disturbios. Adicionalmente este esquema de control no utiliza información de la variable controlada, es decir, no existe retroalimentación hacia el controlador del valor de dicha variable. Otra de las desventajas de esta estrategia es que requiere una mayor inversión en equipos y en mano de obra necesarios para su diseño, implementación y mantenimiento que el control por retroalimentación.

De aquí en adelante se abordan dos de las más importantes y significativas tecnologías dentro del área de control de procesos en la industria, en la academia y, en especial, en lo referente con el alcance de este proyecto, como: el control PID Industrial y el control en cascada como técnica mejorada de control.

2. CONTROL PID

A continuación se realiza una breve descripción de la evolución del control PID basada en el proyecto de fin de carrera de Rodrigo Rojas, estudiante de la Universidad de Costa Rica bajo la dirección del profesor Víctor Alfaro en el año 2002 (Rojas, 2007).

Las estrategias de control PID se incorporan en el ambiente industrial en el primer cuarto de este siglo, con un enfoque de sintonización completamente empírico. El primer controlador, solo proporcional, fue el Modelo 56R de Taylor, el cual fue comercializado en 1933 y era conocido como Fulscope. En los años 1934 y 1935, Foxboro empezó la comercialización del Modelo 40, reconocido como el primer controlador PI y usado en el control de flujo en la industria petrolera. En el año de 1934, Taylor realizó una modificación sobre el Modelo 56R en el cual se incorporó el modo derivativo. El primer controlador que incorporó las tres acciones básicas de control mencionadas, fue el Modelo 100 Fulscope introducido al mercado por Taylor Instruments en 1940 (Blickley, 1991). La diferencia entre la forma en que este controlador fue realizado y la ecuación generalmente utilizada para la descripción de las acciones de un controlador PID originó, desde la creación de éste, múltiples y diversas formas de referirse a él (Alfaro, 2002a).

Durante la década de los años 50, empresas como *Foxboro, Taylor Instrument, Honeywell, Leeds & Northrup, Manning, Maxwell & Moore* y *Swartwout* incursionaron en la comercialización de controladores electrónicos. El que se considera como el primer controlador PID electrónico, fue comercializado por *Swartwout Co*. con el nombre de AutroniC en 1951. Estos equipos electrónicos resultaron capaces de realizar todas las funciones que previamente solo eran posibles con aparatos neumáticos, además de incluir funciones como suma, multiplicación, raíz cuadrada y otras operaciones matemáticas. De todos modos existía una desconfianza en el uso de tubos al vacío hasta que aparecieron los controladores de estado sólido, como el *Consotrol* de *Foxboro* en 1959. *Honeywell* introdujo en 1959 el uso de la señal eléctrica en el rango de 4 a 20 mA en corriente continua, el cual se convertiría en un estándar industrial años más tarde.

La década de los años 60 se caracteriza por el auge de los controladores lógicos programables (PLC). El primer PLC, el 084, fue fabricado por *Bedford Associates* (más adelante *Modicon*) por pedido de *General Motors*. Este PLC consistía en un equipo de resolución por lógica secuencial o "*Ladder Logic*". La ventaja de la incursión de los PLC a la industria fue la habilidad de programar el sistema, algo que no se podía lograr con paneles de relés electromecánicos. En 1975, *Honeywell* anunció su arquitectura de Control Distribuido Total (TDC). El *Yokogawa Centum* y el *Honeywell TDC2000* tienen como base el concepto de que varios lazos de control microprocesados podían ser controlados por un minicomputador supervisor y de esta manera introdujeron los primeros sistemas de control distribuido. En 1976 se introduce el primer controlador PID electrónico digital y en 1981, *Leeds & Northrup* empiezan la comercialización del primer controlador PID autosintonizable. *Honeywell* en 1982 introduce el *UDC500* basado en microprocesador, para el control de lazo y que incluía conectividad con una central de comando. En 1987, *Foxboro* populariza los controladores 760 y 761, los primeros en emplear tecnología en inteligencia artificial. Desde entonces, la evolución de los controladores ha sido orientada hacia la mejora de los mecanismos de auto sintonía y desarrollo de los programas para la simulación de lazos de control (Strothman, 1991).

En la actualidad, y pese al sorprendente desarrollo de la teoría de control y del soporte tecnológico necesario para su implementación, el controlador de estructura PID se emplea casi con exclusividad en el ambiente industrial de todo el mundo, donde cerca del 95 % de los lazos de control emplean un PID, en particular para controlar procesos térmicos y químicos (Heong et all, 2005).

El controlador PID, en donde sus términos proporcional, integral y derivativo se refieren a las acciones de control tomadas usualmente sobre el error, a pesar de ser el más usado a nivel industrial y académico, existen varias formas de referirse a él y no existe un algoritmo único para este controlador, ni siquiera una nomenclatura única para referirse a sus parámetros y características.

2.1. CARACTERÍSTICAS DEL CONTROL PID

Como su nombre lo indica el controlador PID es una técnica de control que se basa fundamentalmente en el control del lazo a través de la aplicación de tres señales que están establecidas por componentes dinámicos y estáticos.

La componente proporcional es una salida del controlador proporcional al error, presenta un camino directo en el lazo (sin retraso de fase propio) entre la señal de error y la acción de control. La componente integral tiene como función eliminar el error en estado estacionario, como inconveniente fundamental la acción integral pura presenta un efecto desestabilizador importante debido al retraso de fase de 90° que posee su función de transferencia, este efecto desestabilizador puede ser reducido si a la acción integral del controlador se le adiciona una acción proporcional. Si ante un disturbio o cambio del valor de consigna, la dinámica con que el sistema alcanza el nuevo estado estacionario no es la adecuada, puede incluirse una tercera acción correctora que es la acción derivativa dada por la componente derivativa, que es una manera de anticiparse al error futuro. El objetivo de esta última acción es hacer que la señal de control se incremente con la pendiente del error más que con su valor actual y/o la sumatoria de sus valores pasados. Este efecto de adelanto en el tiempo de la acción derivativa se traduce en un incremento de la estabilidad relativa del sistema. La sintonización del controlador se reduce entonces al ajuste de estas tres constantes, en muchas aplicaciones, estos parámetros no son suficientes para satisfacer en forma simultánea las especificaciones de rechazo a perturbaciones y de seguimiento de la consigna, de ahí que actualmente la mayoría de controladores PID presentan cambios importantes en su estructura (Alfaro, 2008).

2.2. ESTRUCTURAS DEL CONTROL PID

No existe una representación única del algoritmo PID debido a una falta de estandarización por parte de los fabricantes. Las funciones de transferencia de los controladores PID son alguna variante de las estructuras básicas y universales que son la estructura serie, y la estructura paralela (Yun Li, 2006).

Estructura PID Serie. También llamada Controlador Interactivo, como se puede ver en (1) el componente derivativo influye en la parte integral por tanto las partes están constantemente interactuando. Existe una razón histórica de la preferencia del controlador de estructura serie, ya que los primeros controladores neumáticos fueron más fáciles de construir usando esta forma, a pesar de los avances y cambios tecnológicos de los fabricantes dicha estructura se ha mantenido por razones de tipo comercial (Rojas, 2007).

$$G_s(s) = Kp\left(1 + \frac{1}{sT_i}\right)(1 + sT_d) \tag{1}$$

Donde: *Kp es la* Ganancia proporcional, T_i *es el tiempo* integral y T_d *es el* tiempo derivativo.

La forma serie es también llamada forma clásica. Esta representación tiene una interpretación interesante en el dominio de la frecuencia; los ceros corresponden a los valores inversos de los tiempos derivativo e integral. Todos los ceros del controlador son reales y las acciones integral o proporcional puras no pueden ser obtenidas con valores finitos de los parámetros del controlador (Alfaro, 2002b).

Estructura PID Paralela. También conocida como no interactuante porque el tiempo integral T_i no influye en la parte derivativa, así como el tiempo derivativo T_d no influye con la parte integral y de esa forma sus componentes no interactúan entre si, ver (2). También se conoce como forma estándar y algunas veces es llamado “algoritmo ideal o académico”, donde las acciones proporcional, integral y derivativa son no interactivas en el dominio del tiempo. Este algoritmo admite ceros complejos, lo que es útil cuando se controla sistemas con polos oscilatorios. La forma paralela es la más general, debido a que se pueden obtener acciones proporcional, integral y derivativa puras con parámetros finitos. El controlador puede también tener ceros complejos, siendo, por tanto, la forma más flexible. Sin embargo, es también la forma donde los parámetros tienen poca interpretación física (Alfaro, 2002b).

$$G_p(s) = Kp\left(1 + \frac{1}{sT_i} + sT_d\right) \tag{2}$$

Donde: *Kp es la* Ganancia proporcional, T_i *es el tiempo* integral y T_d *es el* tiempo derivativo.

2.3. VARIANTES DEL CONTROL PID

Como se mencionó anteriormente las estructuras serie y paralela son las más conocidas y puede ser encontradas en diversos paquetes software y módulos hardware; no obstante es muy común actualmente encontrar sus estructuras modificadas. Dentro de las características de desempeño más relevantes en cuanto al manejo de la señal de referencia y robustez debido al ruido, como se explica en (Yun Li et al, 2006), están:

Filtro pasa bajo componente derivativo. Tal como se muestra en (1) y (2) la componente derivativa corresponde a un diferenciador puro el cual amplifica los componentes de alta frecuencia de la señal entrante al controlador; esto es un problema ya que en el campo industrial, el ruido es un elemento inherente que siempre está presente afectando de esta manera el buen rendimiento del proceso. Cuando se presenta esta situación, la diferenciación resulta en una señal de control de amplitud teóricamente infinita. Para evitar este inconveniente se adiciona comúnmente un filtro pasa bajo de primer orden a la componente derivativa para limitar su ganancia en alta frecuencia; ante esta adición la componente derivativa es representada por (3). La diferencia en la representación es debido a la estructura de la componente derivativa en cada controlador.

$$sT_d \cong \frac{sT_d}{sT_d/_N + 1} \qquad sT_d \cong \frac{sT_d + 1}{sT_d/_N + 1} \qquad (3)$$

b)

Filtro Derivativo PID Paralelo Filtro Derivativo PID Serie

Donde *N* es una constante, la mayoría de fabricantes de controladores PID lo toman en el rango de 3 a 33.

Velocidad de Realimentación. Una de las características que origina diferentes representaciones, ecuaciones y nombres para los controladores PID es la forma de uso que se le da al componente derivativo, si éste se aplica a la señal de error o solamente a la señal realimentada, aunque tradicionalmente se entiende que los tres modos de control actúan sobre la señal de error para lograr su eliminación.

Cuando el ruido o un disturbio brusco se presenta en un sistema, una forma ideal de eliminar sus efectos es por medio del uso de un filtro pasa bajo y así evitar o suavizar las perturbaciones generadas por estos fenómenos; sin embargo, en el caso de cambios bruscos y frecuentes del valor de consigna, el modo derivativo introducirá saltos indeseables en la señal de salida del controlador, por lo que es frecuente que éste se aplique solamente a la señal realimentada.

Filtro del valor de consigna. La estrategia usada con la parte derivativa también es aplicable al componente estático del controlador, a la parte proporcional; esto con el fin de reducir la sensibilidad de un cambio en el valor de consigna y evitar sobre impulsos; de esta manera mejorar la respuesta transitoria del controlador. Para tal fin se implementa en el término proporcional un factor de peso *b* para la señal de referencia, este factor puede tomar valores entre 0 y 1.

Finalmente se hace uso del parámetro U_o el cual representa el valor de estado estable o *bias* y que entrega el controlador cuando el error es cero, normalmente, su valor normalizado es de 0.5. Una de las ventajas con su adición es la disminución de oscilaciones de la señal de control y reducción de los tiempos de establecimiento, para mayores detalles ver resultados obtenidos por los autores del presente trabajo consignados en el Anexo A. Los modelos resultantes con las modificaciones mencionadas anteriormente para la estructura PID Industrial paralela se refleja en la ecuación 5 y para la estructura PID industrial serie en la ecuación 6.

$$U(s) = K_c\left(bR(s) - Y(s) + \frac{1}{sT_i}E(s) - \frac{sT_d}{{sT_d}/{N} + 1}Y(s)\right) + Uo \tag{5}$$

$$U(s) = K_c\left(1 + \frac{1}{sT_i}\right)\left(\frac{sT_d + 1}{{sT_d}/{N} + 1}\right) + Uo \tag{6}$$

Donde: *Kc es la* Ganancia proporcional, T_i *es el tiempo* integral, T_d *es el* tiempo derivativo, *b el* factor de pesaje del valor de consigna, N es la constante del filtro de realimentación de velocidad, Uo el bias y finalmente U(s), R(s), Y(s) y E(S) son, respectivamente, las

transformadas de laplace de las señales: esfuerzo de control, consigna, variable controlada y error.

2.4. EL EFECTO WINDUP DEL INTEGRADOR

Si el actuador que ejecuta la acción de control ha saturado los límites, y la saturación no es tenida en cuenta en un diseño de control lineal, el integrador puede sufrir *windup o reajuste excesivo*, este fenómeno causa oscilaciones de baja frecuencia y conlleva a la inestabilidad. El *windup* es debido a que los estados saturados del controlador no son compatibles con la señal de control saturada, y la corrección futura es ignorada hasta que se elimina la saturación del actuador. Una forma racional para manejar el problema de *windup* es tener en cuenta, en la etapa de diseño de control, las limitaciones en la entrada. Sin embargo, este enfoque es complejo y la ley de control resultante es complicada. Las no linealidades del actuador no siempre son conocidas *a priori*. Un enfoque más común en la práctica es agregar una realimentación extra de compensación en la etapa de la aplicación del control, que será incluida en la componente integral del PID. A este tipo de técnicas de compensación se les denomina estrategias *antiwindup* (AW) como se explica en (Bohn and Atherton, 1999). Las estrategias de manejo del *windup* más representativas se discuten en el anexo D.

2.5. EL FENÓMENO *BUMP TRANSFER*

Cuando se cambia el modo de salida del controlador industrial entre manual y automático y viceversa, los transitorios en la señal de salida, debidos a la conmutación entre dos valores diferentes en cada modo, generan un salto en el esfuerzo de control dirigido al actuador, lo que puede perturbar en forma no deseada el sistema bajo control. En el caso de la conmutación entre modos de funcionamiento de un controlador industrial, el método que logre minimizar el salto en la entrada de la planta se denomina *bumpless transfer* (BT) o transferencia sin saltos. Sin embargo, lograr minimizar el salto no siempre es preferible, ya que esto puede provocar un relativo pobre seguimiento del valor de consigna. Es por esto que se debe emplear un método que no sólo reduzca el salto en la entrada de la planta sino que también garantice un buen seguimiento del valor de

consigna. Adicionalmente, debido a que el controlador es un sistema dinámico, es necesario asegurar que el estado del sistema sea el correcto cuando se conmute el controlador entre modos manual y automático. Cuando el sistema está en modo manual, el algoritmo de control produce una señal que puede ser diferente de la señal de control manualmente generada. Entonces, es necesario asegurar que las dos salidas coincidan en el momento de la conmutación, para ello en modo manual el controlador debe realizar un buen seguimiento del valor en manual, si se logra esto se consigue una "transferencia suave" o sin saltos "*bumpless"* al pasar al modo automático (Peng et al, 1996). En el anexo D se estudian algunas de las técnicas AWBT (*Antiwindup- Bumpless transfer*) que la literatura ofrece y se realiza un experimento con la técnica seleccionada.

2.6. SINTONIZACIÓN DE UN CONTROLADOR PID

Entre los principales problemas que se encuentran al momento de la sintonización de un controlador PID, es que con frecuencia se desconoce la forma en que sus modos de control (proporcional, integral y derivativo) han sido implementados y como se relaciona uno con otro dentro del controlador particular a utilizar. Además es importante conocer la ecuación o función de transferencia del controlador al momento de su sintonización, puesto que, para un mismo conjunto de parámetros (ganancia, tiempo integral y tiempo derivativo), el sistema de control se comportará de manera diferente si todos los modos de control procesan la señal de error (en paralelo) o si cada modo procesa la salida del modo anterior (en serie); si todos ellos se aplican directamente al error o no, y otras variantes posibles.

La sintonización de los controladores PID consiste en la determinación del valor óptimo de sus parámetros, que permitan lograr un comportamiento del sistema de control aceptable y robusto de conformidad con algún criterio de desempeño establecido. Para poder realizar la sintonización de los controladores, primero debe identificarse la dinámica del proceso y a partir de ésta determinar los parámetros del controlador utilizando un método de sintonización. Para esta crucial tarea puede procederse de diferentes formas dependiendo fundamentalmente del proceso a controlar y de la información disponible *a priori*.

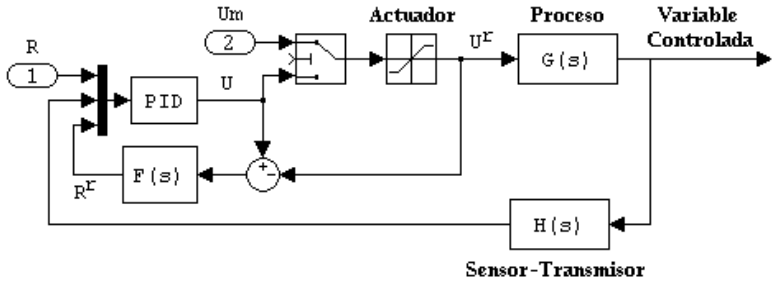
El desarrollo de los métodos de sintonización ha sido extenso desde que *Ziegler and Nichols* propusieron su procedimiento en 1942 (Ziegler and Nichols, 1942). Normalmente, un controlador PID se sintoniza de acuerdo al modo de operación ya sea actuando como regulador, en donde se busca la insensibilidad ante disturbios, o actuando como servomecanismo, ésta condición obliga realizar un buen seguimiento del valor deseado.

En el Anexo B se realiza un estudio de los métodos empíricos más utilizados y sus ecuaciones para sintonizar algoritmos PID, garantizando rechazo a perturbaciones (reguladores) o seguimiento a cambios en el valor de consigna (servomecanismos), además se realiza la comparación de cada método al sintonizar un PID de estructuras paralelo y serie.

2.7. PID INDUSTRIAL AWBT

Con base en la descripción de las variantes realizadas al PID estándar y al PID serie en la sección 2.3 y teniendo en cuenta la elección de la técnica Condicionante (para mayor información remitirse al anexo D), que se presenta en la ilustración 1, como técnica AWBT para eliminar los efectos de *windup* y *bump transfer* descritos en la secciones 2.4 y 2.5, se consignan los diagramas en bloques de un controlador PID Industrial AWBT con capacidad de conmutación Manual/Automático, en sus estructuras serie y paralela diseñados para el presente proyecto, en las ilustraciones 2a y 2b respectivamente.

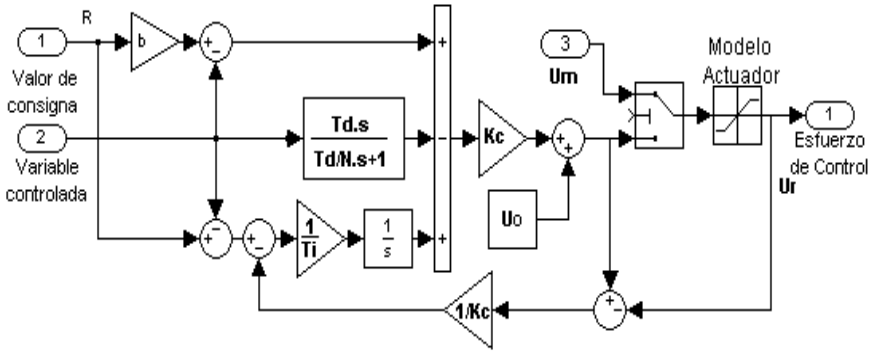
Ilustración 1. Diagrama de bloques Técnica Condicionante AWBT



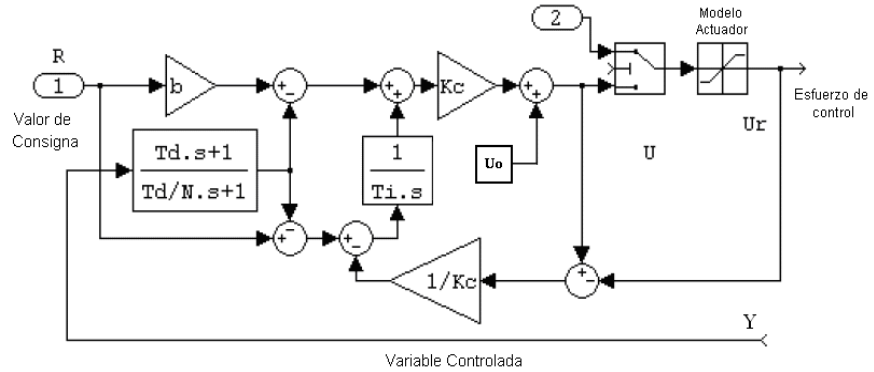
La Técnica Condicionante de la ilustración 1, realimenta una señal llamada referencia realizable R^r, esta es tal que si se aplica al controlador en lugar de la referencia R, la

salida de control U habría sido igual a la entrada real de planta U^r obtenida con la referencia R y bajo esta situación el actuador no está saturado. Esto puede lograrse mediante una compensación adicional que realimenta $U-U^r$ al componente integral del PID mediante un compensador F(s), generando la señal referencia realizable, para una mayor comprensión de la técnica ver anexo D.

Ilustración 2. Diagrama de bloques PID Industrial AWBT



a) Controlador PID Industrial Paralelo AWBT



b) Controlador PID Industrial Serie AWBT

En la ilustración 2 se aprecian las variantes realizadas a la estructura PID explicadas en la sección 2.3: el pesaje en el valor de consigna *b*, el filtro en la variable controlada, la realimentación de la velocidad, el bias Uo y la técnica Condicionante, donde el compensador F(s) de la ilustración 1 es implementado como 1/Kc, siendo Kc la ganancia proporcional. Cuando F(s) se implementa como una constante prescrita el esquema se denomina “algoritmo AW de realimentación lineal”, la técnica Condicionante es un caso particular del “algoritmo AW de realimentación lineal”, cuando esta constante es el inverso de la ganancia de la componente proporcional (Peng et al, 1996).

3. CONTROL EN CASCADA

Uno de los métodos más utilizados para reducir al mínimo los disturbios que entran en un proceso es el control en cascada o de circuitos múltiples. El control en cascada puede acelerar también la respuesta del sistema de control, reduciendo la constante de tiempo de la función de transferencia del proceso que relaciona la variable manipulada con la salida del mismo. El control en cascada se define como la configuración donde la salida de un controlador de realimentación es el punto de ajuste para otro controlador de realimentación, por lo menos. Más exactamente, el control de cascada involucra sistemas de control de realimentación o circuitos que estén ordenados uno dentro del otro.

Algunos de los términos y conceptos utilizados dentro del estudio del control en cascada son: lazo interno o secundario: lazo que controla la variable secundaria o variable manipulada de todo el proceso, lazo externo o primario es el lazo de control de la variable primaria o variable controlada. Tal como se describe en (Saucedo, 2001) el control en cascada se usa comúnmente en los siguientes casos:

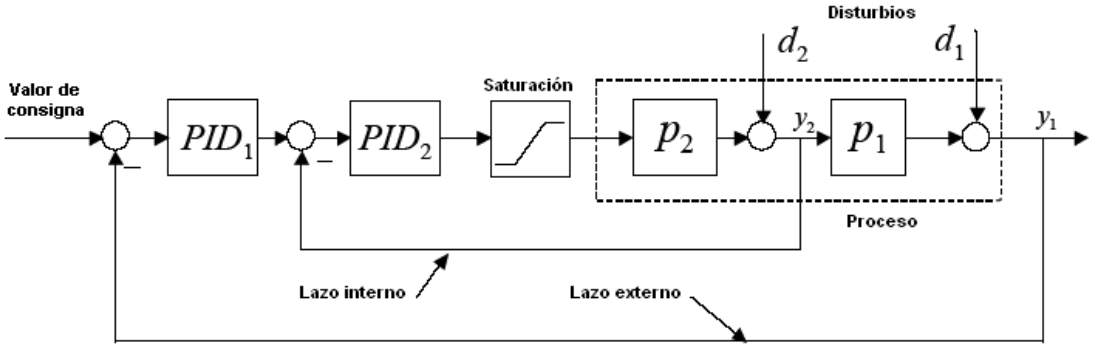
a) Cuando se instala un posicionador para manejar una válvula y el interés primordial es controlar el gasto en una tubería. En este caso la válvula es el proceso secundario y la fricción del vástago es la principal perturbación del lazo interno.
b) Cuando el lazo secundario es el control de flujo, cuyo punto de operación está fijado por el controlador primario. En este caso el problema mayor lo representa la no linealidad de la medición de flujo y la no linealidad de la válvula.
c) Cuando el lazo secundario es el control de temperatura y el punto de operación lo fija un controlador primario (también de temperatura o de composición química), como en el caso de intercambiadores de calor o reactores químicos.
d) Cuando existen varias unidades en paralelo que alimentan a un proceso, y sucede que la carga se reparte más o menos igualmente entre dichas unidades y se desea que el operador pueda variar manualmente alguna de ellas o inclusive sacarla de operación; se desea que el controlador secundario redistribuya la carga entre las demás unidades, de tal manera que la variable de interés se mantenga regulada.

3.1. CARACTERÍSTICAS DE UN SISTEMA DE CONTROL EN CASCADA

Comúnmente, hay tres características principales presentes en el control en cascada para que sea eficaz: la constante de tiempo del lazo cerrado del lazo secundario debe ser menor entre un tercio y un quinto de la constante de tiempo del lazo primario, el lazo secundario debe incluir una fuente de perturbación de proceso importante, y la variable de proceso que se manipula debe ser capaz de desplazar a la variable controlada primaria a su valor deseado. El modo de control para el lazo interno es el más simple compatible con las necesidades del proceso (generalmente proporcional). El lazo interno se ajusta para lograr una respuesta enérgica en la variable manipulada.

En la ilustración 3 se puede ver el diagrama en bloques de un sistema de control en cascada mínimo, es decir con solo dos lazos de control, un lazo interno y un lazo externo. El lazo interno está compuesto por el controlador PID_2, el bloque de saturación que representa el modelo del actuador y P_2 que es la planta a controlar por parte del controlador secundario, la cual se ve sometida al disturbio d_2 y y_2 que representa la variable controlada del lazo interno. El lazo externo está compuesto del controlador PID_1 cuyo esfuerzo de control es el valor de consigna para el controlador PID_2 del lazo interno, P_1 es la planta a controlar por parte del controlador externo, y_1 representa la variable controlada primaria que puede afectarse por el ruido d_1.

Ilustración 3. Diagrama de bloques de un control en cascada



Fuente: http://www.learncontrol.com/pid/cascade2.html

3.2. CONSIDERACIONES EN LA IMPLEMENTACIÓN DEL CONTROL EN CASCADA

Algunas de las consideraciones que se deben tener en cuenta en la implementación de un control en cascada tal como se describen en (Lamanna, 2005), están:

Encontrar la variable secundaria controlada más ventajosa, es decir, determinar cómo el proceso puede ser mejor dividido. Para la selección de la variable controlada secundaria en un sistema de control en cascada es importante partir de algunas reglas como las descritas a continuación, para llegar a una mejor selección de la variable secundaria:

Regla 1. Diseñar el lazo secundario de manera que contenga los disturbios más influyentes. Estos disturbios, las cuales entran en el lazo secundario son las únicas para las cuales el sistema de cascada mostrará mejoría sobre el control de retroalimentación simple.

Regla 2. Seleccionar una variable secundaria cuyos valores estén definidamente y fácilmente relacionados a los valores de la variable primaria. Durante una operación no perturbada la relación entre la variable primaria y la variable secundaria debe estar representada por una sola línea y si esta es una línea recta, la sintonización de los controles es mucho más simple.

Regla 3. Incluir en el lazo secundario tantos disturbios como sea posible, manteniendo al mismo tiempo un control relativamente rápido.

Regla 4. Escoger una variable secundaria de control que permita al controlador secundario operar a la ganancia más alta posible.

El modo integral del controlador primario controlará las compensaciones de errores en la medida de la variable secundaria. También la repetitividad es más importante que la precisión para el sensor del lazo interno.

3.3. SELECCIÓN DE MODOS DEL CONTROL EN CASCADA

A continuación se describen algunas de los modos de configuración de un esquema de control en cascada, tal como aparece en (Chen, 2006).

Controlador Primario. En este caso el controlador primario tiene la misma función como un controlador simple realimentado, y por lo tanto la selección de modos para el controlador primario debería seguir las mismas guías que para un controlador simple.

Modo proporcional en el Controlador Secundario. El controlador secundario debe tener modo proporcional actuando en la señal de error y también debería seguir los cambios en el valor de consigna, tan rápido como sea posible con un pequeño sobre-impulso y disminuir la razón de asentamiento, debe conseguir transmitir los cambios en la salida del controlador primario (valor de consigna del controlador secundario), al elemento final del control al menos tan rápido como si este no estuviera allí.

Modo Integral en el Controlador Secundario. Agregar un bloque integral resulta en una reducción de la ganancia proporcional, además el bloque integral algunas veces no es necesario en el controlador secundario para eliminar el error en estado estable, ya que el bloque integral del controlador primario puede ajustar el valor de consigna del controlador secundario para compensar dicho error.

Si el lazo secundario es rápido y sometido a grandes disturbios entonces:

- El error en estado estable en el controlador secundario debe requerir acciones correctivas por el controlador primario.
- Existe una desviación de la variable controlada primaria del valor de consigna.

A pesar de esto la parte integral se agrega frecuentemente por dos razones.

- La primera, ya que un controlador proporcional produce error en estado estable, el lazo secundario debería tener parte integral si se quiere eliminar por completo el efecto del disturbio, evitando que éste se propague al lazo primario.

- En segundo lugar, generalmente el control en cascada es operado en forma parcial con el control primario fuera de operación, por ejemplo, cuando el sensor primario esta fuera de servicio o está siendo calibrado. En este caso si se introduce el modo integral en el controlador secundario, la respuesta de dicho controlador tenderá a ser más oscilatoria; pero el resultado puede que no sea muy significativo cuando el lazo secundario es mucho más rápido que el lazo primario.

El modo integral no debería ser usado en lazos secundarios en los siguientes casos.

- Si la ganancia es limitada por la estabilidad.
- Si los disturbios dentro del lazo interno, no causan grandes errores en estado estable en la variable controlada de dicho lazo.

Modo Derivativo en el Controlador Secundario. El modo derivativo debe ser usado para compensar retardos en el sensor o retardos de tiempo en el lazo interno. Dicho modo solo debe actuar sobre la variable del proceso y no sobre el error. El uso de la parte derivativa le permite, una alta ganancia del controlador del lazo interno, con menor sobre-impulso y menor razón de asentamiento.

El modo derivativo en el controlador secundario no se sugiere usualmente cuando: el sistema en cascada es PID + PID, lo cual conlleva demasiadas variables a sintonizar y, principalmente, ya que es indeseable poner dos unidades derivativas en serie.

3.4. SINTONIZACIÓN DEL CONTROL EN CASCADA

Para el control en cascada se hace uso de una ley de control PID, lógicamente los modos deben ser seleccionados para cada controlador dependiendo de las necesidades del proceso. La metodología para sintonizar los controladores en cascada teniendo en cuenta que se realiza de manera secuencial es la siguiente:

1. El controlador secundario es ajustado primero porque el lazo interno afecta la dinámica a lazo abierto del lazo externo. Se debe poner el controlador del lazo

interno en manual y sintonizarlo como un lazo PID normal; el controlador del lazo externo no está en operación (debe estar en manual).

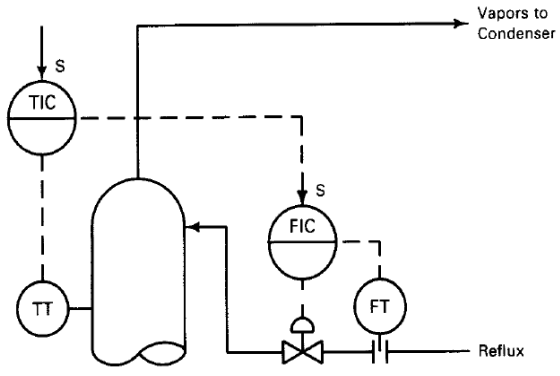
2. El lazo externo se sintoniza (con el lazo interno en automático y el lazo externo en manual) usando cualquiera de los métodos de sintonización disponibles. Se puede sintonizar como un lazo PID normal (Flores, 2000b).

Se debe tener en cuenta que la respuesta en lazo cerrado del lazo externo debe ser 3-5 veces más lenta que el lazo interno, es por ello que después de sintonizar, es conveniente ajustar los valores del PID del lazo externo para alcanzar esto. La variable secundaria debe seguir su valor de consigna tan rápido como sea posible, son permitidos pequeños sobre impulsos (5%) y oscilaciones. La respuesta de decaimiento de un cuarto no es recomendada (sobreimpulso 50%). Se deben usar métodos de sintonización IAE para cambios en el valor de consigna según se recomienda en (Chen, 2006).

3.5. VARIABLES MÁS COMUNES EN LAZOS INTERNOS

Lazo interno con flujo como variable secundaria. El flujo es el lazo mas interno en la mayoría de los sistemas de control en cascada. Un ejemplo de este sistema se puede observar en la ilustración 4.

Ilustración 4. Control en cascada de una Columna de Destilación

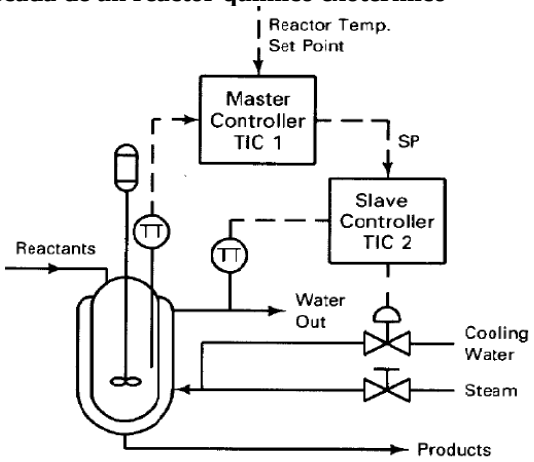


Fuente: (Chen, 2006) p 22

En ilustración 4 se presenta el control en cascada de la temperatura en una columna de destilación, en donde el lazo interno regula el flujo que regresa a la columna por medio de una servo válvula. El reflujo es medido por un transmisor de caudal (FT), este envía su señal a un controlador de flujo (FIC), el cual recibe como valor de consigna la señal proveniente de un controlador de temperatura (TIC), el cual regula la temperatura en la columna recibiendo la medida de esta por medio del transmisor de temperatura (TT), el controlador de temperatura también recibe el valor de consigna (S) de temperatura deseado en la columna de destilación.

Lazo interno con temperatura como variable secundaria. En la ilustración 5 se puede apreciar un sistema de control de temperatura cuando se configura como variable secundaria. Algunas dificultades con el uso de la temperatura como variable secundaria son el retraso del sensor y posibilidad de experimentar *windup*, sin embargo hacer uso del modo derivativo para compensar el retardo en el sensor, siendo el tiempo derivativo igual a la constante de tiempo del sensor, y por otro lado emplear esquemas AW son buena solución.

Ilustración 5. Control en cascada de un reactor químico exotérmico



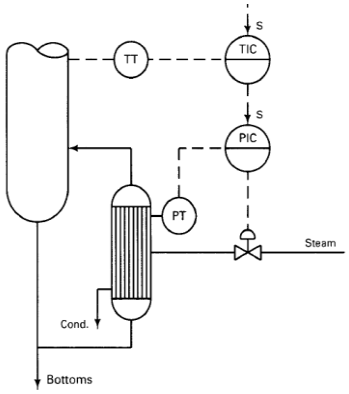
Fuente: (Chen, 2006) pag 24

En la ilustración 5 se presenta el control en cascada de la temperatura en un Reactor Químico Exotérmico, el controlador del lazo externo (TIC1) se encarga de controlar la temperatura de la mezcla que contiene el reactor y enviar el valor de consigna para el controlador del lazo interno (TIC 2), este a su vez con base en la temperatura del agua a

la salida del reactor determina la cantidad de caudal de agua fría que se aplica al vapor entrante a la chaqueta.

Lazo interno con presión como variable controlada. La presión puede ser fácil, rápida y confiablemente medida, es una de las variables más idóneas para ser incorporadas en un lazo interno. En la ilustración 6 es utilizada para controlar la temperatura en una caldera que produce vapor condensado.

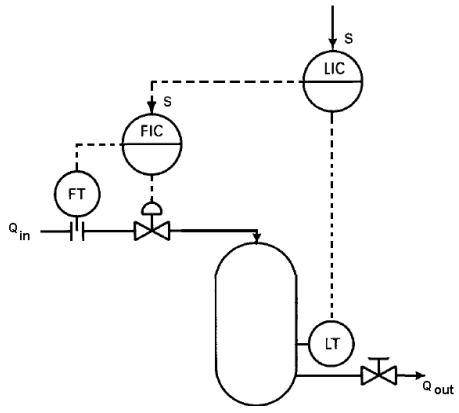
Ilustración 6. Control en cascada de un Intercambiador de calor



Fuente: (Chen, 2006) pag 24

En el caso de estudio del presente trabajo se desea realizar un control en cascada del nivel en un tanque que se ve sometido a disturbios en el caudal de entrada. Un esquema típico del control en cascada para el proceso de nivel descrito es mostrado en la ilustración 7.

Ilustración 7. Control en cascada para un proceso de nivel



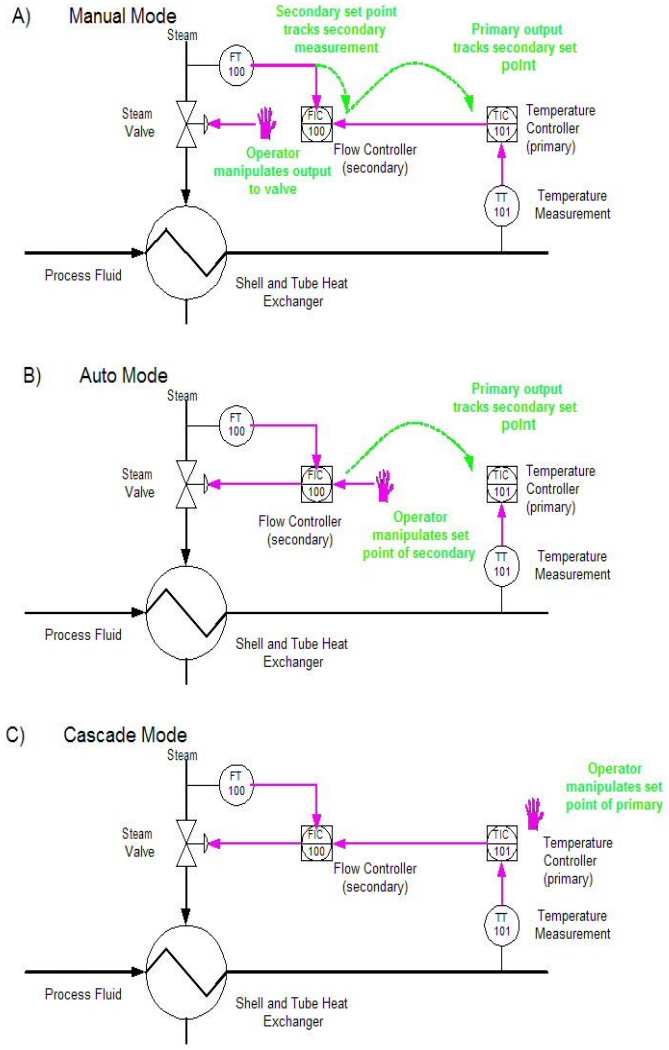
En la ilustración 7 se aprecia que el controlador de nivel del lazo primario (LIC), de acuerdo al nivel real (LT) y al nivel deseado (S), determina la cantidad de caudal que debe permitir ingresar la servo válvula al tanque, este caudal es la señal de consigna (s) para el controlador del lazo interno (FIC) y este, a su vez, de acuerdo a la medición real del caudal de entrada al tanque (FT), aplica los correctivos necesarios en la servo válvula de tal manera que se permita el paso del caudal solicitado por el controlador del lazo primario.

3.6. MODOS DE OPERACIÓN DEL CONTROL EN CASCADA

En la mayoría de aplicaciones, el lazo de control no está funcionando como un lazo en cascada todo el tiempo. El operador tiene la capacidad de cambiar el modo. Los modos manual y automático suelen utilizarse durante el arranque mientras el sistema en cascada es utilizado para la operación normal.

Para ilustrar los modos de operación en cascada se ha tomado como ejemplo un intercambiador de calor con un sistema de control en cascada, donde el flujo de vapor es la variable manipulada y la temperatura de un fluido es la variable controlada. La ilustración 8A muestra el modo manual, el valor de consigna del controlador de flujo sigue la variable de flujo actual, en el modo automático, ilustración 8B, la salida del controlador de temperatura sigue el valor de consigna del controlador de flujo y en el modo Cascada, ilustración 8C, el controlador de temperatura manipula el valor de consigna del controlador de flujo. Por lo tanto, al pasar de un modo a otro no hay cambios bruscos en la salida de la válvula (Shaw, 2006).

Ilustración 8. Modos típicos de un sistema en cascada



Fuente: http://www.learncontrol.com/pid/cascade2.html

3.7. EL CONTROL PID Y EL CONTROL EN CASCADA EN LABORATORIOS DE LA ACADEMIA.

Uno de los principales desafíos en la enseñanza de la Ingeniería consiste en confrontar teoría y práctica. Los laboratorios de Control Automático, de Control Industrial, de control de procesos, entre otros, cumplen un rol específico en la formación de especialistas en Control e Instrumentación, donde los alumnos pueden aplicar conocimientos y metodologías adquiridas en asignaturas teóricas en la resolución de problemas específicos. En este tipo de laboratorios se experimenta con un sistema físico en donde el

estudiante se enfrenta a un proceso real, al trabajar con plantas reales aparecen fenómenos como no linealidades, saturaciones de los actuadores, etc. o situaciones accidentales como irregularidades eléctrico-mecánicas, cables abiertos, etc. estas situaciones son difícilmente experimentables y analizables en una simulación debido a su naturaleza estocástica y limitaciones de las herramientas software; la verdad es que la aparición de estos fenómenos y la interacción con los mismos contribuyen a enriquecer la experiencia práctica adquirida por el estudiante.

En todos los laboratorios académicos se emplean técnicas de control, en especial el control PID, siendo este usado para el control de diversas plantas, típicamente, bajo un esquema realimentado. Esto es lamentable ya que a pesar de ser el esquema de control más común, no es el único empleado en la industria. Por lo que es necesario que los usuarios de estos laboratorios experimenten con esquemas alternativos.

A nivel local el Programa de Ingeniería en Automática Industrial cuenta con el laboratorio de control de procesos, el cual posee plantas en donde se ejecutan prácticas de velocidad y posición, control de temperatura, presión y control de nivel y flujo, controladas todas con PID bajo un esquema realimentado. Estas prácticas tienen como propósito que los estudiantes identifiquen las características dinámicas de cada una de las diferentes plantas, configuren y sintonicen controladores PID y analicen el comportamiento del sistema de control. La presencia de un único esquema control, hace que no se experimente dentro de la formación académica con esquemas alternativos de control como lo es el Control en Cascada, técnica muy aplicada en el control de procesos industriales.

4. SISTEMAS OPERATIVOS DE TIEMPO REAL

En la actualidad es muy común el uso del computador en ambientes como la oficina, el comercio, los hogares y con un buen porcentaje de crecimiento en la industria. Sin embargo existen algunos inconvenientes al incorporar su uso en este último medio, en particular debido a las características que deben cumplir el software y hardware que deben ser usados en este tipo de aplicaciones industriales.

Los sistemas operativos son una parte esencial en un sistema de cómputo, y son estos los encargados de permitirle al usuario interactuar con la máquina de una manera sencilla y eficiente. Los sistemas operativos en el presente son multitarea y/o multiusuario, por lo tanto el sistema operativo es el encargado del correcto funcionamiento del sistema y que los programas de usuario no afecten el funcionamiento de dicho sistema. Sin embargo para el ambiente industrial los sistemas operativos convencionales, no ofrecen algunas características que este ámbito requiere, y es preciso contar con otro tipo de sistemas denominados Sistemas Operativos de Tiempo Real (SOTR), los cuales no difieren mucho de los sistemas operativos normales, pero que ofrecen opciones adicionales a estos tales como determinismo, estabilidad y tolerancia a fallos.

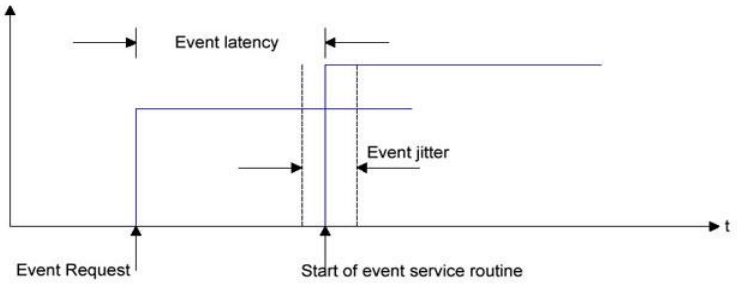
4.1. DEFINICIÓN DE SISTEMA OPERATIVO DE TIEMPO REAL

En un ambiente industrial el sistema operativo tiene que ser fiable, por eso se habla de un sistema operativo que tiene características adicionales a los sistemas operativos convencionales, estos se usan cuando el sistema exige requisitos de tiempo estrictos en sus operaciones de procesado o gestión de datos. Un sistema operativo de tiempo real tiene exigencias temporales bien definidas, el procesamiento debe realizarse dentro de unos intervalos de tiempo definidos, o de lo contrario se producirá una falla en el sistema.

La característica principal de un SOTR es la respuesta determinista limitada en el tiempo ante eventos internos ó externos, tales como interrupciones hardware externas, interrupciones software internas ó interrupciones de reloj internas. Una de las medidas de

rendimiento de un SOTR es la latencia, ó tiempo desde que ocurre el evento y éste es tratado (Proctor, 2002), ver ilustración 9.

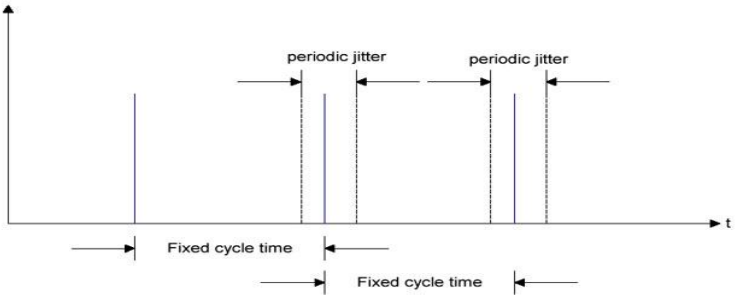
Ilustración 9. Medida de Latencia



Fuente: Intelligent Systems Division National Institute of Standards & Technology

La otra medida es el *jitter*, ó variaciones en el periodo normal de ocurrencia de eventos periódicos, ver ilustración 10.

Ilustración 10. Medida del *jitter* de un sistema



Fuente: Intelligent Systems Division National Institute of Standards & Technology

Todos los sistemas operativos tienden a tener una baja latencia y un bajo *jitter*, pero los sistemas operativos de tiempo real requieren que esos valores estén determinados y que no dependan de la carga del sistema.

En algunas ocasiones se confunde a un sistema que es rápido con sistemas DE tiempo real, Los primeros son denominados sistemas EN tiempo real y su principal característica es la velocidad de respuesta para tener la sensación de realismo, en este tipo de sistemas una demora en la respuesta no provoca fallo, a diferencia de los segundos donde su principal característica es que las tareas se realicen correctamente en un lapso de tiempo

definido. Luego se considera que un sistema de tiempo real está correctamente funcionando únicamente si produce el resultado correcto dentro de los intervalos de tiempo determinados, entonces un sistema de tiempo real se puede definir como “Un sistema en el cual el tiempo en el que se produce la salida es significativo”. Esto generalmente es porque la entrada corresponde a algún movimiento en el mundo físico, y la salida está relacionada con dicho movimiento. El intervalo entre el tiempo de entrada y el de salida debe ser lo suficientemente pequeño para una temporalidad aceptable (Burns y Wellings, 2003).

4.2. REQUISITOS DE UN SISTEMA OPERATIVO DE TIEMPO REAL

Los requisitos de un SOTR no deben confundirse con sus características, los requisitos deben entenderse como las capacidades que debe cumplir el sistema operativo cualquiera que sea, para ser considerado de tiempo real, los requisitos comúnmente exigidos a los SOTR son los siguientes:

4.2.1. Determinismo

Un sistema operativo es determinista, si es predecible, esto quiere decir que realiza las operaciones en lapsos de tiempos fijos y predeterminados independientemente de la carga del sistema. El que un sistema sea determinista depende de la velocidad de respuesta a las interrupciones y de la capacidad del sistema para gestionar todas las peticiones en el tiempo requerido.

4.2.2. Sensibilidad

Se define como el tiempo que consume el sistema operativo de tiempo real en dar el servicio correspondiente a la interrupción después de haberla reconocido. El tamaño de esta ventana de tiempo depende de:

- El tiempo de ejecución necesario para ejecutar la ISR (*Interrupt Service Routine*).

- El efecto de anidamiento de interrupciones, que implica que el sistema debe atender la interrupción más prioritaria, así haya reconocido otra de menor prioridad antes.

4.2.3. Control de usuario

Es una de las principales diferencias en cuanto a un sistema convencional, implica que el usuario puede dar mayor prioridad a los procesos de su interés, modificar y controlar el planificador a sus requerimientos, así como los procesos residentes en memoria.

4.2.4. Fiabilidad y seguridad

Se refiere a la confianza que genera el sistema, inherentemente los sistemas operativos de tiempo real interactúan con el entorno, sirviéndose de sensores y actuadores para tal fin, por lo tanto el sistema tiene que brindar confianza y seguridad de que los datos de entrada, salida y señales de control son los indicados.

4.2.5. Tolerancia a fallos

Cuando un sistema operativo de tiempo real, presenta una falla este debe de estar en la capacidad de poder continuar con las tareas y/o funciones mas criticas conservando el máximo de datos y rendimiento, hasta que la falla sea corregida.

4.3. CARACTERISTICAS DE LOS SOTR

Los sistemas operativos de tiempo real, poseen características que los sistemas operativos convencionales no ofrecen, y que los hacen especialmente aplicables a sistemas en donde se requiera una fuerte interacción con el medio (Zuluaga, 2005), algunas de ellas son:

4.3.1. Mayor interacción con el medio

La interacción con el medio es mayor que en otro tipo de sistemas operativos, debido a que estos sistemas son usados para el control de sistemas, por lo tanto interactúan con sensores y elementos finales de control que le permitan modificar el ambiente en el cual operan, en estos casos se los denomina sistemas reactivos, ya que reaccionan ante estímulos del medio.

4.3.2. Concurrencia

Por concurrencia se entiende la existencia de varias actividades simultáneas o paralelas. Ejemplo de ello lo constituye la concurrencia de varios programas que se conmutan en un procesador. Aunque esta concurrencia no es real en un instante dado (si sólo existe un procesador), sí es real en un intervalo más amplio de tiempo.

4.3.3. Determinismo temporal

El determinismo en un sistema operativo, se define como la capacidad de realizar las operaciones en momentos de tiempos fijos y determinados o en intervalos de tiempo predefinidos. Por lo tanto el tiempo que el sistema toma para realizar los procedimientos requeridos se puede determinar de antemano.

4.4. CLASIFICACIÓN DE LOS SOTR

Los sistemas operativos de tiempo real se pueden clasificar de varias maneras entre ellas se encuentra la clasificación de acuerdo al tipo de procesamiento, de acuerdo a la planificación de tareas, de acuerdo a la integración con el medio físico y también de acuerdo a las escalas de tiempo (Díaz, 2005), sin embargo la clasificación comúnmente más usada es de acuerdo a los límites temporales, según esta clasificación los sistemas operativos de tiempo real se clasifican de la siguiente manera.

4.4.1. Sistemas de tiempo real estricto (*Hard Real Time*)

En este tipo de sistemas los límites temporales son críticos, si no se cumple todos los requisitos de tiempo en las operaciones, se genera una falla en el sistema; esta falla es completa y deja sin posibilidad de continuar con el funcionamiento. En este tipo de sistema se prefiere un trabajo imperfecto dentro de rangos permisibles pero que sea terminado a tiempo, algunas características son: tienen un plazo de respuesta estricto, su comportamiento temporal es determinado por el entorno, el comportamiento en sobrecargas es predecible, tiene requisitos de seguridad críticos, provee tolerancia a fallos (mediante redundancia activa), el volumen de datos es reducido.

4.4.2. Sistema de tiempo real flexible (*Soft Real Time*)

Son aquellos sistemas en los cuales se pueden dejar de cumplir algunos requerimientos específicos pero a diferencia de los anteriores esto no causa una falla total del sistema, este puede continuar su ejecución y el único efecto apreciable es que el valor de las respuestas decrece con el tiempo. Algunas características de este tipo de sistemas son: su plazo de respuesta es flexible, tienen un comportamiento temporal determinado por la computadora, el comportamiento en sobrecargas es degradado, los requisitos de seguridad no son críticos, permite la recuperación en caso de fallos, manejan un gran volumen de datos.

4.5. ARQUITECTURA DE LOS SOTR

Las interrupciones son el mecanismo mediante el cual los dispositivos de E/S notifican al procesador a través de una señal de que poseen información; cuando se genera la interrupción el procesador deja a un lado el código que está ejecutando e inicia el ISR, esta rutina permite obtener la información del dispositivo, el objetivo principal del cambio de un sistema operativo convencional a uno de tiempo real es el manejo de las interrupciones. El procesamiento de interrupciones en el kernel estándar se divide en 2 tareas, la primera se encarga de leer los datos del dispositivo físico y escribirlos en un buffer, esto se conoce como manejador de interrupciones, y la otra se encarga de pasar

los datos del buffer a otro para que sean accesibles por el kernel. Cuando el ISR se está ejecutando todas las interrupciones están inhibidas, con la consecuente producción de un retardo impredecible ante la atención a otras interrupciones que se puedan haber producido y por lo tanto los valores de latencia y *jitter* se incrementan. Para conseguir reducir la latencia y el *jitter* se han desarrollado distintas alternativas que modifican el *kernel*, enfocándose en este aspecto principalmente.

Todas las arquitecturas que a continuación se relacionan son propuestas básicamente para distribuciones Linux (López Zamarrón, 2004), es decir sistemas operativos libres de los cuales es posible obtener el código fuente. En la actualidad hay dos corrientes de diseño para los sistemas operativos de tiempo real.

4.5.1. Atención prioritaria en el kernel estándar (*preemptable kernel*)

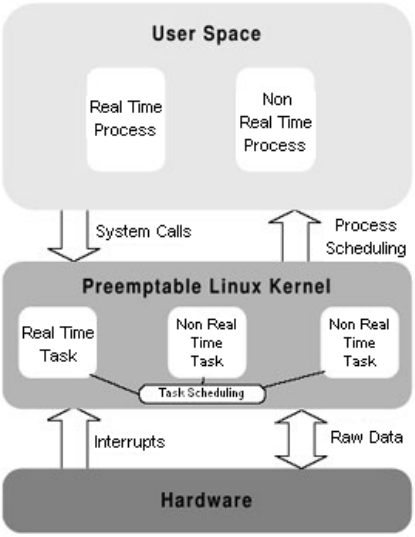
Esta metodología modifica el *kernel* en profundidad de forma que los procesos de kernel ejecuten con máxima prioridad y que puedan interrumpir a procesos de menor prioridad en el acceso a los recursos que necesiten. Este método implica cambios en los manejadores de interrupciones para que las interrupciones de alta prioridad no sean bloqueadas por el manejador de interrupciones mientras está manejando otra de menor prioridad. El resultado de esta metodología es una latencia y un *jitter* del orden de 1 milisegundo en un *Pentium* a 100 Mhz. A partir de la versión 2.5.4 del *kernel* de Linux se incorpora esta metodología.

Como se puede observar en la ilustración 11, la tarea de tiempo real está controlada por el planificador del *kernel* y es una más de las tareas que controla el *kernel.* Esta tarea hace referencia a los procesos de tiempo real en el espacio de usuario. El planificador sabe que las tareas de tiempo real tienen mayor prioridad que las tareas que no son de tiempo real.

Esta metodología es adecuada para aplicaciones de audio y vídeo donde el periodo de interrupciones es del orden de 1 milisegundo, pero inadecuado cuando se habla de

menos de 1 milisegundo. A continuación se relacionan los beneficios y limitaciones de esta estrategia.

Ilustración 11. Arquitectura de Kernel Preemtable



Fuente: http://www.ciclope.info/doc/rtos/rtos.php

Beneficios.

- Desarrollo de aplicaciones de tiempo real más fácil.
- Protección de memoria disponible.
- Los programas de tiempo real no pueden forzar el *kernel*.
- Acceso completo a los servicios del sistema (TCP/IP, I/O).
- Hilos de POSIX[1] disponibles para las funciones de tiempo real.
- Soporte de herramientas para facilitar la depuración.

Limitaciones.

- El rendimiento no es lo suficientemente bueno para los requerimientos de baja latencia en el rango de microsegundos.
- El peor caso de latencia de una interrupción es desconocido ya que no es posible probarlo.

[1] POSIX, es el acrónimo de Portable Operating System Interface, la X viene de Unix como seña de identidad de la API. Estos son una familia de estándares de llamadas al sistema operativo definidos por el IEEE, persiguen generalizar las interfaces de los sistemas operativos para que una misma aplicación pueda ejecutarse en distintas plataformas. Tomado de la Enciclopedia Wikipedia. Octubre de 2008.

- Cambio fuerte en el código fuente del *kernel*.
- Cada vez que sale una nueva versión del *kernel* es necesario probarlo en profundidad.
- Para realizar el análisis de funcionamiento son necesarios todos los controladores de dispositivos y módulos.
- Rendimiento global del sistema reducido.

4.5.2. Modificaciones sobre el *kernel* estándar (Patch)

Existen 4 estrategias de modificación del *kernel* de Linux para proveer capacidades de tiempo real. Tres de ellas implican añadir un segundo *kernel* (*kernel* dual) para manejar las tareas de tiempo real y el cuarto implica modificar directamente el código del *kernel* para añadir características de tiempo real. Estas estrategias que proveen capacidades de tiempo real al sistema operativo tienen ventajas e inconvenientes:

Beneficios.

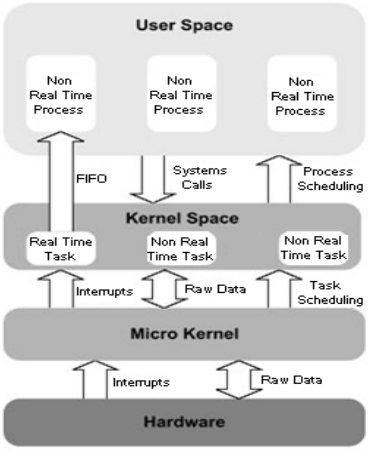
- Cambios de contexto y latencia de interrupciones muy bajo (5 a 10 microsegundos).
- El *kernel* estándar de Linux se ejecuta como una tarea de baja prioridad del *microkernel*.
- Capa de abstracción hardware e interrupciones.
- Garantizada una planificación determinística.

Limitaciones.

- Todas las características de tiempo real deben ser implementadas como módulos.
- Se debe ser conocedor de Linux en profundidad.
- Interacción entre el *kernel* y los drivers de los dispositivos.
- Código incorrecto puede provocar el fallo del *kernel*.
- Más difícil depurar el código.
- Las llamadas al sistema de Linux no pueden tomar el control y el rendimiento no puede ser garantizado.

Micro – kernel. Esta estrategia añade un segundo *kernel* que en realidad es una capa interfaz entre el hardware y el *kernel* estándar, lo que se llama tradicionalmente HAL "*Hardware Abstraction Layer*". Esta capa, *micro-kernel*, controla la ejecución de las tareas de tiempo real y ejecuta el *kernel* estándar como una tarea en *background*, es decir, el *kernel* estándar sólo ejecuta cuando no hay tareas de tiempo real pendientes. En la ilustración 12 se muestra la disposición de esta arquitectura.

Ilustración 12. Arquitectura con micro - kernel



Fuente: http://www.ciclope.info/doc/rtos/rtos.php

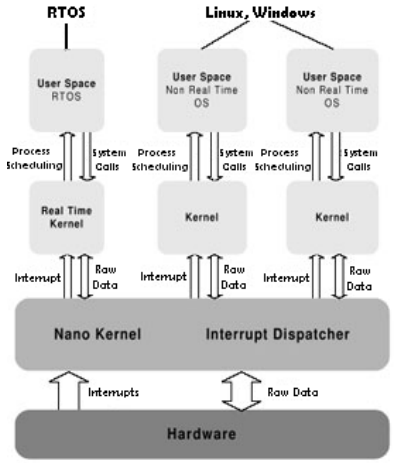
Por lo tanto el *microkernel* intercepta las interrupciones *hardware* y asegura que las tareas de tiempo real se ejecuten con la mayor prioridad posible de forma que la latencia se minimice, para conseguir que se cumplan los tiempos de respuesta requeridos. Un ejemplo de implementación de esta metodología son *RTLinux (Real time Linux)* y RTAI *(Real Time Application Interface).*

Nano - kernel. Esta estrategia es similar a la primera, *microkernel*, pero consigue evitar la patente que tiene Yodaiken[2] sobre ésta, de forma que este *nanokernel* únicamente captura las interrupciones *hardware* y permite la ejecución paralela de varios sistemas operativos por encima de él. En este sentido, esta estrategia no desemboca directamente en un sistema operativo de tiempo real, sino que simplemente es una capa intermedia

[2] Victor Yodaiken creador de RT Linux patenta la arquitectura original en la que se basa RTLinux. A partir del código de Yodaiken, se desarrolla RTAI.

entre el hardware y un sistema operativo, que puede ser de tiempo real ó no. En la ilustración 13 se observa un esquema de esta arquitectura. Una implementación de esta estrategia es ADEOS. Los desarrolladores de este proyecto fueron precisamente los que pusieron el nombre de *nanokernel* para diferenciarlo claramente de la estrategia de *microkernel* patentado por Yodaiken.

Ilustración 13. Arquitectura de nano - kernel



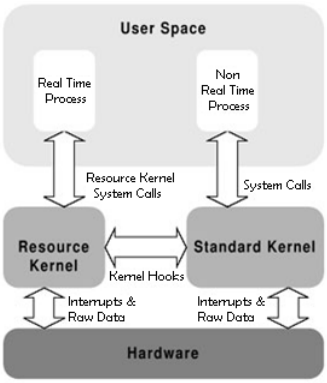
Fuente: http://www.ciclope.info/doc/rtos/rtos.php

Extensión con un nuevo kernel de acceso a los recursos (Recurso-*kernel*). Esta estrategia añade un *kernel* de forma que éste proporciona una puerta de acceso a los recursos, como puede ser el sistema de archivos, el puerto paralelo, etc, tanto para el *kernel* estándar como para los procesos de usuario. El recurso *kernel* no sólo captura las interrupciones sino que proporciona un mecanismo donde los programas de usuario pueden requerir, reservar y garantizarse un porcentaje finito de los recursos como pueden ser de CPU, memoria, etc. En la ilustración 14 se observa el esquema de esta arquitectura.

Extensiones posix de tiempo real añadidas al *kernel*. Esta estrategia consiste en modificar directamente el kernel estándar de Linux para añadir librerías que implementan las extensiones de tiempo real de POSIX. El resultado es un kernel conforme al estándar IEEE 1003.1d. No añade un segundo kernel, las modificaciones realizadas al kernel

consisten en la implementación de relojes, señales, semáforos, memoria compartida, planificador por prioridades, según lo especificado en IEEE 1003.1d.

Ilustración 14. Arquitectura Recurso - Kernel



Fuente: http://www.ciclope.info/doc/rtos/rtos.php

Existen 2 aproximaciones diferentes para esta estrategia. La primera KURT *(The Kansas University Real Time Linux)* que únicamente implementa los relojes conforme al estándar IEEE 1003.1d y la segunda es *TimeSys Linux* que añade al "*preemptable kernel*" un planificador de *kernel*. El problema es que el parche con el planificador no proporciona una alta resolución en los relojes, la cual es necesaria para tareas de tiempo real repetitivas.

4.6. MECANISMOS DE SINCRONIZACIÓN

Es una característica básica que todo kernel debe proveer. La sincronización brinda mecanismos para permitir que los procesos se comuniquen y se sincronicen. La sincronización evita la inconsistencia de datos, el cual es un problema muy común en los sistemas concurrentes. La manera clásica para resolver la sincronización es utilizando mutexes o semáforos (Veiga, 1999).

4.6.1. Mutexes

Son mecanismos que proveen exclusión mutua (los procesos se excluyen unos a otros), para el acceso a una zona critica. Suele sugerirse la idea de imaginar a la sección críticas

como una habitación en la que solo puede entrar un proceso. El *mutex* es la puerta a dicha habitación que puede ser trabada y destrabada. El proceso que traba la puerta (El *mutex*), es el único que puede destrabarla. Este concepto es diferente del semáforo. Los *mutexes* tienen asociada la idea de dueño, mientras que los semáforos no.

4.6.2. Semáforos

El mecanismo semáforo consta básicamente de dos operaciones primitivas, señal (*signal*) y espera (*wait*) que operan sobre un tipo especial de variable semáforo "*S"*. La variable semáforo puede tomar valores enteros y, sólo puede ser accedida y manipulada por medio de las operaciones señal y espera, con la excepción del valor en su inicialización, se definen de la siguiente forma:

Wait(S). Es una operación que decrementa el valor de su argumento semáforo (*S)*.

Signal(S). Es una operación que incrementa el valor de su argumento semáforo (S) siempre y cuando, no haya procesos bloqueados en la cola de semáforos.

4.7. MECANISMOS DE COMUNICACIÓN

El sistema *Unix* proporciona un conjunto de instrucciones para comunicación entre procesos. Este conjunto se encuentra definido dentro de lo que se conoce como IPC (*Inter Process Communication*). Estas instrucciones proporcionan un canal de comunicación entre dos o más procesos. Existen varios tipos de canales IPC, cada uno de ellos proporciona un medio de comunicación entre procesos (Wall, 2000).

4.7.1. Tuberías

Es un mecanismo de transmisión de información unidireccional o *semi-duplex*, permite conectar la salida de un proceso con la entrada de otro, existen dos tipos de tuberías.

Tuberías sin nombre. Carecen del mismo porque nunca necesitan una ruta de acceso y por lo tanto nunca existen en el sistema de archivos. Consisten de dos descriptores de archivo asociados con un *inode* de memoria. Las tuberías sin nombre no permiten el

intercambio de datos entre procesos no relacionados entre sí, es decir para que la comunicación exista los procesos deben provenir de un mismo proceso padre.

Tuberías con nombre. Más conocidas como FIFOS (*First in Input, First in Out),* se denomina con nombre porque equivalen a un archivo, es decir tienen presencia en el sistema de archivos su importancia radica en poder comunicar procesos provenientes de diferente padre.

4.7.2. Memoria compartida

La memoria compartida es un mecanismo muy similar en funcionamiento a los semáforos y a las colas de mensajes. Mediante memoria compartida, como su nombre indica, se pueden crear zonas de memoria que es compartida por varios procesos. De este modo los cambios que un proceso realice a los valores almacenados allí, son visibles para los demás procesos que utilicen esa misma memoria. Por este motivo, en algunos casos será necesario garantizar el acceso en exclusiva para evitar inconsistencias en los datos mediante mecanismos ya vistos como los semáforos. Por ejemplo, si dos procesos acceden a la vez a esta área e intentan modificarla seguramente la modificación no será correcta.

4.7.3. Colas de mensajes

Las colas de mensajes proporcionan un mecanismo para que dos procesos puedan comunicarse entre ellos. Como su nombre lo indica la comunicación se efectúa a través del envío/recepción de mensajes. Los procesos que se comunican a través de este medio no necesitan tener ningún parentesco entre ellos; basta que tengan los medios necesarios para poder acceder a la cola de mensajes para depositar y retirar mensajes de ella. La cola de mensajes reside en el núcleo del sistema operativo, un mensaje es uno o más bloques de información con ciertas estructuras de control "*streams"* asociadas a ellos. Los mensajes pueden ser de diferentes tipos, el cual identifica el contenido del mensaje y el significado del tipo de mensaje es asunto del usuario.

5. DISEÑO E IMPLEMENTACIÓN DEL SISTEMA DE CONTROL

En este capítulo se describe el diseño y la implementación del sistema de control en cascada en tiempo real para la planta de tanques interactuante. Inicialmente se explica el proceso y la planta del caso de estudio, posteriormente la selección del SORT utilizado para garantizar la ejecución de tiempo real en el proceso y se describe el *software* de control, el software de interfaz gráfica y el *hardware* seleccionados, por último se recopila toda esta información y se detalla la implementación completa del sistema en una plataforma de cómputo convencional.

5.1. DESCRIPCIÓN DEL CASO DE ESTUDIO

La planta caso de estudio se presenta en la ilustración 15, en ella se aprecia el sistema de tanques interactuantes con su instrumentación, la planta consta de un tanque plástico grande para almacenamiento y 3 tanques plásticos pequeños, posee un circuito hidráulico y la instrumentación necesaria para efectuar el control de flujo de agua por el circuito o el control de nivel de agua en uno de los tanques o en dos tanques interactuantes.

Cabe resaltar que a la planta de tanques interactuantes se le hicieron importantes mejoras tales como reubicación de la bomba de agua, con lo cual se logra un caudal más uniforme, se reubico la tubería de alivio (bypass) para disminuir el caudal que la bomba impulsa, se adicionó una tubería para generar un disturbio variable, mediante una válvula manual de obturador tipo asiento y una electroválvula, al caudal de entrada y se instaló el sensor de caudal para que cumpliera con los criterios de instalación sugeridos por el fabricante, con lo cual se mejoró el caudal mínimo medido cuando el sensor va de cero a 10 gpm, se instaló y acondicionó todo el cableado para el envío y recepción de señales de los transmisores y por último se instaló un sensor y un transmisor para el caudal de salida del tanque de nivel. La ilustración 15a presenta la planta de nivel antes de los cambios y la ilustración 15b presenta el nuevo aspecto de la planta de nivel.

Ilustración 15. Planta de Tanques Interactuantes

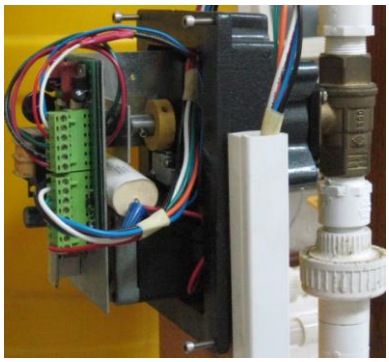
a) Antes de las modificaciones

b) Después de las modificaciones

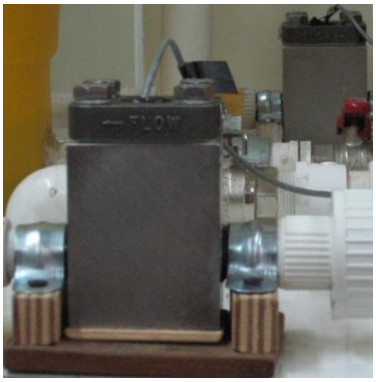
La instrumentación final del sistema se aprecia en la ilustración 16, consiste en una bomba centrífuga QB 128 (ilustración 16a), que entrega un caudal máximo de 38 L/min, una válvula de control de flujo W.E. Anderson ABV111 con un obturador tipo bola manejada por un actuador AMC-100A serie 4078 (ilustración 16b), el cual puede ser usado para un rango de entrada de 1-5 V o 4-20 mA, dos sensores de flujo Metalex 525 +GF+SIGNET (ilustración 16c), utilizados para sensar el caudal de entrada y salida del sistema, 2 transmisores de flujo +GF+SIGNET 8550-1 (ilustración 16d) con salida de 4-20 mA, un transmisor de nivel implementado por medio del transmisor de presión diferencial YOKOGAWA EAJ110 con salida de 4-20 mA (ilustración 16d) en el rango de 0mm a 30mm, una electro-válvula marca Shenling Modelo 2W 260 12 para generar un disturbio en el caudal de entrada al sistema, dos válvulas manuales de asiento que sirven una como bypass y la otra como regulación de la proporción del disturbio (ilustración 16e), y válvulas de bola de accionamiento manual todo unido con tubería PVC de ¾".

Ilustración 16. Instrumentación Planta de tanques interactuantes

a) Motobomba

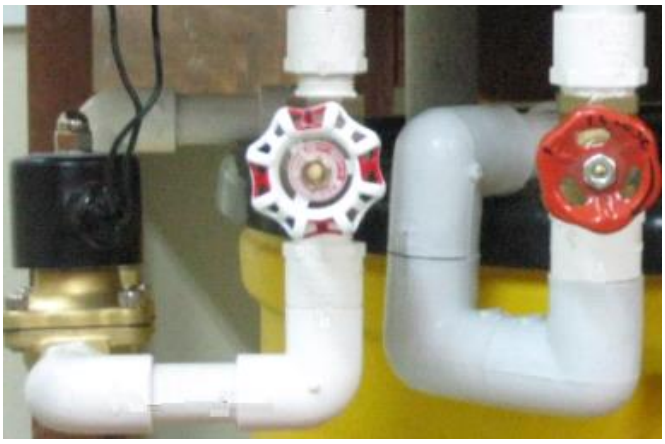


b) Servo-válvula



c) Sensor de caudal

d) Transmisores: nivel y caudal



e) Electroválvula y válvulas manuales de disturbio y bypass

Finalmente al sistema se le implementó un tablero de mando de características industriales que permite controlar de manera eficiente las conexiones y comunicaciones entre la instrumentación, la planta y un PC de características generales en donde corre el sistema de control en tiempo real. El tablero consta de dos pulsadores ON/OFF de

alimentación (ilustración 17a), el primero permite energizar el circuito de control de la instrumentación y de control de la bomba, el segundo posee un enclavamiento con el primer pulsador para la aplicación de potencia a la bomba cuando el contactor se energize, estas conexiones se puede apreciar con más detalle en los diagramas del anexo H. El circuito de alimentación de potencia de la bomba posee un contactor y una protección contra sobre corriente (ilustración 17b), en el panel también se aprecia un soporte con cuatro elementos (ilustración 17c), los cuales son dos pulsadores START y STOP para encender y apagar la bomba en forma manual, y dos indicadores luminosos o pilotos, el verde indica que la bomba está encendida funcionando normalmente y el rojo indica sobrecarga o sobrecorriente de la misma. En el panel también se ubican 3 borneras (ilustración 17d), marcadas en el panel como B1, B2 y B3: en B1 se conectan el cableado de la Potencia hacia la servo-válvula, electro-válvula y bomba, en B2 se conectan las señales de control de potencia de la servo-válvula, electro-válvula y bomba y en B3 se conectan las señales analógicas provenientes de los transmisores con su respectivo acondicionamiento por medio de resistencias de precisión. Finalmente en el panel se encuentra un tablero adaptador de señales de entrada y salida implementado para el proyecto (ilustración 17e), con la tarjeta PCI 6024, ver anexo H para el diagrama eléctrico, en el tablero se aprecia la Placa CB-658 LP que permite realizar las conexiones externas desde y hacia la tarjeta y de esta manera enviar y recibir las señales generadas desde el sistema de control por medio del bus de comunicaciones R6868, además se aprecia el circuito de protección y amplificación de las señales digitales provenientes de la tarjeta implementado con el circuito integrado ULN2803, por medio del cual se logra la corriente suficiente para activar los cuatro relés de estado sólido CX240D5. Dos de estos son usados para encender y apagar la bomba desde la interfaz gráfica, otro para energizar la electro-válvula (Disturbio) y el último para energizar la servo válvula y con ella toda la instrumentación de la planta (Instrumentación).

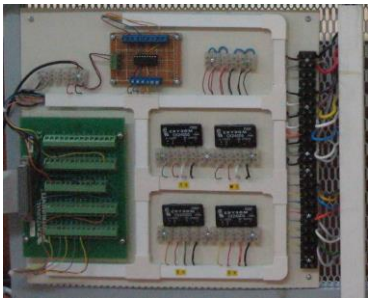
Ilustración 17. Elementos del panel de control

a) Pulsadores ON/OFF

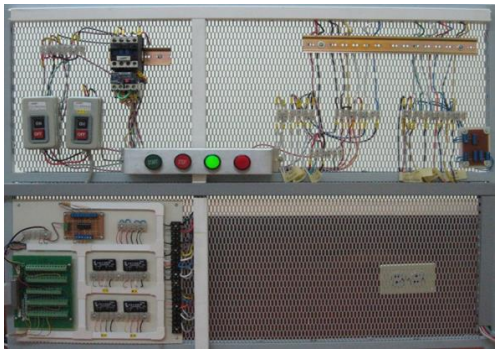
b) Contactor y relé de protección

c) Pulsadores y pilotos

d) Borneras B1, B2 y B3



e) Tablero adaptador de señales de entrada y salida PCI 6024E



f) Tablero de control total

5.2. SELECCIÓN DEL SISTEMA DE TIEMPO REAL

Existen diferentes mecanismos Sw y Hw por medio de los cuales se pueden realizar aplicaciones en tiempo real en el control de procesos, ya sea por medio de lenguajes de programación de tiempo real o sistemas operativos especializados en el caso de Sw o por medio de microcontroladores, procesadores digitales de señales (DSP) o controladores lógicos programables (PLC) en el caso de Hw. En estos últimos se pueden embeber las aplicaciones que son programadas para ejecución en tiempo real, sin embargo, las desventajas más relevantes de estos dispositivos son su alto costo y principalmente su deficiente interfaz Hombre-Máquina. Es por ello que en este proyecto, el sistema de control estará implementado en un computador convencional de propósito general, y para ello se necesita de un sistema operativo que tenga características deterministas o que se le puedan adicionar, es decir un SOTR.

Como se puede apreciar en el Anexo E, existen diferentes sistemas operativos de tiempo real de uso comercial y de distribución libre. La selección está enmarcada en los SOTR de libre distribución ya que no se cuenta con el soporte económico para la inversión en licencias necesarias para la adquisición de un software comercial.

Para la selección del SOTR se tiene en cuenta las siguientes características:

Tabla 1. Características SOTR

Características	Sistemas operativo de tiempo real comerciales				Sistemas de tiempo real de fuente						
	Monta vista Hard hat Linux	Time Sys Linux/RT	Lineo Embedix Real time	REDSonic RED-ICE Linux	RTLinux	RTAI	RTAI+LXRT	RED-Linux	KURT	Linux - SRT	Molnar-Patch
Kernel de tiempo real separado	Con RTlinux	Con RTAI	Con RTAI	Con RTAI	✓	✓	✓				
Programación en espacio de usuario	✓	✓	✓	✓	2	✓	1	✓	✓	✓	✓
Mejoramiento de preemtion al kernel de linux	✓	Dentro de RK		✓				✓			✓
Completo preemtible al kernel de linux	Publico	publico									
Planificador preciso dentro de linux	✓	Con RK dentro de RTAI		✓		✓		✓	✓	✓	
Incluye soluciones para inversión de prioridad dentro de linux		Con RK dentro de RTAI									
Respuesta "rápida" las interrupciones <1ms	Con RTlinux	Con RTAI	Con RTAI	Con RTAI	✓	✓	✓				
Respuesta "moderada" las interrupciones <5ms dentro de linux	✓	Con RK		✓					✓		✓
Mejora la precisión para la medición de tiempo y contadores dentro de linux		Con RK dentro de RTAI		✓				✓	✓		
Nota: (1) = Limitado (2) = Muy limitado											

Fuente: Tomado y modificado de http://www.linuxdevices.com

En la tabla 1 se confrontan las características más relevantes de un SOTR con los sistemas operativos de este tipo tanto comerciales y los de distribución libre, la tabla muestra como RTAI, cumple con algunas de estas características, pero también indica como SOTR comerciales se apoyan en RTAI para desempeñar también algunas de ellas.

A nivel de experimentación, también se han realizado varias comparaciones entre los SOTR, lo que ha permitido a los investigadores realizar una diferenciación en el uso de dichos sistemas en sus aplicaciones y realizar una buena elección, tal como se muestra

en (Barbalace et al, 2008), (Ripoll, 2002) y (Mattei and Ludiciani, s.f). Con base en los resultados obtenidos en dichos estudios se concluye que RTAI es un SOTR libre con una comparable predictibilidad, presentando las menores latencias y siendo este el más eficiente y apto para ser implementado dentro de una plataforma de cómputo convencional. Adicionalmente entre las ventajas, está que en sus nuevas versiones incluyen a RTAI-Lab que es una aplicación para monitorear en espacio de usuario las tareas de tiempo real. También con RTAI y RTAI-Lab se han implementado una amplia variedad de proyectos de control y monitoreo, éstos se han realizado principalmente en Europa. Entre los proyectos más representativos se pueden mencionar: el control de un péndulo invertido generando código en tiempo real con RTAI-Lab como se muestra en (Bucher et al, 2004), también el sistema de estabilización de una esfera, proyecto del Instituto Nacional de Investigaciones en Informática y Automática INRIA que además usó RTAI-XML (Mannori et al, 2006), y otro proyecto interesante es el control en tiempo real de un manipulador industrial con RTAI-Linux el cual ha sido desarrollado en el Laboratorio de Mecatrónica de la Universidad de Modena y Reggio Emilia en el año 2003 (Secchi et al, 2003).

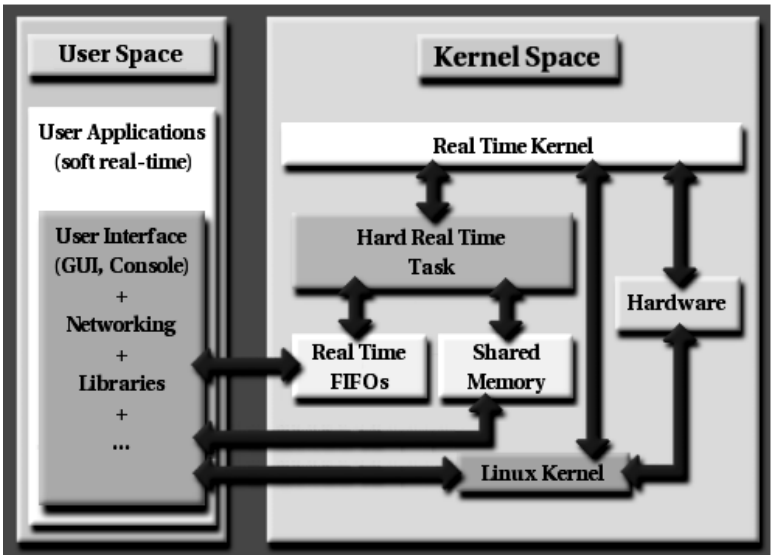
Todo lo anterior indica que RTAI es la arquitectura de tiempo real más idónea que permitirá darle al presente proyecto, no solo, las características de tiempo real necesarias, sino las herramientas de soporte a emplear en su implementación.

5.2.1 RTAI

RTAI posee una arquitectura *micro-kernel,* la cual se caracteriza por un pequeño *kernel* que se sitúa entre las capas de Linux y el hardware del computador. El *micro-kernel* se encarga de tomar todas las interrupciones y de la comunicación hardware-software para que reaccione a las restricciones de tiempo real duro. Este también re-implementa la función del planificador regular del *kernel* para agregar tareas de tiempo real y servicios de procesamiento de tiempo real, es importante advertir que RTAI implementa ambos, las tareas de tiempo real en el espacio del *kernel* y el procesamiento de tiempo real en el espacio de usuario. La comunicación entre las tareas de tiempo real y el espacio de usuario es posible a través de Mensajes, *Mailboxes* y Fifos. La tarea de tiempo real puede

ser ejecutada como un módulo del *kerne*l que posee mayor prioridad dentro del proceso. La arquitectura del sistema RTAI se muestra en la ilustración 18.

Ilustración 18. Arquitectura de RTAI



Fuente: Tomado y modificado de http://chrtai.sourceforge.net/

RTAI posee una aplicación en espacio de usuario llamada RTAI-Lab, que se embebe dentro de la aplicación *Scilab/Scicos* y permite desarrollar tareas de control de tiempo real y de monitoreo en forma de diagramas en bloques, los cuales pueden ser compilados y ejecutados en el sistema operativo de tiempo real Linux/RTAI. RTAI-Lab está incluido en la distribución de RTAI. RTAI fue desarrollado en el Departamento de Ingeniería Aeroespacial del Politécnico de Milano (DIAPM) y el principal desarrollador del proyecto RTAI-Lab es Roberto Bucher de la *Scuola Universitaria Professionale della Svizzera Italiana* (SUPSI) (Bucher, Mannori and Netter, 2008). Hasta el momento de la edición de este documento la última versión de RTAI es la 3.6.1. La versión utilizada para el desarrollo y la implementación del sistema es la versión 3.6.

Adicionalmente RTAI-Lab permite.

- Desarrollar y ejecutar software de tiempo real en forma local, remota o distribuida.
- Monitorear la ejecución de un controlador de forma local, remota o distribuida.
- Cambio de los parámetros del sistema en línea.

- Agrega una paleta RTAI-Lib de bloques a Scilab/Scicos para desarrollar diagramas de bloques de tiempo real.
- Permite "*host and target*" para comunicación vía netrpc.
- Osciloscopio Virtual *xrtailab* y monitoreo de aplicaciones permitiendo interactuar con el ejecutable de tiempo real.
- Generación automática de código de tiempo real desde *Scilab/scicos.*
- Posibilidad para migrar de RTAI a *Matlab/Simulink/RTW*
- Interfaz para el hardware de adquisición de señales y otros dispositivos soportados por Comedi.

RTAI-Lab está basado en las siguientes herramientas.

Scilab/Scicos. *Scilab* es un software CACSD **(***Computer Aided Control System Design*) de fuente abierta para computación numérica. *Scilab* incluye *Scicos* un editor de diagrama de bloques que puede ser usado para crear simulaciones y generar automáticamente código compilado implementando tareas en tiempo real (Bunks et al, 1998). La última versión usada de *Scilab/Scicos* es la 4.1.2.

RTAI-Lib. Es una paleta de bloques de *Scicos* que permite diseñar diagramas de bloques con sensores y actuadores. Esta provee una interfaz entre RTAI y el hardware de adquisición de señales. Los diagramas de bloques que usa RTAI-Lab pueden ser compilados dentro del software *Scilab/Scicos*.

Xrtailab. Es un software que emula un osciloscopio que se puede conectar a procesos que se ejecutan en tiempo real. Esto permite visualizar y monitorear señales y eventos de tiempo real usando indicadores, *scopes* y *leds* simulados. *Xrtailab* también permite ajustar parámetros de las aplicaciones de tiempo real mientras se está ejecutando el código de tiempo real. Esta aplicación hace parte de RTAI.

Comedi. Proporciona los *drivers*, librerías de funciones y una API *(Application Program Interface)* para interactuar con cientos de dispositivos hardware de adquisición de señales

(Schleef, Hess and Bruyninckx, 2007). La librería de comedi viene distribuida con los siguientes paquetes.

Comedi es una colección de controladores o *drivers* para una variedad común de tarjetas de adquisición de datos que se conectan a la *main-board* del computador. Los controladores son implementados como un módulo del kernel de Linux proporcionando funcionalidad y comunicación con los diferentes dispositivos.

Comedilib es una librería en espacio de usuario que proporciona una interfaz de desarrollo amigable para dispositivos que se pueden comunicar por medio de Comedi. Esta librería aporta la documentación, utilidades de configuración y calibración, y programas de demostración.

Kcomedilib es un módulo del *kernel* de Linux que proporciona la misma interfaz como Comedilib en el espacio kernel, adecuado para tareas de tiempo real. Es efectivamente una "biblioteca del Kernel" para usar Comedi desde tareas de tiempo real. Las versiones usadas de Comedi es la 7.76 y comedilib en su versión 7.22.

RTAI adicionalmente se presenta como una alternativa comparada con sistemas de tiempo real comerciales típicos, en la tabla 2 se muestran las funcionalidades que ofrecen paquetes comerciales, solo que en un solo paquete de distribución libre, RTAI las recoge a todas.

Tabla 2. Comparativa de RTAI-LAB con sistemas comerciales

APLICACIONES	SISTEMAS COMERCIALES	RTAI
Drivers de Comunicación	Proveedores comerciales	*COMEDI*
Procesamiento numérico.	*Matlab/Simulink*	*Scilab/Scicos*
SOTR	*QNX, VxWorks, Windows NT*	*RTAI*
Generación de Código de Tiempo Real	*Mathworks-Real Time Workshop*	*RTAI-Lib*
Interfaz grafica	*LabView*	*Xrtailab*

Fuente: www.rtai.org/RTAI-Lab-tutorial.pdf

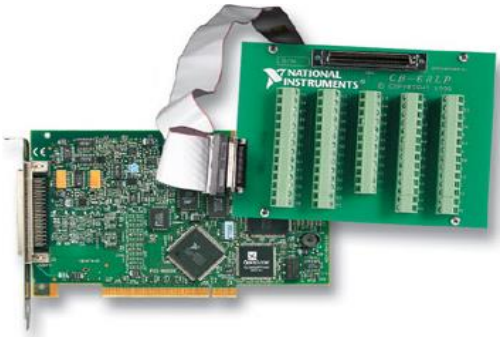
Todo el sistema debe estar soportado por un sistema operativo convencional Linux, para este caso se selecciona *Fedora Core 3*, debido a que posee como compilador gcc en su versión 3.4.2, versión que es estable y apta para la compilación e instalación manual de

paquetes del sistema operativo de tiempo real seleccionado, según las recomendaciones de los autores (Bucher, Mannori and Netter, 2008). Para realizar una correcta instalación del sistema de tiempo real completo se ha generado una guía de instalación paso a paso descrita en el Anexo F, junto con la descripción de las herramientas que incluye RTAI-Lab, con las versiones del software utilizado y sitios de descarga.

5.3. COMUNICACIÓN Y ADQUISICIÓN DE DATOS

Para la comunicación de las señales generadas desde la instrumentación de la planta y desde la estrategia de control (PC-SOTR), se utiliza la tarjeta PCI 6024E y la placa CB-68LP de la *National Instruments,* esta última permite realizar conexiones externas desde y hacia la tarjeta y de esta manera enviar y recibir las señales generadas desde el sistema de control por medio del bus de comunicaciones R6868, ver ilustración 19. Se ha seleccionado esta tarjeta porque los *drivers* de comunicación están soportados dentro de la librería Comedi, que es compatible con RTAI. El driver provisto por comedi se denomina *ni_pcimio*, el cual se asocia a la tarjeta mediante comandos desde un terminal. Los pasos de configuración de la tarjeta y el módulo asociado se explican en el Anexo G. Para mayor información de la tarjeta ver (National Instruments, 1998).

Ilustración 19. Hardware de Comunicaciones



Fuente: National Instruments.

5.4. DEFINICIÓN DE LA ESTRUCTURA DEL SISTEMA DE CONTROL EN CASCADA

Una vez instalado el sistema de tiempo real con base en el anexo F, se procede a realizar el diseño del esquema de control en cascada. El diseño de un sistema de control en cascada se realiza de acuerdo al proceso a controlar, para el caso de estudio, y con base en la teoría analizada se ha elegido una estrategia PI como lazo interno y una estrategia PID como lazo externo. Además como requerimiento del presente trabajo se ha determinado emplear la ley de un controlador industrial PID AWBT en sus estructuras serie y paralela; de ahora en adelante cuando se haga referencia a un controlador PI o PID, se debe entender que se hace referencia a un controlador industrial PID con AWBT.

Para la implementación se sigue un procedimiento de tres pasos, donde cada esquema de control se asocia a una tarea de tiempo real y cada tarea de tiempo real se asocia de igual manera a un diagrama en bloques, es decir cada una de ellas se ejecuta en el sistema como una tarea de tiempo real independiente y se ejecutan secuencialmente a medida que el usuario avance en la práctica de laboratorio. Los esquemas de control a implementar son:

- Control realimentado con un controlador PID paralelo.
- Control realimentado con un controlador PID serie.
- Control en cascada (Con un controlador PI + Controlador PID).

Adicionalmente se realizan los diagramas en bloques con el fin de generar las tareas de tiempo real para las prácticas de Histéresis de la Válvula, Identificación de la planta, Control PID paralelo y serie, Identificación de los lazos primario y secundario para sintonización del control en cascada y por último la práctica del control en Cascada.

Para realizar un eficaz y adecuado uso de las tareas de tiempo real generadas a partir de los diagramas de bloques mencionados, el usuario del sistema contará con la opción de aplicarlas de manera secuencial y de seleccionar el esquema de control de acuerdo al acondicionamiento de la guía para la ejecución completa de la práctica, ver Anexo I.

5.4.1. Diseño del sistema software para el control en cascada

La herramienta software actual para el desarrollo de la práctica en la planta de tanques interactuantes se ejecuta en una plataforma de computo convencional que tiene como sistema operativo *Windows XP*, y como programa de soporte para las interfaces hombre-máquina (H-M) y el algoritmo del controlador virtual el software *Labview* de *National Instruments* (versión 7.0). Las prácticas que se realizan por medio de esta aplicación son:

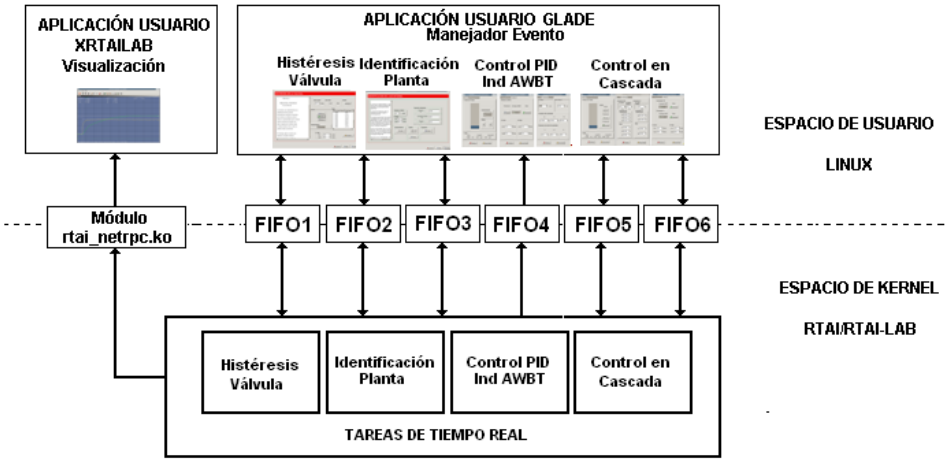
- Histéresis de la Válvula
- Control On-Off
- Control PID

Para el nuevo sistema de control se ha conservado dos de las practicas anteriores: histéresis de la válvula y el control PID, ésta última con la diferencia que se implementa un PID Industrial AWBT. Las prácticas implementadas en el nuevo sistema de control son:

- Histéresis de la Válvula.
- Control PID Industrial AWBT
- Control en Cascada

En la ilustración 20 se aprecia la arquitectura del sistema software para el nuevo sistema de Control en cascada.

Ilustración 20. Arquitectura del sistema Sofware del Control en Cascada



En dicha ilustración se muestra que a cada práctica se le asigna una tarea de tiempo real que corre en el espacio de *kernel*, generada previamente con la herramienta Scilab/Scicos, dichas tareas se comunican por medio de FIFOS con la interfaz de usuario, y por medio de ellas se realiza el envío de comandos y parámetros hacia las tareas de tiempo real. Adicionalmente se puede ver en el espacio de usuario que se utilizarán dos interfaces graficas la primera basada en el entorno de desarrollo para aplicaciones graficas Glade, por medio del ella el usuario selecciona la práctica a realizar al igual que los métodos de sintonización, se envía comandos al sistema, cambio de parámetros, aprecia el estado del nivel en el tanque, entre otras. La segunda interfaz de usuario es el osciloscopio virtual Xrtailab que provee RTAI-Lab, en el se presentan las tendencias de las variables como son nivel, flujo, los esfuerzos de control y se pueden realizar de forma online la medición de las mismas. Esta interfaz de usuario se comunica con las tareas de tiempo real por medio del modulo *rtai_netrpc.ko* que se encuentra en espacio de *kernel*.

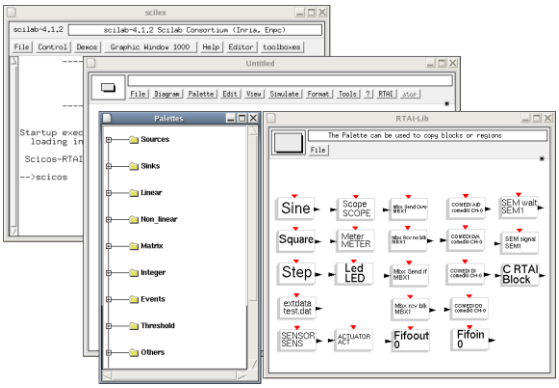
5.5. IMPLEMENTACIÓN DE LAS TAREAS EN TIEMPO REAL DEL SISTEMA DE CONTROL

5.5.1 Creación de un diagrama de bloques en Scilab/Scicos

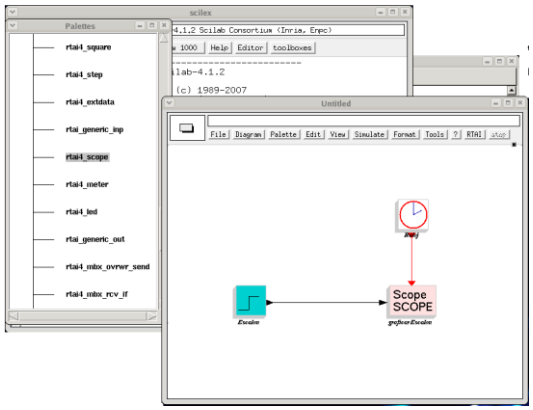
Para el diseño de los diagramas de bloques se utiliza Scilab/Scicos en su versión 4.1.2, soportada por RTAI. Inicialmente se debe arrancar el software de diseño, en una consola de comandos o terminal y como usuario del sistema se escribe *scilab* con lo cual aparece la ventana de la aplicación, luego en el *prompt* de *scilab* se escribe *scicos,* aparece otra ventana en donde se realizará el diseño. En el menú *Palettes* con *click* sostenido se selecciona *Palette* o *Pal Tree* y aparecen el conjunto de librerías para construir diagramas de bloques, entre las que se destacan: *Sources, Sinks, Linear, Non Linear*, *Events* y la paleta que adiciona RTAI-Lab llamada RTAI, tal como se aprecia en la ilustración 20.

De acuerdo al tipo de señal a generar o de la arquitectura del diagrama de bloques se seleccionan los bloques contenidos en las librerías. Con el software Scilab/Scicos se deja a iniciativa del programador el diseño del diagrama de bloques, ver un ejemplo en la ilustración 21.

Ilustración 21. Software de diseño Scilab



a) Opciones de Scilab



b) Diagrama de bloques básico realizado en Scilab/Scicos

Se debe tener en cuenta que dentro de cada diagrama de bloques el uso de un evento de reloj (bloque rojo) se utiliza para activar los demás bloques periódicamente con la frecuencia deseada por el usuario y de acuerdo a las condiciones del proceso a controlar.

Los diagramas de bloques diseñados con *Scicos* contienen dos tipos de vínculos: regulares y de activación. Los vínculos regulares transmiten señales y los vínculos de activación transmiten información de activación y sincronización. Por defecto, los vínculos de activación se señalan en rojo y los puertos regulares en negro, también los puertos que reciben y envían señales son colocados en ambos lados de los bloques mientras que las entradas de activación y salidas son respectivamente en la parte superior y en la parte inferior de los bloques, la mayoría de bloques siguen esta convención, pero pueden ser definidos por el usuario.

Teniendo los componentes *software* a punto, se procede a realizar la implementación de las tareas de tiempo real del sistema de control. Los diseños realizados de las estrategias de control se presentan en las ilustraciones 22 a la 31. En los diagramas de bloques la adquisición de los datos se realiza por medio de bloques Comedi asociados a las

variables del proceso y señales de activación de los equipos tal como se muestra en la Tabla 3.

Tabla 3. Bloques Comedi asociados a variables y parámetros del sistema.

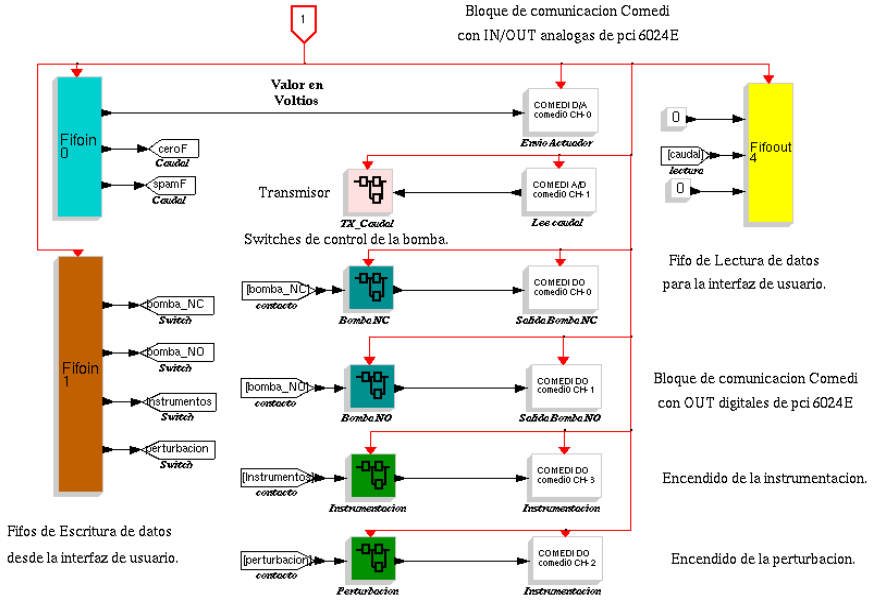
Variables y Parámetros	Bloque Comedi Asociado
Nivel	
Caudal de entrada	
Caudal de Salida	
Esfuerzo de Control Manual o Automático	
Señal Normalmente Cerrado (NC) Bomba	
Señal Normalmente Abierto (NO) Bomba	
Señal Activación Perturbación	
Señal de Activación Instrumentos	

5.5.2 Diagrama de bloques de la tarea de tiempo real de la histéresis de la válvula

A partir de este diagrama, ver ilustración 22, se genera la tarea de tiempo real, que permite realizar la práctica de la curva de histéresis para el actuador, en ella se aprecian las fifos de comunicación. En este caso la fifo 0 sirve para comunicar el valor del voltaje 1-5 V hacia el actuador y el valor del cero y el *span* del transmisor de flujo. El bloque denominado TX_caudal (Bloque rosado) hace una conversión del valor leído desde la tarjeta normalmente de 1 a 5 voltios y lo convierte a unidades de ingeniería de 0 a 10 gpm. La fifo 4 envía el valor del caudal hacia la interfaz de usuario, esta fifo también es capaz de enviar el valor de nivel (primera entrada) y el valor del caudal de salida (tercera entrada). La fifo 1 comunica las señales discretas de disparo para la bomba, electroválvula (disturbio) y servo válvula (instrumentación), a través de los bloques comedi definidos en la tabla 3, estos bloques están comunicados con las salidas discretas de la tarjeta PCI 6024E. Los bloques Bomba_NC, Bomba_NO, Instrumentación y Perturbación,

son bloques que permiten conmutar entre un 0 y un 1, dependiendo del valor que le llegue de la interfaz de usuario. El bloque transmisor de caudal, las salidas de la fifo1: los bloques Bomba_NC, Bomba_NO, Instrumentación y Perturbación y la fifo 4 con sus respectivas entradas, se usan en todos los diagramas de bloques de las tareas por lo que no se volverá a mencionar su función dentro de los diagramas.

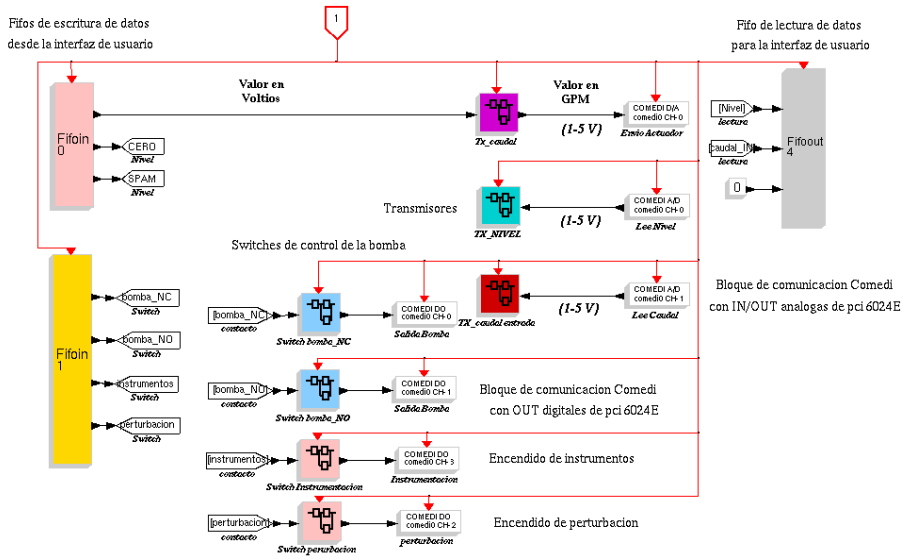
Ilustración 22. Diagrama de bloques Histéresis de la Válvula



5.5.3 Diagrama de bloques de la tarea de tiempo real para la identificación de la planta de nivel

En el diagrama de bloques de la ilustración 23 muestra la distribución de los bloques para la identificación de la planta de nivel (curva de reacción), en este diagrama el transmisor de caudal cumple una función igual que en el anterior diagrama solo que ahora el valor viene desde la interfaz de usuario. También están presentes en el diagrama un bloque TX_nivel (Azul), el cual permite convertir el voltaje de 1 a 5 voltios del transmisor de nivel a unidades de ingeniería de 0 a 30 centímetros. El bloque TX_caudal_salida, convierte la señal de 1 a 5 voltios del transmisor de caudal de salida a unidades de ingeniería de 0 a 10 gpm, estos bloques transmisores cumplen la misma función en todos los diagramas de bloques por lo tanto no se volverá a hacer referencia a ellos a menos que exista una nueva función que estos desarrollen.

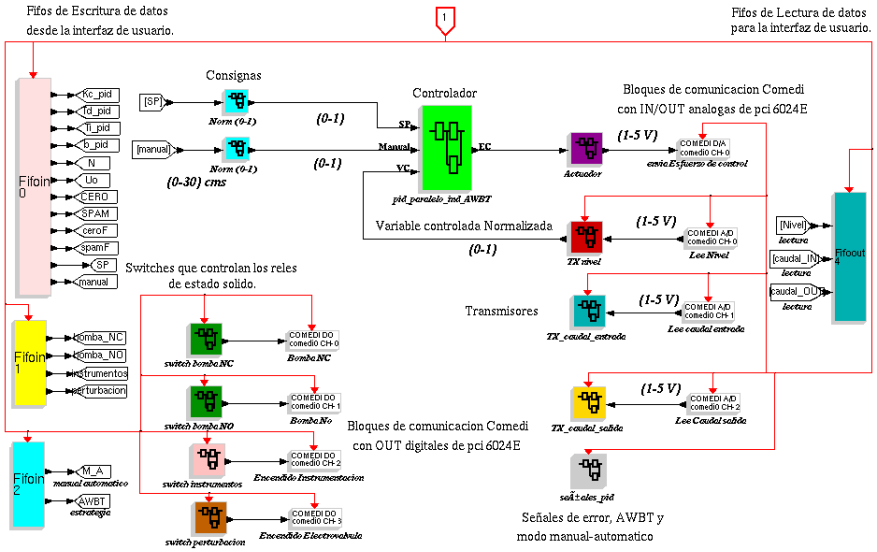
Ilustración 23. Diagrama de bloques para la Identificación de la Planta de nivel.



5.5.4 Diagrama de bloques de la tarea de tiempo real del esquema realimentado con PID industrial paralelo AWBT

La ilustración 24 muestra el diagrama de bloques del sistema de control con un PID paralelo como ley de control, en este diagrama la fifo 1 comunica los valores de las constantes de sintonización, parámetros del controlador como Uo, pesaje del valor de consigna (b), valor de la constante del filtro derivativo (N), el valor de las consignas manual y automática, y los valores de cero y *span* de los transmisores de caudal y flujo, todos estos valores vienen desde la interfaz de usuario. En el diagrama se aprecian dos nuevos bloques denominados Norm (0-1), estos bloques convierten los valores en unidades de ingeniería de 0 a 30 cms del valor del nivel a valores de 0 a 1 para las operaciones normalizadas en el controlador, así mismo el bloque TX_nivel cumple una nueva función en el diagrama y es convertir la señal de 1 a 5 voltios del transmisor de nivel a valores de 0 a 1 para el controlador, el bloque denominado actuador (morado), tiene la función de convertir el valor entre 0 y 1 del esfuerzo de control del controlador a valores de 1 a 5 voltios para aplicárselos a la servo válvula. Para la creación de la tarea de tiempo real del control con PID Industrial serie AWBT, basta con cambiar el bloque del controlador paralelo por el bloque del controlador serie y el diagrama queda listo para ser generado.

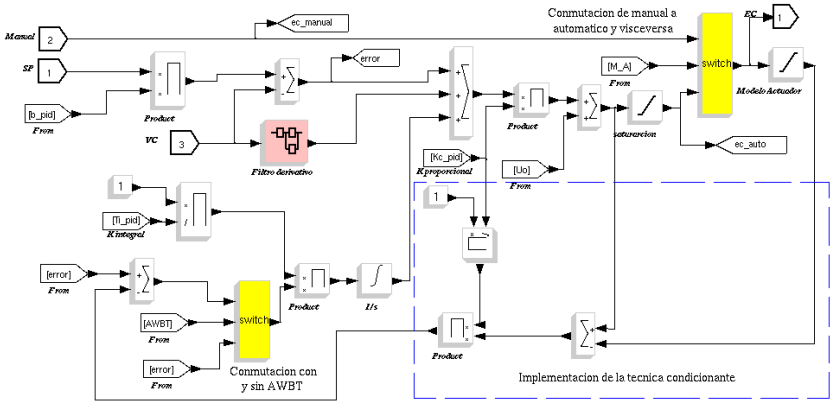
Ilustración 24. Diagrama de bloques del esquema realimentado con PID Paralelo AWBT



5.5.5 Diagrama de bloques del controlador PID industrial paralelo AWBT

En la ilustración 25 se presenta el diagrama de bloques del PID Paralelo AWBT, en el se aprecian los mandos manual y automático (M_A), el pesaje en el valor de consigna *b* (b_pid), el filtro de primer orden en la componente derivativa bloque rosado (ecuación 3a capitulo 2). Además se muestran dos conmutadores (de color amarillo), el conmutador superior permite la conmutación de mando manual al automático y viceversa y el conmutador inferior permite la activación de la estrategia AWBT, en el diagrama también se muestra la implementación de la técnica Condicionante como estrategia AWBT.

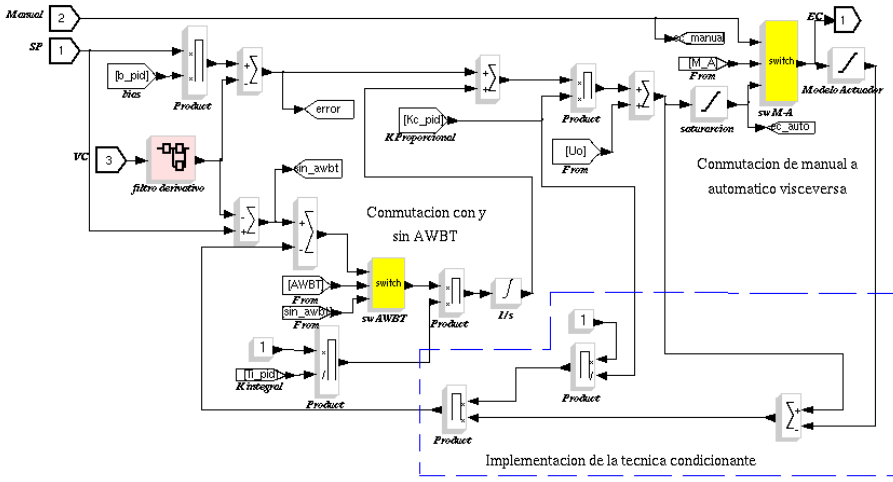
Ilustración 25. Estructura Interna PID paralelo industrial AWBT



5.5.6 Diagrama de bloques de un controlador PID industrial serie AWBT

En la ilustración 26 se indica el diagrama de bloques de la estructura Interna PID serie industrial AWBT, se puede apreciar los mismos conmutadores presentes en el PID paralelo: el conmutador de manual a automático y el conmutador para activar o desactivar la estrategia AWBT. En este controlador también se implementa la técnica Condicionante como estrategia AWBT, así como el filtro derivativo (Bloque rosado), tal como se indica en la ecuación 3b del capítulo 2.

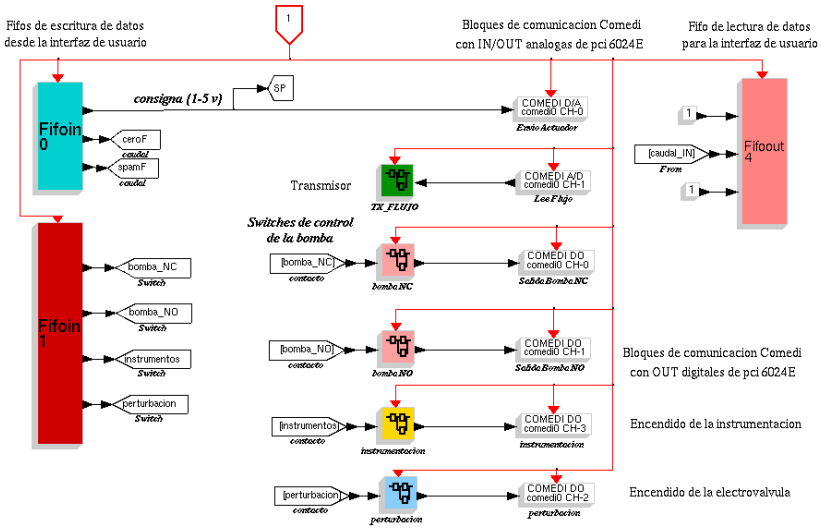
Ilustración 26. Estructura Interna PID serie industrial AWBT



5.5.7 Diagrama de bloques de identificación del lazo interno para el control en cascada

En la ilustración 27 se muestra el diagrama de bloques empleado en la identificación del lazo interno, este diagrama es utilizado para realizar la identificación del sistema a manipular, en este caso flujo, con base en la respuesta transitoria del mismo ante la presencia de un cambio en escalón del voltaje de entrada al actuador, se debe notar como por medio de la fifo 0 se fija el valor de consigna de 1 a 5 voltios desde la interfaz de usuario hacia el bloque comedi que se comunica con la servo válvula.

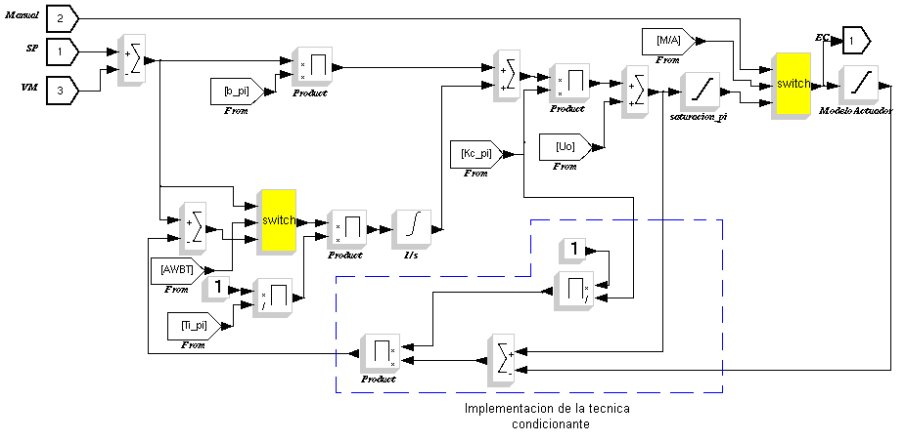
Ilustración 27. Diagrama de bloques Identificación del Lazo interno – Proceso Flujo



5.5.8 Diagrama de bloques del controlador PI para el lazo interno del sistema de control en cascada

En la ilustración 28 se indica el diagrama de bloques del controlador PI empleado como ley de control en el lazo interno del control en cascada, en él se puede apreciar los conmutadores de manual automático y de activación/desactivación de la estrategia AWBT, se muestra también la implementación de la técnica condicionante como estrategia AWBT.

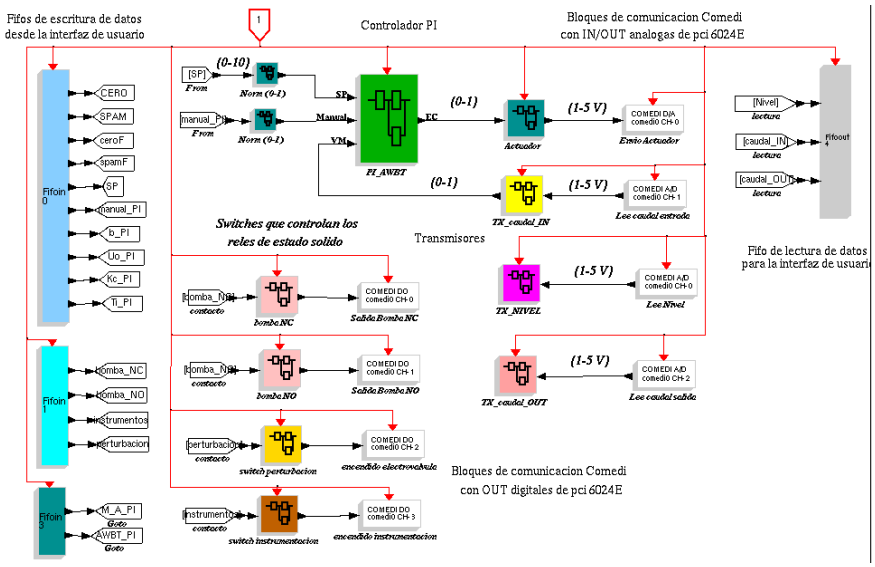
Ilustración 28. Diagrama de bloques que implementa el controlador PI



5.5.9 Diagrama de bloques del esquema realimentado con control PI para el lazo interno del sistema de control en cascada

En la ilustración 29 se muestra el diagrama de bloques de la ley de control PI AWBT, para el lazo interno, a través de la fifo 0, se reciben todos los parámetros del controlador así como los parámetros de configuración de los transmisores, la fifo 3 recibe las señales correspondientes a la conmutación de manual a automático y la activación de la estrategia AWBT.

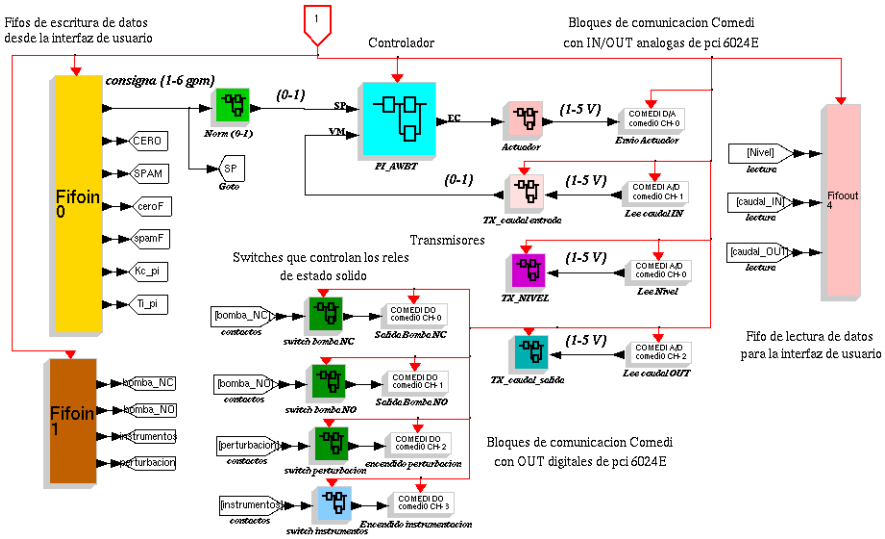
Ilustración 29. Diagrama de bloques del control PI para el lazo interno – Proceso Flujo



5.5.10 Diagrama de bloques de identificación del lazo externo del sistema de control en cascada

En la ilustración 30 se muestra el diagrama de bloques de identificación del lazo externo, la fifo 0 suministra los datos correspondientes a las constantes del lazo interno Kc_pi y Ti_pi, y los valores de cero y *span* para los controladores de nivel y caudal. Hay que resaltar en este diagrama de bloques, cómo para la identificación del lazo externo la ley de control PI hace parte de la planta a identificar condición necesaria para la implementación de un sistema de control en cascada (Sección 3.4).

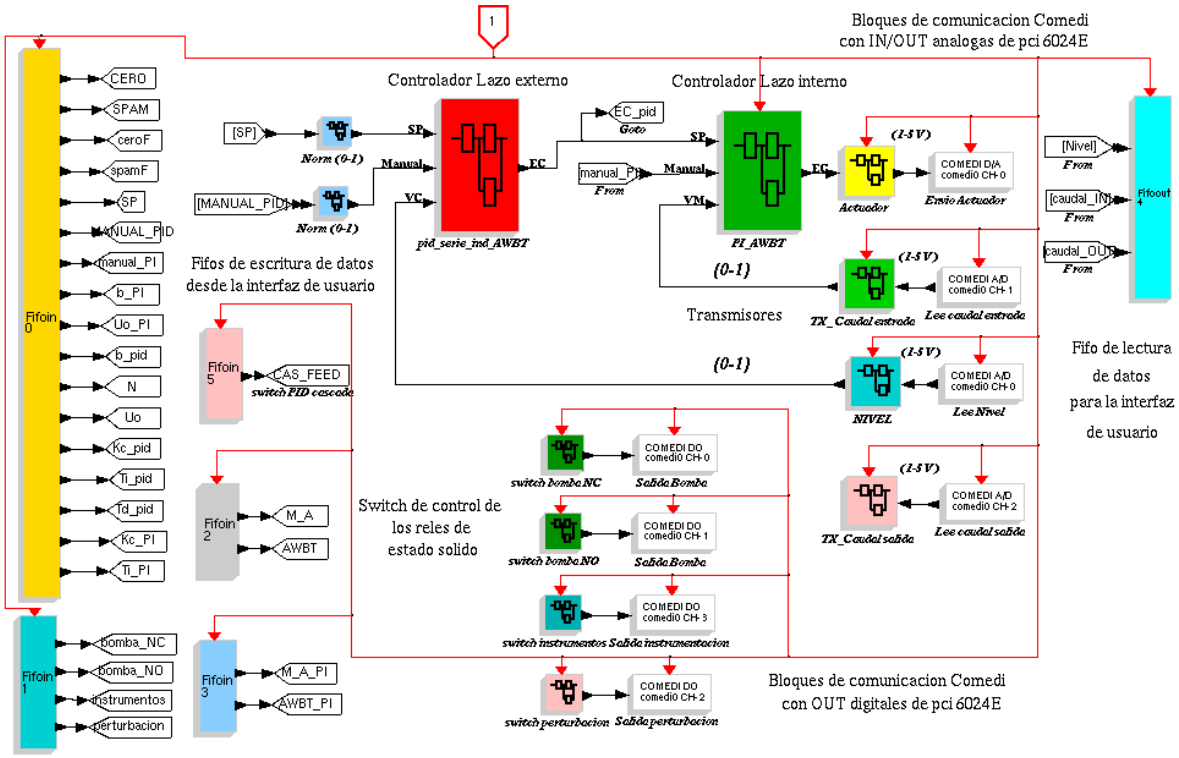
Ilustración 30. Diagrama de bloques de identificación del lazo externo – Proceso nivel



5.5.11 Diagrama de bloques de del sistema de control en cascada

La ilustración 31 muestra la implementación en diagrama de bloques del sistema de control en cascada, en el diagrama se observa el controlador interno PI (Bloque verde) y el controlador externo PID (Bloque rojo), claramente se ven los lazos de control interno, conformado por el bloque transmisor de caudal y el bloque actuador, se observa también las consignas manual y automática, notando que la consigna automática es el esfuerzo de control del PID. El lazo externo está compuesto por el transmisor de nivel y el controlador PID, también posee las consignas manual y automática, que provienen de la fifo 0. La fifo 0 también comunica los valores de las constantes de sintonización de los controladores PI y PID, los parámetros de configuración de los transmisores (cero y *span*) y los parámetros de optimización de los controladores (Uo, b, N en el caso del PID). En este diagrama de bloques se usan la fifo 2 y 3 para comunicar las señales discretas de cambio de los conmutadores de manual a automático y de activación / desactivación de la estrategia AWBT en los controladores. La fifo 5 permite comunicar desde la interfaz de usuario una señal discreta que conmuta el cambio de ley de control en cascada a ley de control PID, con esto se busca facilidad a la hora de la comparación de resultados.

Ilustración 31. Diagrama de bloques del sistema de control en cascada



5.6. METODOLOGÍA PARA LA SINTONIZACIÓN DEL CONTROL EN CASCADA

La metodología aquí realizada esta basada en una aproximación convencional de sintonización de lazos controlados por un control en cascada en donde el controlador primario o maestro es un controlador PID y el controlador secundario o esclavo es un controlador P o PI. Los pasos a seguir son los siguientes.

1. **Identificación proceso secundario:** con ambos controladores en manual realizar la identificación de los parámetros dinámicos del proceso secundario (ganancia, constante de tiempo, retardo si lo hay). Esta actividad se realiza introduciendo una señal de escalón a la entrada de dicho proceso.

2. **Selección de la estrategia de control del lazo secundario**: para realizar la selección de la estrategia de control del lazo interno se deben tener en cuenta los siguientes casos (Cooper, 2008).

   Caso 1. Si el proceso secundario no es por lo menos 3 veces más rápido que el proceso global, no es lo suficientemente rápido para un controlador PI.

   Caso 2. Si el proceso secundario es de 3 a 5 veces más rápido que el proceso global, entonces el control P tendrá un desempeño similar al control PI.

   Caso 3. Si la rapidez del proceso secundario es >5 veces que el proceso global, un controlador PI proporcionará mejor rendimiento.

3. **Sintonización controlador interno:** una vez obtenidas las constantes de tiempo del proceso y si se ha seleccionado un control PI, para su sintonización se debe elegir un método de acuerdo a las características de respuesta deseadas, ya sea por medio de lugar geométrico de raíces (LGR) o como en este caso se usaron métodos de sintonización basados en criterios de integral o robustez como los relacionados en (O'Dwyer, 1999), se debe comprobar la respuesta del proceso. Esto se realiza con el controlador maestro en manual. Para la discriminación de los métodos de sintonización ver el anexo B

4. **Identificación proceso primario:** luego de sintonizar el lazo interno se procede a realizar la identificación de los parámetros dinámicos del proceso primario con el controlador secundario en automático. Para ello se introduce un cambio en escalón y se obtienen los parámetros dinámicos de la respuesta del proceso. Es ideal aproximar la respuesta del sistema a un modelo de primer orden con tiempo muerto (POMTM).

5. **Selección y sintonización de la estrategia de control del lazo primario**: generalmente en un control en cascada el controlador externo o maestro utiliza un control PID. Se recomienda utilizar un controlador PID industrial y para su sintonización se debe tener en cuenta si se desea rechazo a perturbaciones (regulador) o seguimiento de la consigna (servomecanismo), para ello se pueden

utilizar los métodos de sintonización basados en criterios integrales, robustez como los mencionados en (Aström and Hägglund, 1984), (O'Dwyer, 1999), (Alfaro, 2002) y (Alfaro, 2003) y que fueron probados y discriminados en el anexo B.

6. Para la definición de los parámetros de identificación a partir de la respuesta transitoria (Ganancia Kc, Retardo L y constante de tiempo Tao), se usaron varios métodos gráficos de identificación, entre estos están el método de la tangente de Ziegler y Nichols, el método de dos puntos de Smith y el método de los tres cuartos de Alfaro, como se muestra en (Alfaro, 2001). Como se concluye en este estudio los mejores métodos de fácil aplicación son el método dos puntos de Smith y el método de los tres cuartos de Alfaro, el método de Ziegler y Nichols aunque es usado con frecuencia, según el estudio, es el que más error introduce debido a la subjetividad cuando se traza la tangente sobre la curva de reacción. Por lo tanto para este proyecto se usaron el método de Smith y el método de Alfaro para la identificación de los parámetros de la planta POMTM.

7. Con ambos controladores en automático se comprueba la respuesta del control en cascada introduciendo disturbios y verificando el comportamiento deseado del proceso.

5.7. COMUNICACIÓN ESPACIO DE USUARIO CON ESPACIO DE KERNEL

En la planta de tanques interactuantes el *software* de control e interfaz Hombre-Máquina se realiza por medio del software *Labview* (versión 7.0) de la *National Instruments*, ver ilustración 20. Por medio de dicha interfaz se realiza la visualización del comportamiento temporal de las diferentes variables de proceso: caudal, nivel y esfuerzo de control, permite introducir los valores para las ganancias proporcional, integral y derivativa para la sintonización del control PID, conmutar entre los modos manual y automático, fijar el valor de salida para el esfuerzo de control y modificar el valor deseado para la variable controlada. También observar gráficamente el valor del nivel en el tanque, y adicionalmente se puede seleccionar la variable a controlar ya sea nivel o flujo. A pesar de parecer completa la interfaz, presenta inconvenientes ya que el programa implementado en *Labview* no dispone de captura y análisis in situ de la curva de respuesta del proceso.

Por tales motivos se pretende implementar en el espacio de usuario la arquitectura que se presenta en la Ilustración 33.

Ilustración 32. Interfaz de Usuario inicial

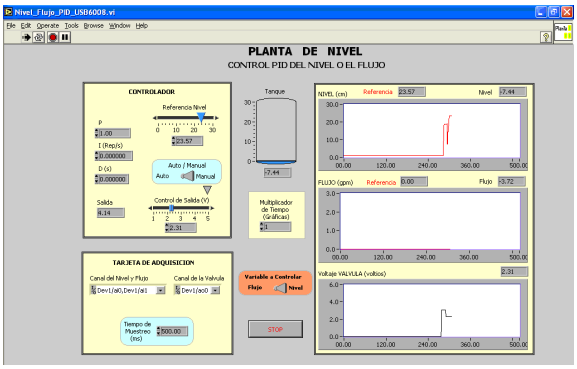
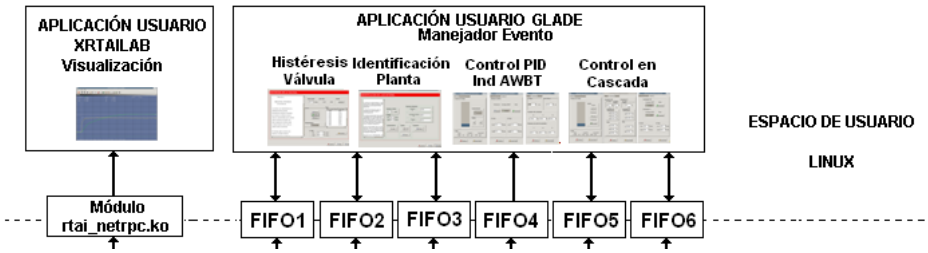


Ilustración 33. Arquitectura Sistema de Supervisión en espacio de Usuario Control en Cascada.



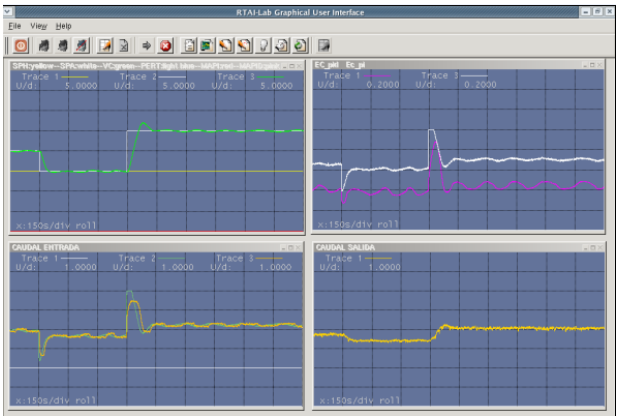
En la ilustración 33 se muestra que en el espacio de usuario estará corriendo el osciloscopio virtual Xrtailab y la interfaz de supervisión diseñada en Glade-GTK. En el espacio de *kernel* estarán corriendo las tareas de tiempo real generadas con base en los diagramas de bloques asignados a cada práctica de laboratorio. La comunicación entre espacio de usuario y espacio de *kernel* se realiza por medio de fifos (entre la interfaz de supervisión y las tareas) y por medio del modulo rtai_netrpc.ko (entre xrtailab y las tareas de tiempo real).

5.7.1. Interfaz de Visualización: XRTAILAB

En el espacio de usuario se utilizan dos interfaces. La primera interfaz es el osciloscopio *Xrtailab,* ilustración 34, el cual permite monitorear las señales de control y observar las tendencias de las variables del proceso en el tiempo, ver anexo F. Esta interfaz es muy versátil, puesto que da soporte grafico de usuario, permitiéndole interactuar con las

señales del proceso: realizar mediciones de tiempo y amplitud, almacenar datos, entre otras. *Xrtailab* se conecta desde el espacio de usuario usando el módulo de RTAI *rtai_netrpc.ko* con la tarea de tiempo real que se encuentra en espacio de *kernel*.

Ilustración 34. Interfaz de usuario Xrtailab



La comunicación entre las tareas de tiempo real y xrtailab, se realiza por medio del modulo rtai_netrpc.ko, provisto por RTAI-Lab, que se debe cargar antes de iniciar a ejecutar las tareas de tiempo real.

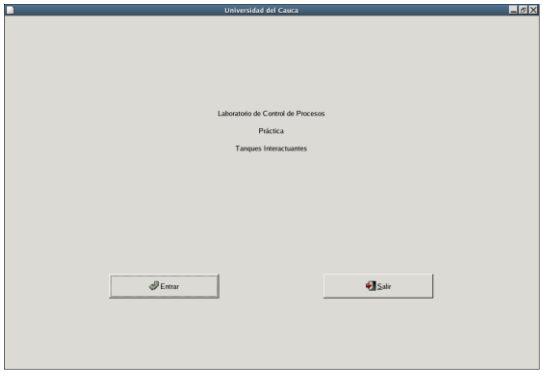
5.7.2. Interfaz de usuario

La segunda interfaz de usuario permite al estudiante realizar las diferentes prácticas de manera secuencial y por medio de ella seleccionar y sintonizar la ley de control, cambiar parámetros y constantes del sistema de cualquier tarea en línea. El software utilizado para el desarrollo de dicha interfaz es Glade-2.6.0 incorporado en la distribución de Linux Fedora Core 3, dicho software permite diseñar interfaces en forma de proyectos y permite a la vez generar el código en lenguaje C, C++ y Ada, modificándolo de acuerdo a las necesidades del proyecto. La comunicación de la interfaz diseñada con las tareas de tiempo real se realiza por medio de FIFOs *("First In- First Out")*, las cuales permiten escribir y leer desde y hacia la tarea de tiempo real. La interfaces desarrolladas en la herramienta de diseño se presentan a continuación:

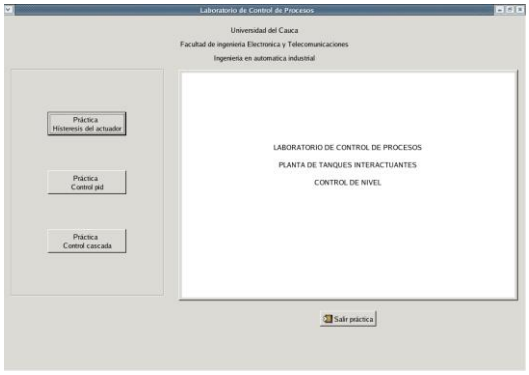
En las ilustraciones 35a, se muestran la ventana principal de la aplicación que consta, sencillamente de dos botones uno para continuar con la aplicación y otro para salir de ella. En la ilustración 35b se muestra la ventana de selección de prácticas, consta de tres

botones correspondientes a las practicas ***"Histéresis válvula", "Control PID"***, ***"Control en cascada"*** y un botón para volver a la ventana principal.

Ilustración 35. Ventana principal y de selección de prácticas



a) Ventana Principal de la interfaz



b) Ventana selección de prácticas

En la Ilustración 36, se indica la interfaz para la ejecución de la práctica de la Histéresis de la válvula, desde ella el usuario activa toda la instrumentación del sistema por medio del botón ***"Encender"***, también permite insertar el valor manual hacia el actuador en un rango de 1 a 5 voltios, los datos capturados se almacenan en la tabla ***"Datos"*** con el fin de obtener los puntos para graficar la histéresis del actuador. Los botones ***"Guardar"***, ***"Modificar"***, ***"Eliminar"***, *permiten modificar cualquier dato de la tabla. Desde esta ventana se puede energizar la bomba por medio de los botones* ***"ON"***, ***"OFF"***. El botón ***"Actualizar"*** sirve para enviar el dato de voltaje al actuador, cada vez que se modifique por parte del usuario, en la parte izquierda de la ventana se presenta una explicación acerca del fenómeno de la histéresis y las pautas para utilizar la ventana con RTAI. A esta ventana se accede desde la ventana de prácticas ilustración 35b pulsando el botón ***"Histéresis Válvula"***.

Ilustración 36. Ventana práctica histéresis de la válvula

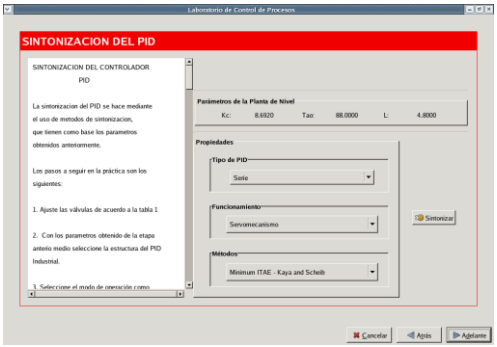
En la ilustración 37a se presenta la interfaz de identificación de la planta, a la cual se accede desde la ventana de prácticas ilustración 35b, pulsando el botón ***"Control PID"***. Igual que en la interfaz anterior existe el botón ***"Encender"*** que energiza la instrumentación del sistema, y los botones ***"ON"*** , ***"OFF"*** para el suministro de potencia a la bomba, estos botones, antes mencionados, aparecen en gran parte de las ventanas de la aplicación, por lo que no se volverán a mencionar. Desde esta ventana se envía un escalón a la planta que oscila en el rango de 0 a 6 gpm, por medio del botón ***"Enviar"***, se transmite el dato del escalón y los valores del cero y *span* a la tarea de tiempo real. Las entradas de texto ***"Kc"***, ***"Tao"***, ***"L"*** permiten insertar al sistema las constantes obtenidas en la identificación y por medio del botón ***"Guardar"*** el valor de las mismas queda disponible para sintonizar el controlador. Al igual que en la ventana de observación de la histéresis, en la parte izquierda de esta ventana también hay una ayuda para el usuario, acerca de la identificación de los procesos.

Por medio de la interfaz que se presenta en la ilustración 37b, a la cual se llega pulsando el botón ***"Adelante"*** de la ventana anterior, se realiza la sintonización del control PID industrial AWBT, ya sea serie o paralelo, al ingresar a esta interfaz se cargan automáticamente los valores de los parámetros identificados de la planta, esto también sirve como verificación de los datos insertados al sistema. Para sintonizar la ley de control PID se deben seleccionar: el tipo de PID (serie o paralelo), el funcionamiento del PID (Regulador o servomecanismo) y el tipo de método de sintonización. De acuerdo al anexo B se seleccionaron tres métodos de cada ley de control y funcionamiento para el uso en la interfaz, después de haber seleccionado estos tres ítems, se debe pulsar el botón

sintonizar e internamente se calcularan las constantes a partir de los parámetros identificados de la planta POMTM.

Ilustración 37. Ventanas Identificación de la planta – Sintonización Control PID AWBT

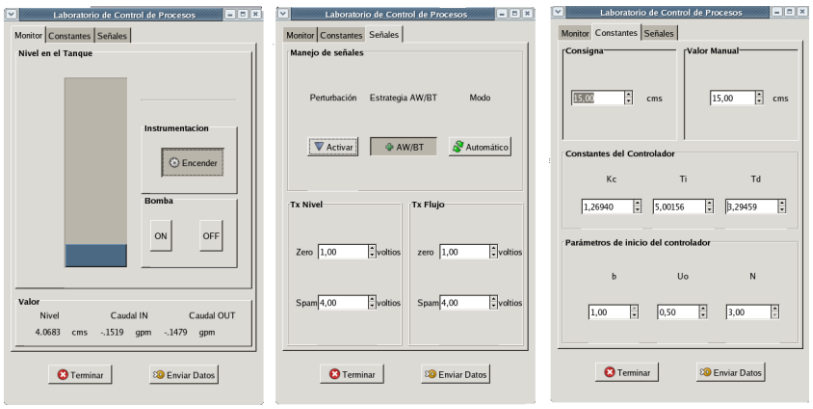
a) **Identificación Planta**



b) **Sintonización Control PID AWBT**

Una vez se seleccionan las opciones, por medio del botón ***"Sintonizar"*** automáticamente se carga la interfaz de la ilustración 38, en donde aparecen la constantes de sintonización calculadas de acuerdo al método seleccionado de la ventana anterior junto con los restantes parámetros, dados por defecto, del PID AWBT (Uo=0,5, b=1 y N=3). El valor de las consignas manual y automática varían de 0 a 30 cms. Esta ventana permite mostrar el nivel en el tanque por medio de un grafico animado, así como ver los valores numéricos del caudal de salida, entrada y el nivel. Para comunicar los valores de los parámetros del controlador a la tarea se usa el botón ***"Enviar Datos",*** este escribe la fifo para que la tarea los lea de esta en el espacio de kernel.

Ilustración 38. Ventana parámetros práctica control PID AWBT

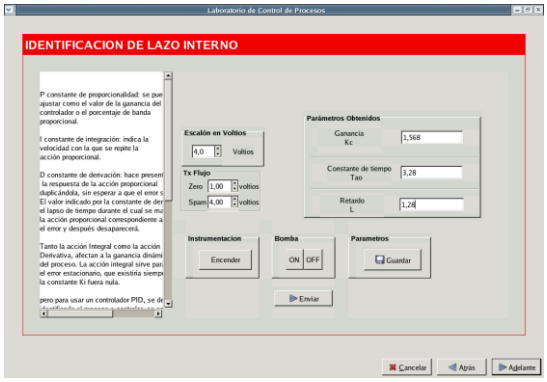


En esta interfaz de la ilustración 38 se observan tres nuevos botones, su uso es el siguiente, el botón ***"Automático"*** es el botón que permite conmutar el controlador entre

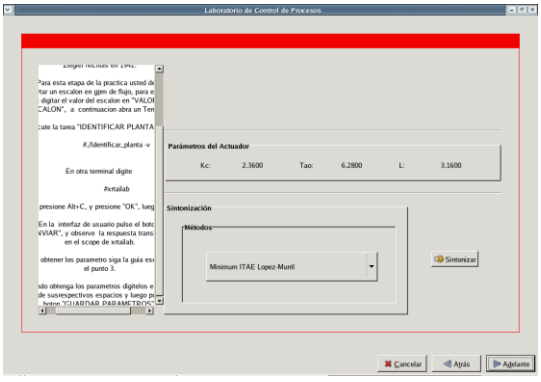
manual y automático, el botón ***"Activar"*** enciende la electroválvula con lo cual se aplica un disturbio al caudal de entrada, el botón ***"Ejecutar"*** activa la estrategia AWBT implementada en el controlador por medio de la técnica Condicionante. Estos tres botones aparecen en las ventanas de supervisión de la práctica, por lo tanto no se volverán a citar. Finalmente, la interfaz de la ilustración 38 contiene un botón ***"Terminar",*** el cual permite salir de esta ventana de supervisión y quedar en la ventana de sintonización ilustración 37b, para permitir la selección de otro método de sintonización u otro tipo de modo (regulador o servo). Se debe prestar atención a cual tarea se está ejecutando en el SOTR, a la hora de seleccionar el tipo de controlador (Serie o paralelo). Si no se desea continuar la práctica simplemente se debe pulsar el botón ***"Adelante"***.

La ilustración 39a muestra la ventana de identificación del lazo externo, el rango de valor del escalón a la entrada del sistema varía entre 1 y 5 voltios, los demás botones cumplen la misma función que en la ilustración 36a. A esta ventana se accede desde la ventana de prácticas pulsando el botón ***"Control en cascada".*** En la ilustración 39b se pueden apreciar los parámetros de identificación insertados en la ventana anterior y los métodos que de acuerdo al anexo B permiten una mejor sintonización de un control PI, igual que para la ley de control PID solo se seleccionaron los métodos más representativos según los resultados presentados en el mencionado anexo. El botón ***"Sintonizar"***, hace que el sistema calcule las constantes de sintonización internamente y las cargue en la ventana mostrada en la ilustración 39b. Dicha ilustración permite la sintonización de la ley de control PI, a esta ventana se accede pulsando el botón ***"Adelante"***, de la ventana mostrada en la ilustración 39a.

Ilustración 39. Ventana identificación y sintonización lazo interno (Proceso flujo)



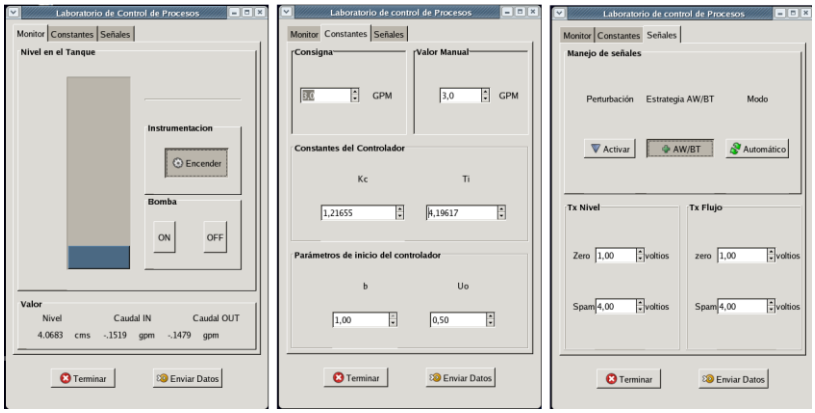
a) **Identificación lazo interno**



b) **Sintonización Lazo interno**

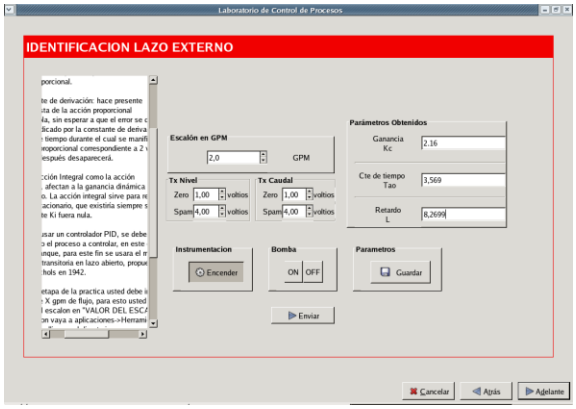
En la ilustración 40 se presenta la ventana de supervisión del controlador PI, los botones cumplen las mismas funciones de la ilustración 38, la diferencia radica lógicamente en que no aparece el termino derivativo y por lo tanto tampoco el valor de la constante del filtro del término derivativo, también en el valor de las consignas manual y automática que varían en un rango de 0 a 6 gpm, los cuales son aplicados directamente al actuador en la tarea de tiempo real.

Ilustración 40. Ventana control PI para lazo secundario



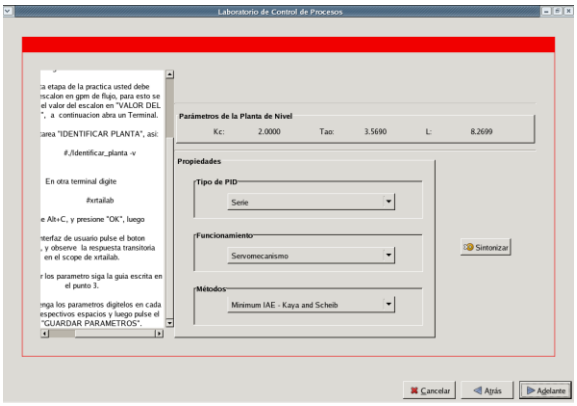
La ilustración 41 muestra la ventana de identificación del lazo externo, para este caso se puede variar el escalón de consigna entre 0 y 6 gpm, además permite almacenar los parámetros dinámicos al igual que algunas ventanas, también tiene una ayuda en la parte izquierda que indica al usuario lo concerniente a la actividad que desarrolla.

Ilustración 41. Ventana identificación lazo externo



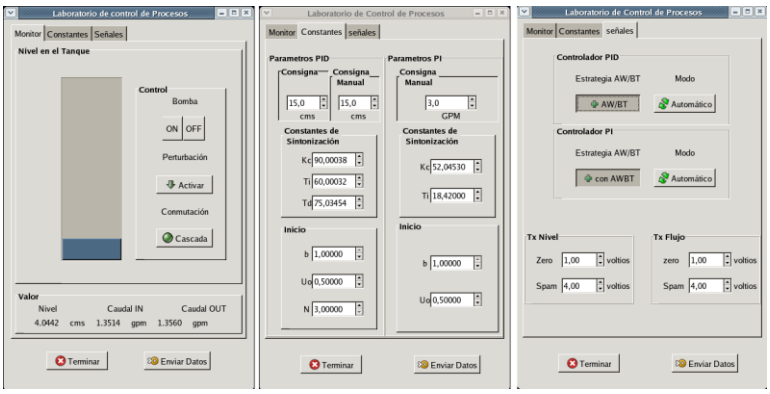
La ilustración 42 muestra la ventana de sintonización del controlador del lazo externo del sistema de control en cascada, en ella se pueden apreciar los parámetros insertados en la ventana anterior, también permite seleccionar el tipo de controlador, el funcionamiento y el método. Al pulsar el botón ***"Sintonizar"***, el sistema calcula las constantes y abre la ventana mostrada en la ilustración 43.

Ilustración 42. Ventana de sintonización del controlador del lazo externo- Control PID



La ilustración 43 muestra la ventana que permite interactuar con un sistema de control en cascada, La mayoría de los botones se han explicado anteriormente, el único botón que no se ha explicado es el botón ***"Cascada"***, este botón permite, conmutar la ley de control de un PID en lazo simple a un PID en cascada y viceversa, sin embargo el cambio no es inmediato debido a que para el control en lazo simple, se deben insertar las constantes obtenidas en la práctica de control PID, de lo contrario el PID en lazo simple no tendrá un buen funcionamiento.

Ilustración 43. Ventana Control en Cascada



6. INTEGRACIÓN Y EVALUACIÓN DEL SISTEMA DE CONTROL PROPUESTO

6.1. INTEGRACIÓN DEL SISTEMA

Luego de realizado el diseño e implementación del sistema de control propuesto, se procede a realizar la comunicación vía hardware de la planta con el sistema de control (PC-SOTR-PLANTA). En la ilustración 44 se presenta el sistema final, en dicha ilustración se aprecia la planta con su sistema hidráulico e instrumentación, el tablero de conexiones y el PC donde se ejecuta el sistema de control de tiempo real.

Ilustración 44. Sistema final planta de nivel y sistema de control

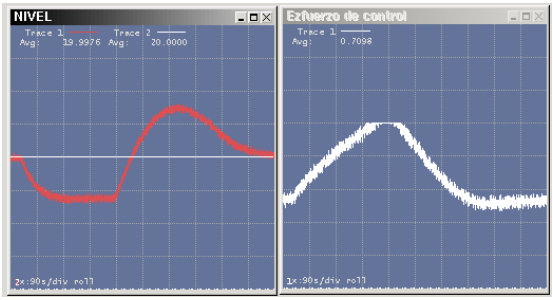
Para la comunicación del sistema de tiempo real con el planta se diseño un circuito de mando y comunicaciones que se implemento en un tablero o panel de señales mostrado en la ilustración 17e, el diagrama de conexión de este panel se muestra en el anexo H. Diversas tareas se llevaron a cabo en el diseño e instalación de las diversas interfaces eléctricas con el sistema de tiempo real, entre las más relevante están:

Interfaz tarjeta PCI 6024E – Circuito de mando: Identificación del voltaje y corriente umbral para los relés de estado sólido marca CX240D5 para su activación desde el PC y comandar la activación de la electro bomba, la servo válvula y la electro válvula. Se

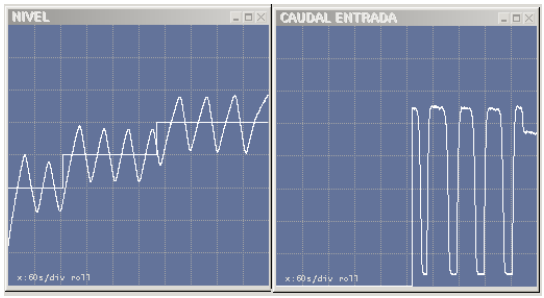
encontró un voltaje mínimo de 3 V y 30 mA, por lo que se fijo un voltaje de activación de 5 voltios con una corriente de 13 mA. Sin embargo dado que la tarjeta no entrega más de 4 mA, se emplea un adaptador de potencia con un ULN 2803. Se verifico el normal funcionamiento del diseño en protoboard, creando una tarea de tiempo real que se comunica por medio de la tarjeta PCI para enviar una señal discreta, que efectivamente activa una carga en eléctrica de potencia por medio del relé de estado sólido. Hecho esto se procedió a cablear el panel de adaptación de acuerdo al esquema mostrado en el anexo H, se hicieron pruebas de continuidad y normal funcionamiento obteniéndose los resultados esperados.

Interfaz tarjeta PCI 6024E – Instrumentos de 4-20 mA: Las señales de 4-20 mA provenientes del transmisor de nivel, del transmisor de caudal de entrada y del transmisor de caudal de salida, se convierte en voltajes de 1-5 v por medio de resistencias de 250 ohmios estos voltajes se aplican a entradas ADC de la tarjeta PCI 6024E por medio de la placa de conexiones CB 68-LP NI. La configuración de la comunicación entre las anteriores variables del proceso con los bloques comedi para la adquisición de las señales con la placa se aprecia en la tabla 4. Para probar el correcto funcionamiento en la toma de los datos se creó una tarea de tiempo real que permitiera leer y desplegar los voltajes adquiridos por la tarjeta. En esta prueba el esfuerzo de control aplicado a la válvula se generaba desde una fuente de alimentación externa a la tarjeta, los voltajes asociados a los variables de proceso de los tres transmisores se leyeron y registraron sin ningún problema. El último paso en estas pruebas de integración fue crear una tarea de tiempo real que permitiera manualmente variar el esfuerzo de control aplicado a la servo válvula, adquirir simultáneamente los datos provenientes de los transmisores y adicionalmente, en esta misma tarea, debía ser posible activar y desactivar cada uno de los 4 relés de estado sólido. La prueba se realizo con éxito, por lo que con esto se finalizaron las pruebas de integración del sistema y se concluía que todo operaba de acuerdo a lo esperado. Sin embargo en la adquisición de señales de los transmisores se presentaron problemas de ruido como se muestra en la ilustración 45a. Se realizaron diversas pruebas para determinar la fuente del mismo, encontrándose que provenía de los mismos transmisores, posiblemente debido al uso de la fuente de 24 v de la servo válvula. Se optó por diseñar un filtro digital de primer orden para la supresión del ruido, se implemento este filtro y se obtuvieron los resultados mostrados en la ilustración 45b.

Ilustración 45. Presencia de ruido en la adquisición de señales



a) Presencia de ruido



a) Supresión del ruido

Tabla 4. Configuración comunicación variables del proceso

VARIABLE	TIPO	BLOQUE COMEDI	PLACA CB 68-LP NI	
			PIN +	PIN -
Nivel	Analógica (AI)	COMEDI A/D comedi0 CH 0	68	67
Caudal de Entrada	Analógica (AI)	COMEDI A/D comedi0 CH 1	33	32
Caudal de Salida	Analógica (AI)	COMEDI A/D comedi0 CH 2	65	64
Esfuerzo de Control Manual o Automático (Voltaje válvula)	Analógica (DAC 0)	COMEDI D/A comedi0 CH 0	22	54
Señal Normalmente Cerrado (NC) Bomba	Digital	COMEDI DO comedi0 CH 0	49	50
Señal Normalmente Abierto (NO) Bomba	Digital	COMEDI DO comedi0 CH 1	17	18
Señal Activación de Disturbio	Digital	COMEDI DO comedi0 CH 2	52	53
Señal de Activación Instrumentos	Digital	COMEDI DO comedi0 CH 3	47	9

6.2. PRUEBAS Y RESULTADOS

Pruebas de latencia de RTAI: La primera prueba se realiza al sistema operativo de tiempo real. RTAI presenta una serie de comandos para medir la latencia del SOTR en el espacio de usuario y en el espacio de *kernel*. Esta prueba sirve para verificar el

desempeño del sistema y comprobar que es apto para aplicaciones de tiempo real. Para ejecutar la prueba de latencia se debe acceder al ejecutable de la prueba; en un terminal se ejecutan los siguientes comandos:

cd /usr/realtime/testsuite/user/latency (para el espacio del usuario)

./run Este comando corre la tarea y con *Ctrl+C* se para la prueba.

La salida del sistema es:

RTAI latency calibration tool
period = 100000 (ns)
average time = 1 (s)
use the FPU
start the timer
timer_mode is oneshot

RTAI Testsuite - USER latency (all data in nanoseconds)
2008/08/28 12:02:18

RTH	lat min	ovl min	lat avg	lat max	ovl max	overruns
RTD	3352	3352	6764	12571	12571	0
RTD	3352	3352	7125	19276	19276	0
RTD	3352	3352	6867	15924	19276	0
RTD	3352	3352	6778	15924	19276	0
RTD	3352	3352	6782	11733	19276	0
RTD	3352	3352	6814	19276	19276	0
RTD	3352	3352	6884	20952	20952	0
RTD	3352	3352	6846	20952	20952	0
RTD	3352	3352	6810	18438	20952	0
RTD	3352	3352	7022	15924	20952	0
RTD	3352	3352	6884	20952	20952	0
RTD	3352	3352	6961	21791	21791	0
RTD	3352	3352	7021	23467	23467	0
RTD	3352	3352	6854	18438	23467	0
RTD	3352	3352	6824	20114	23467	0
RTD	3352	3352	6879	21791	23467	0
RTD	3352	3352	6792	14248	23467	0
RTD	3352	3352	7030	20114	23467	0
RTD	3352	3352	6884	20952	20952	0
RTD	3352	3352	6899	19276	23467	0
RTD	3352	3352	6830	19276	23467	0

En donde:

lat min = latencia mínima del sistema
ovl min = *(overall)*, latencia general mínima del sistema
lat avg = latencia promedio
lat max = latencia máxima del sistema
ovl max = *(overall)*, latencia general máxima del sistema
overruns = Numero de periodos que no han sido suficientes para la atención de un evento.

En el espacio de kernel la prueba de la latencia se ejecuta con los siguientes comandos:

cd /usr/realtime/testsuite/kern/latency (para el espacio del kernel)

./run

La salida del sistema es:

RTAI latency calibration tool
period = 100000 (ns)
avrgtime = 1 (s)
do not use the FPU
start the timer
timer_mode is oneshot

RTAI Testsuite - KERNEL latency (all data in nanoseconds)

RTH	lat min	ovl min	lat avg	lat max	ovl max	overruns
RTD	2514	2514	6102	11733	11733	0
RTD	2514	2514	5992	11733	11733	0
RTD	2514	2514	6014	16762	16762	0
RTD	2514	2514	5975	13410	16762	0
RTD	2514	2514	5965	13410	16762	0
RTD	2514	2514	6054	20114	20114	0
RTD	2514	2514	6056	19276	20114	0
RTD	2514	2514	6052	15086	20114	0
RTD	2514	2514	6050	20114	20114	0
RTD	2514	2514	5972	20114	20114	0
RTD	2514	2514	5990	16762	20114	0
RTD	2514	2514	5988	20952	20952	0
RTD	2514	2514	5983	14248	20952	0
RTD	2514	2514	6021	13410	20952	0
RTD	2514	2514	5975	13410	16762	0
RTD	2514	2514	5988	14248	20952	0
RTD	2514	2514	5978	14248	20952	0
RTD	2514	2514	5966	17600	20952	0
RTD	2514	2514	5979	20114	20952	0

Por medio de la prueba se mide la diferencia entre el tiempo del momento que el evento ocurre y el momento en que éste es tratado o que en realidad es llamado por el planificador. Los resultados indican que en espacio de usuario la latencia mínima es de 3352 ns y la latencia máxima de 23467 ns y en espacio de *kernel*, 2514 ns para la mínima latencia y 20952 ns para la máxima latencia.

Es por ello que se considera a RTAI como un sistema apto para la implementación de tareas de tiempo real. Esta prueba se debe realizar antes de implementar tareas de tiempo real con el fin de analizar posibles valores indeseados en los resultados de latencias. Además se debe correr la prueba por un periodo largo de tiempo aproximadamente superior a 2 horas como lo recomiendan los desarrolladores.

Prueba RTAI-Lab / PCI 6024E / Control PI de planta RC: La segunda prueba del sistema consiste en el diseño del control, simulación y ejecución en tiempo real de un sistema de control realimentado PI para un circuito eléctrico RC, el diseño del control, la simulación y los resultados obtenidos en la ejecución de tiempo real se pueden ver en (Flórez, Cabezas y Díaz, 2008a), en donde se comprueba la eficiencia del sistema RTAI-Lab.

Prueba RTAI-Lab / PCI 6024E / Control Cascada de planta RC: La tercera prueba del sistema consiste en el diseño de un control en cascada para un circuito eléctrico RC, que representa un proceso y un segundo sistema RC de menor constante de tiempo que representa la dinámica de un actuador. Dicho control en cascada se diseña con el fin de minimizar el efecto de los disturbios en la variable manipulada que para este caso es el voltaje de salida del circuito RC que representa al actuador. El anterior análisis se realiza comparando la respuesta de rechazo de las perturbaciones entre un sistema de control *realimentado* y el control en cascada, ver (Flórez, Cabezas y Díaz, 2008b).

Prueba RTAI-Lab / PCI 6024E / Control PID Windup: La cuarta prueba consiste en la validación del control PID Industrial AWBT con la planta caso de estudio ante la presencia del fenómeno *windup*. Mediante este experimento se analiza la respuesta transitoria de la variable controlada (VC) ante la presencia de una gran disturbio en la variable manipulada

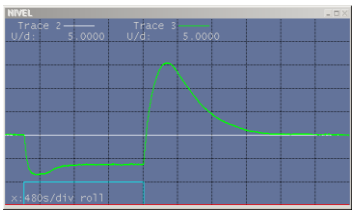
(VM), consistente en una caída en el caudal de entrada de aproximadamente 2 gpm durante 1680 s, estando el sistema en estado estable con VC = 15 cm y el control PID sin activar la estrategia AWBT, en la segunda parte se introduce el disturbio bajo las mismas condiciones, pero esta vez el PID tiene activada la estrategia AWBT. El control utilizado es el PID serie AWBT sintonizado con el método *Minimun ITAE de Kaya & Sheib* como regulador, descrito en el Anexo B; los valores de las constantes de sintonización se pueden apreciar en la tabla 5. La tarea de tiempo real para el PID serie AWBT se crea del diagrama de bloques de la ilustración 26. La ilustración 46 representa la respuesta del sistema sin la activación del AWBT, estas gráficas fueron tomadas con el osciloscopio Xrtailab, en todas, a), b), y c), el eje de tiempo se encuentra en 480 s por división, en la ilustración 46a para la VC el eje de las ordenadas es de 5 cm por división, en la ilustración 46b para el EC el eje de las ordenadas es de 0.2 por división y en la ilustración 46c para la VM el eje de las ordenadas equivale a 1 gpm por división. En 46a se observa como el sistema, antes del disturbio, se encuentra en estado estable con VC en un valor de 15 cms (traza verde) siendo el valor de consigna de 15 cms (traza blanca), el EC se mantiene en 0.6 (traza violeta) ver ilustración 46b y el caudal, esto es la VM, se encuentra aproximadamente en 2,5 gpm, el disturbio ocurre en la VM (traza naranja) durante 1680 s, ver traza azul claro en las tres ilustraciones. Se puede apreciar en las ilustraciones 46a y 46b como el disturbio es tan crítico que el nivel en el tanque se perturba cayendo abruptamente, esto a pesar de que el controlador toma los correctivos solicitándole al actuador que deje pasar todo el caudal que pueda, este responde entrando en saturación pero, a pesar de esto, el nivel en el tanque cae a un valor cercano de 9 cms. Durante el tiempo que dura el disturbio el componente integral del PID (sin AWBT) continua llevando a cabo su acción de acumulación del error. Cuando el disturbio desaparece, la componente integral ha realizado un reajuste tan excesivo que hace que el actuador continúe saturado y la VC presente un sobresalto alrededor del valor de consigna, para después de un tiempo alcanzar el estado estable una vez se cancele el error acumulado por la acción integral. La duración del disturbio es de 1680 s, la perturbación generada en la VC fue tan critica que el controlador PID (sin AWBT) permanece saturado durante 120 s adicionales una vez desaparece el disturbio, tiempo en el cual la salida del controlador comienza a descender y en ese momento también se observa el cambio de pendiente en la curva de la VC hasta que finalmente el sistema se estabiliza nuevamente en 15 cms. La ilustración 46a evidencia como la presencia de *windup* afecta en gran medida la VC. En

cambio cuando se activa la estrategia AWBT en el PID, el sistema, de igual manera, se perturba cayendo el nivel a 9 cms, la diferencia es que, cuando la perturbación desaparece, el transiente generado es de menor sobrepaso y dura menos tiempo antes de estabilizarse nuevamente, la planta y el actuador sólo está saturado mientras la perturbación dure, el efecto se aprecia en la ilustración 47 en donde se demuestra la efectividad de la técnica condicionante como técnica AWBT.

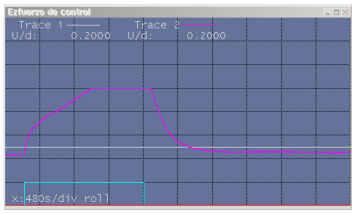
Tabla 5. Constantes de sintonización PID AWBT como regulador

ESTRATEGIA	MODO	MÉTODO	KC	TI	TD
PID SERIE AWBT	REGULADOR	ITAE KAYA & SHEIB	0.61552	311.523	5.64596

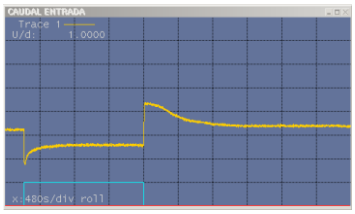
Ilustración 46. Respuesta PID Serie sin AWBT en presencia de un disturbio



a) VC (traza verde), SP (blanco), duración disturbio (azul claro)

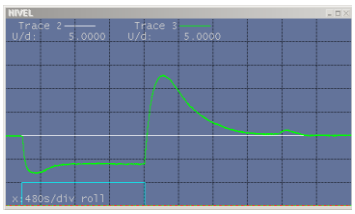


b) EC (violeta), duración disturbio (azul claro)

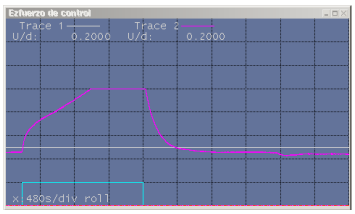


c) VM (naranja), duración disturbio (azul claro)

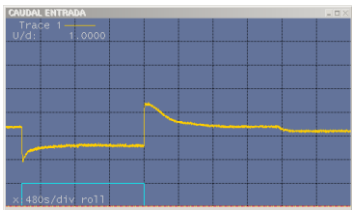
Ilustración 47. Respuesta PID Serie con AWBT en presencia de un disturbio



a) VC (traza verde), SP (blanco), duración disturbio (azul claro)



b) EC (violeta), duración disturbio (azul claro)



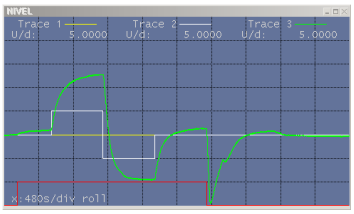
c) VM (naranja), duración disturbio (azul claro)

Prueba RTAI-Lab / PCI 6024E / Control PID Bump Transfer: La quinta prueba consiste en la validación del control PID Industrial AWBT con la planta caso de estudio ante la presencia del fenómeno *bump transfer*. Mediante este experimento se analiza la respuesta transitoria de la variable controlada (VC), durante el cambio de automático a manual y viceversa. Estando el sistema en estado estable VC = 15 cms, el control PID sin estrategia AWBT y la consigna tanto manual como automática en 15 cms, se procede a conmutar el modo de control de automático a manual durante 2400 segundos para posteriormente regresar a modo automático. Estas condiciones se repiten nuevamente pero activando la técnica AWBT. El controlador utilizado es un PID industrial de estructura serie AWBT sintonizado con el método *Minimun ITAE de Kaya & Sheib* como servomecanismo, tal como se describe en el Anexo B; los valores de las constantes de sintonización se pueden apreciar en la tabla 6, estos se obtienen a partir de los parámetros identificados de un modelo POMTM realizado previamente a la planta, ver anexo I para mayores detalles. La tarea de tiempo real para el PID serie AWBT se crea a partir del diagrama de bloques de la ilustración 26.

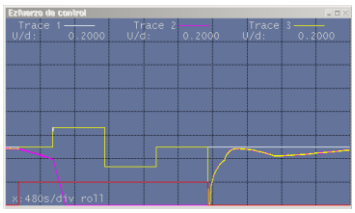
En la ilustración 48a y 48b el eje temporal presenta un valor de 480 segundos por división, para la 48a el eje de las ordenadas presenta un valor de 5 cms por división y para 48b el eje de las ordenadas presenta un valor de 0.2 por división. En la ilustración 48 el PID no tiene habilitado el AWBT, el sistema, estando en modo automático, se encuentra con VC = 15 cms (traza verde), un valor de consigna de 15 cms (traza amarilla) y un EC = 0.5 (traza violeta). Cuando se realiza la conmutación a modo manual (traza roja), el EC en manual es mayor que el EC en automático, por lo la VC comienza a aumenta de valor hasta que se estabiliza en 11 cms. Durante la duración del modo manual el EC automático cae abruptamente saturándose por abajo, esto es debido a que no se tiene una estrategia AWBT habilitada. Después de 2400 segundos nuevamente se conmuta de manual a automático y, ya que en este modo el EC se encuentra saturado en su límite inferior, el controlador le ordena al actuador que no deje pasar caudal a la planta, por lo que se observa una notable caída de la VC, cayendo el nivel a un mínimo de 3 cms, después de unos segundos el controlador se recupera e inicia el proceso de corrección del error hasta que finalmente se estabiliza a los 960 s de producirse la conmutación. Pero lamentablemente se ha producido una perturbación no deseable en la VC, por lo que se aprecia en la ilustración 48a como el efecto *bump transfer* perturba en gran medida la

VC. Cuando la estrategia AWBT en el controlador PID se encuentra activa, se observa en la ilustración 49b como el EC en automático, durante el modo manual, no se satura sino que permanece constante, realmente el AWBT intenta seguir, lo mejor que puede, las variaciones del EC manual (traza amarilla). Estando en el modo manual el sistema se estabiliza en 11 cms, a los 2400 s se realiza la conmutación de manual a automático, pero en esta ocasión, el controlador en modo automático no esta saturado, se encuentra en un valor estable por debajo del valor en manual, por lo que se le facilita realizar su trabajo de regresar nuevamente la VC a la consigna de 15 cm a pesar de hacerlo con un transitorio. Sin embargo la VC ni el EC hacia el actuador sufrieron un gran salto durante la transferencia por lo que la perturbación fue mínima. El efecto se aprecia en la ilustración 49 en donde se demuestra nuevamente la eficiencia de la técnica condicionante como técnica AWBT.

Ilustración 48. Respuesta PID Serie sin AWBT ante un cambio de manual a automático

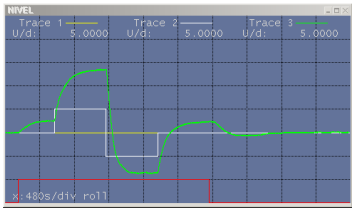


a) VC (traza verde), SP (amarillo), SP Manual (blanca), Conmutación A-M-A (rojo)

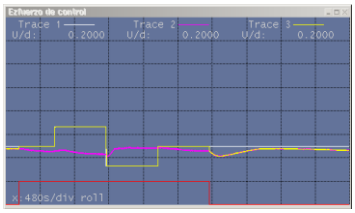


b) EC Manual (amarillo), EC Auto (violeta), Conmutación A- M-A (rojo)

Ilustración 49. Respuesta PID Serie con AWBT ante un cambio de manual a automático



a) VC (traza verde), SP (amarillo), SP Manual (blanca), Conmutación A-M-A (rojo)



b) EC Manual (amarillo), EC Auto (violeta), Conmutación A- M-A (rojo)

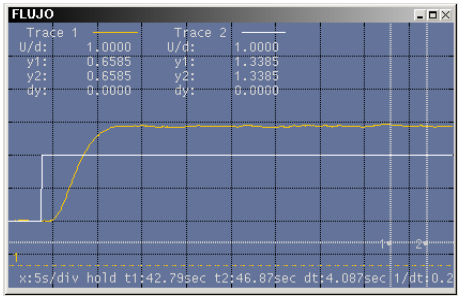
Tabla 6. Constantes de sintonización PID AWBT como servomecanismo

ESTRATEGIA	MODO	MÉTODO	KC	TI	TD
PID SERIE AWBT	SERVOMECANISMO	ITAE KAYA & SHEIB	0.52238	82.98480	4.50457

Prueba RTAI-Lab / PCI 6024E / Control en Cascada: En la sexta prueba se comprueba la capacidad de rechazo ante la presencia de perturbaciones del control en cascada diseñado a partir de la ilustración 33. En este experimento se realiza la sintonización aplicando la metodología descrita en la sección 5.7. La variable secundaria o interna es el caudal de entrada a la planta controlado por un PI AWBT y la variable primaria o externa es el nivel en el tanque controlado por un PID AWBT.

El primer paso es realizar la identificación del lazo secundario, esto se realiza aplicando un escalón de 2 voltios al actuador mientras el caudal se encuentra en un valor estable de 2 gpm, la respuesta de la VM se observa en la ilustración 50. En esta ilustración el eje de tiempo esta en un valor de 5 s por división y el eje de las ordenadas caudal (traza naranja) en un 1 gpm por división, en esta misma imagen se despliega el escalón voltaje (traza blanca) configurado en 1 voltio por división. Los parámetros dinámicos identificados de la planta se obtienen por medio de un algoritmo ejecutado en Scilab, el cual a partir de los valores de la respuesta y de la excitación calcula los parámetros K, tao y L, estos se aprecian en la tabla 7. Ver anexo I para mayores detalles.

Ilustración 50. Identificación Lazo Interno



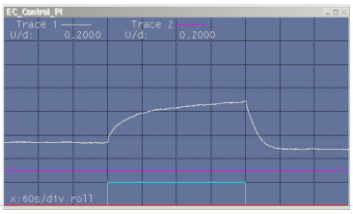
VM (traza naranja)
Excitación (blanca)

Con base en los parámetros conseguidos se sintoniza el control PI AWBT por medio del método IAE Rovira et all como servomecanismo y se procede a verificar su respuesta dinámica. En la ilustración 51 se aprecia la respuesta del lazo interno ante un disturbio en

el caudal, consistente en una caída de 1 gpm. En 51 a y b el eje temporal presenta 60 s por división, en 51a el eje de las ordenadas presenta 1 gpm por división y para 51b es de 0.2 por división. En 51a se aprecia la respuesta del lazo interno ante la presencia de un disturbio, antes del disturbio el sistema se encontraba estable con una VC de 4 gpm, un SP de 4 gpm y un EC de 0.56, el disturbio genera un descenso en el valor de la VC, pero rápidamente es atendido por el controlador PI AWBT por lo que la caída se corrige y el sistema se estabiliza nuevamente en 4 gpm, sin embargo para que esto suceda el EC debió subir a 0.84 con el fin de ordenarle al actuador que entregue mas caudal. En la ilustración 51b se muestra el EC (color violeta) ordenado por el control PI para rechazar el disturbio generado en la VC y de esta manera estabilizar, después de un corto periodo de tiempo, el sistema.

Ilustración 51. Respuesta del Lazo interno

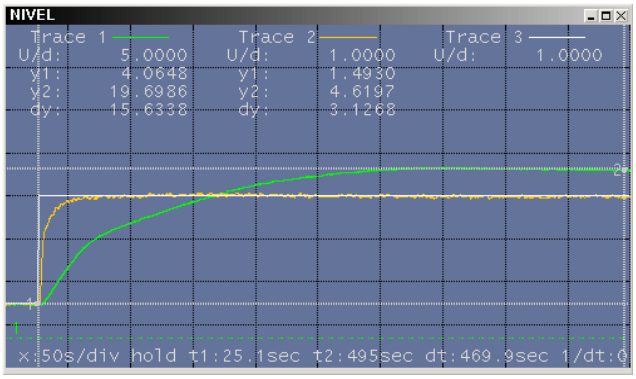
a) VC (traza naranja), SP (amarilla), duración disturbio (azul claro)



b) EC auto (blanco), duración disturbio (azul claro), EC manual (violeta)

Luego de comprobar el funcionamiento del lazo interno, se realiza la identificación del lazo primario empleando, para esto, el lazo interno en automático. La excitación, consistente en un escalón de 2.5 gpm, se aplica al lazo interno y por medio de este a la planta de nivel, estando la VC del sistema en un valor estable de 4 cms. La ilustración 52 ilustra el resultado, en esta el eje de tiempo es de 50 s por división, el eje de las ordenadas es de 1 gpm por división para el caudal y de 5 cms por división para el nivel. En la ilustración se observan 3 señales, la primera es el valor de consigna para el lazo interno SP (traza blanca) que también es la excitación de la planta, la VM (traza naranja) es el caudal aplicado a la planta por parte del lazo interno y la tercera es la VC (traza verde) o el nivel en el tanque. Aplicando el mismo procedimiento de identificación automático aplicado al lazo interno se identifican los parámetros dinámicos para la planta, los cuales se consignan en la tabla 7.

Ilustración 52. Identificación del Lazo Externo



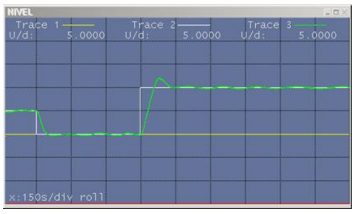
VC (traza verde), SP (amarillo), VM (roja)

Con base en los parámetros dinámicos del lazo externo, se sintoniza el control PID AWBT por medio del método *Minimun ITAE Kaya y Sheib* en modo servomecanismo, las constantes de sintonización se almacenan en la tabla 7. Luego de identificar los dos procesos, se procede a verificar la respuesta del control en cascada bajos dos situaciones: cambio en la consigna y rechazo a perturbación.

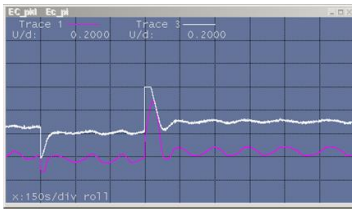
Prueba de cambio en el valor de consigna: En esta prueba estando el sistema en estado estable, VC de 20 cm, se procede a cambiar a una nueva consigna de 15 cm. Los resultados se observan en la ilustración 53. En esta se configura el eje de tiempo en 150 s por división, en la ilustración 53a el eje de las ordenadas es de 5 cms por división, en la 53b el eje está configurado en 0.2 unidades por división y en la 53c tiene el eje esta configurado en 1 gpm por división. La ilustración 53a muestra dos señales la VC (nivel: traza verde) y el valor de consigna SP del lazo externo (traza blanco), en esta ilustración mientras el sistema se estabiliza en un nivel de 20 cm se genera un cambio en el SP a 15 cm y luego un cambio de este a 25 cm, con la sintonización realizada al cascada, el sistema responde rápidamente aunque con una pequeña respuesta oscilatoria. En la ilustración 53b se despliegan los esfuerzos de control de los controladores, el EC del PID (lazo externo, traza blanca) y el EC del PI (lazo interno, traza violeta), el EC del PID representa el caudal ordenado por el lazo externo al interno y el EC del PI es la orden impartida al actuador por el lazo interno para cumplir los requerimientos del lazo externo, y finalmente en la ilustración 53c se observa la consigna aplicada al controlador PI del lazo interno (traza verde) y la VM o caudal entrante a la planta (traza naranja). Tanto en 53b y 53c las señales responden a un cambio en el valor de consigna en el nivel de 5 cms

y luego a uno de 10 cms, en todas las ilustraciones se observa que gracias a la sintonización se reacciona rápidamente ante el cambio en el valor de consigna, se ve claramente como el mayor esfuerzo de control es realizado por el controlador externo y en la ilustración de la VM (ilustración 53c) se observa como el controlador PI hace un buen seguimiento de la consigna incluso ante cambios rápidos

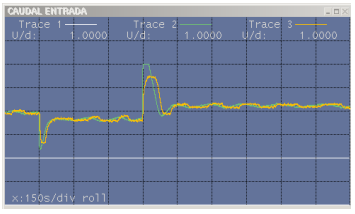
Ilustración 53. Respuesta Control en Cascada ante un cambio en el escalón



a) VC (traza verde) SP (blanca)



b) EC PID (amarillo), EC PI (rojo)

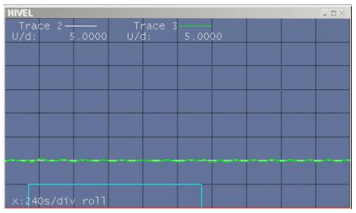


c) SP PI color blanca, VM color azul

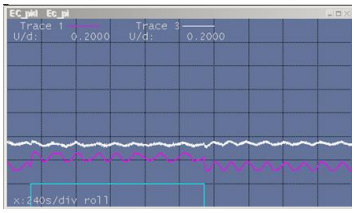
Prueba de rechazo a perturbaciones: Ahora se evalúa la capacidad de rechazo a perturbaciones del control en cascada, la ilustración 54 usa las mismas configuraciones de eje de las ordenadas de la prueba anterior, el eje de tiempo equivale a 240 s por división en cada una. En este experimento se aplica un fuerte disturbio en la VM, este consiste en una disminución en el caudal de 2 gpm, cuando el sistema en cascada se encuentra estable con VC = 10 cms y EC = 0.58. En la ilustración 52a se aprecia claramente que al aplicar el disturbio al sistema (traza azul claro), la VC no se desvía de su valor de consigna, esto se logra ya que el lazo interno detecta la variación en el caudal y modifica el EC del PI (traza violeta) reaccionando rápidamente ante el efecto perturbador del disturbio, se aprecia que el EC del PI corrige el problema evitando que el control PID realice algún gasto de energía (traza blanca) y de esta manera la VC del proceso se mantenga sin mayor cambio durante el disturbio (traza verde), de ahí que la perturbación en la VC sea prácticamente nulo. En el momento que desaparece el disturbio se generan un pico en el caudal de entrada (traza naranja), pero a pesar de ello la VC sigue sin modificaciones. De esta manera se comprueba la gran capacidad del control en cascada de minimizar el efecto perturbador en la VC de un proceso cuando un disturbio es aplicado a la VM.

La respuesta oscilatoria del sistema en cascada se puede suavizar realizando una sintonización final sobre los valores de los parámetros de los controladores entregados por el método de sintonización elegido.

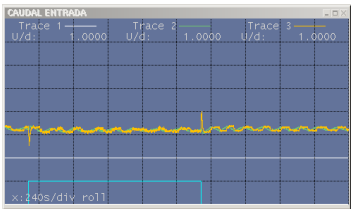
Ilustración 54. Respuesta Control en Cascada ante una perturbación



a) VC (traza verde), SP (blanca), duración disturbio (azul claro)



b)EC PID (blanca), EC PI (violeta), duración disturbio (azul claro)



c) SP PI (verde), VM (naranja), duración disturbio (azul claro)

Tabla 7. Parámetros de sintonización Control en Cascada

ESTRATEGIA	MODO	MÉTODO	K	TAO	L	KC	TI	TD
LAZO INTERNO CONTROL PI AWBT	SERVOMECANISMO	IAE ROVIRA ET AL.	1.4316	2.113	3.215	0.5177	3.11685	--
LAZO EXTERNO PID SERIE AWBT	SERVOMECANISMO	MINIUM ITAE KAYA Y SHEIB	9.849	88.18	6.150	1.53278	88.19546	2.57876

7. CONCLUSIONES

Se ha diseñado e implementado un sistema de control y supervisión de tiempo real bajo PC para un esquema en cascada aplicado a la planta de tanques interactuantes del Laboratorio de control de procesos del Programa de ingeniería en Automática Industrial. En este los estudiantes tendrán la posibilidad de interactuar con un proceso industrial real, a mediana escala, por medio de un sistema de tiempo real dotado de una interface hombre-máquina que implementa el supervisorio, a través del cual se podrá experimentar con: identificación de plantas, sintonización de controladores PID, *windup*, *bump transfer*, control realimentado, control en cascada y rechazo a perturbaciones.

Dentro del control clásico, realimentado y en cascada, se considera que el control en cascada es el primer escalón de mejora de un lazo simple de control. Es una mejora sencilla y en ocasiones económica, principalmente si ya existe la medición de la variable manipulada, y si se dispone como plataforma de control de un sistema fácilmente adaptable.

Se ha diseñado, implementado y validado un control PID industrial de tiempo real, de estructura serie y paralela, que implementa funcionalidades como: filtro de la derivada, filtro del valor de consigna y realimentación de la velocidad y que, a diferencia de los controladores académicos, incluye un mecanismo de *antiwindup* y *bumpless transfer* a través de la técnica condicional.

Con base en las técnicas AWBT investigadas, la técnica elegida a implementar fue el método de Técnica Condicional, este es uno de los métodos que más se utiliza dentro de la práctica y la simulación de sistemas. Al implementar la técnica en la componente integral del PID se disminuye el valor del sobre-pico, así como el tiempo de establecimiento de la salida del sistema, generados por el reajuste excesivo de la componente integral (*windup*). También dicha técnica disminuye en gran parte la presencia de saltos abruptos en la variable controlada por la conmutación en la señal de referencia entre sus modos manual y automático (*bump transfer*), puesto que hace que la

salida del control PID sea lo más cercana posible a la señal de control manual; adicionalmente permite un buen seguimiento de la consigna y disminución de la saturación del actuador ante cualquiera de los dos fenómenos.

RTAI-Lab representa una valiosa opción dentro de las herramientas de fuente abierta para ejecutar diseñar, sintonizar e implementar sistemas de control de tiempo real dentro de un ambiente Linux para las instituciones educativas de los países, no solo, del tercer mundo sino de los países desarrollados. Esta aplicación ha sido ampliamente utilizada junto con Scilab/Scicos por el SUPSI y por el DIAPM, importantes instituciones de la Unión Europea, y ahora por la Universidad del Cauca, pionera en Latinoamérica en el uso de esta herramienta, para realizar la aplicación de tiempo real de un control en cascada en una planta de tanques i0nteractuantes instrumentada. Dicha herramienta permite adicionalmente integrar sistemas de control electromecánicos con diferentes sistemas de control de diferente complejidad, desde controlador PID hasta sistemas difusos.

Se comprobó la eficiencia de algunos métodos empíricos de sintonización para controladores PID, ver anexo B, logrando resultados satisfactorios. Los diversos experimentos realizados empleando los métodos de sintonía incorporados el sistema supervisorio, permitieron obtener valores óptimos para las constantes que sintonizaban el controlador y, en muchas ocasiones, haciendo innecesario realizar sintonización fina sobre estas constantes, cuando se buscaba un mejor buen seguimiento de la consigna, rechazo a perturbaciones o respuesta más suave.

Se acondicionó la Guía de prácticas para la planta de tanques interactuantes del laboratorio de control de procesos, en esta se detallan los pasos a seguir para llevar a cabo los diversos experimentos con el nuevo esquema en Cascada con control PID AWBT, además explica los diversos pasos a seguir para ejecutar el control en el sistema de tiempo real RTAI, permitiéndole al estudiante una vía sencilla para la ejecución de diversas actividades como identificación del proceso, sintonización de las estrategias de control, entre otras.

Finalmente cabe mencionar que se han cumplido a cabalidad los objetivos planteados en al anteproyecto, no sólo por los resultados reportados en este documento, sino por haber

logrado la publicación de dos artículos: “Simulación y Control PI de una planta RC en Tiempo Real con RTAI-Lab”, aceptado y presentado en el evento nacional 4to IEEE Workshop Colombiano en Robótica y Automatización – CWRA 2008, y “Simulación y Control en Cascada de una planta POMTM en Tiempo Real con RTAI-Lab”, artículo aceptado en el evento internacional: XIII Congreso Latinoamericano de Control Automático, CLCA 2008 en Mérida – Venezuela.

8. RECOMENDACIONES Y TRABAJOS FUTUROS

8.1. RECOMENDACIONES

El proyecto que se describe en este documento presenta una solución que ha sido implementada con condiciones de tiempo real, con herramientas de software libre bajo un entorno Linux, es por ello que se sugiere que antes de interactuar con el sistema de realice un estudio previo de las herramientas utilizadas dentro de la aplicación, tales como *Scilab/Scicos*, *xrtailab* y una herramienta muy utilizada en entornos Linux como es el Terminal, que permite el uso de comandos muy semejantes a los de DOS, los cuales son necesarios para el uso del sistema propuesto, debido a que muchos de los usuarios solo están familiarizados con Windows.

El hecho de haber trabajado con la herramienta de procesamiento numérico de fuente abierta como *Scilab/Scicos* para generar las tareas de tiempo real, no descarta la posibilidad de implementar el mismo modelo de solución con base en *Matlab/Simulink*, siendo esta una herramienta comercial. El lector debe adoptar este documento como una ventana de posibilidades de adaptación y mejora, lo mismo que de búsqueda de nuevos ambientes de trabajo.

8.2. TRABAJOS FUTUROS

Por medio del sistema de tiempo real implementado se pueden diseñar e implementar diversas estrategias de control y gracias a las modificaciones realizadas al hardware de la planta de tanques interactuantes, se podrán medir las perturbaciones que se presentan en el proceso; es por ello que una de las estrategias de control que se puede construir es un control *Feedforward*, puesto que una de las condiciones para aplicar este tipo de estrategia de control a un proceso es que se puedan medir cuantitativamente las perturbación que entran en él.

RTAI permite usar una comunicación host–target de manera local o remota, entre la aplicación en espacio de usuario con la tarea en tiempo real. En este proyecto se ha utilizado una comunicación host-target de manera local, es por ello que entre los primeros desarrollos a futuro con el SOTR RTAI/RTAI-Lab se debe considerar la implementación de la comunicación remota. Para ello se cuenta con la herramienta RTAI-XML, que es un servidor que se comunica por medio de llamadas con el lugar en donde un proceso de tiempo real, (el Target) está funcionando. Un programa cliente genérico (el Host), puede alcanzar el servidor mediante la red TCP/IP, utilizando un protocolo estándar sobre la base de XML, y por lo tanto interactuar con el Target, de esta manera monitorear el estado del proceso en tiempo real, ver las señales generadas por el sistema y también cambiar parámetros. En simples palabras, RTAI-XML provee una simple vía de comunicación remota entre aplicaciones de control, adicionando flexibilidad al SOTR RTAI, sin perder las principales características de una implementación abierta.

9. BIBLIOGRAFÍA

ALFARO, Víctor. Ecuaciones para Controladores PID Universales. Artículo de divulgación, San José, Costa Rica, 2002a. p. 11-20.

ALFARO, Víctor. Identificación de procesos sobre-amortiguados utilizando técnicas de lazo abierto. Artículo de divulgación, San José, Costa Rica, 2001a. p. 13-18.

ALFARO, Víctor. Métodos de sintonización de controladores PID que operan como reguladores. Articulo de divulgación, Universidad de Costa Rica, San José, Costa Rica, 2002b, p. 21-36.

ALFARO, Víctor. Métodos de sintonización de controladores PID que operan como servomecanismos. Artículo de divulgación, Universidad de Costa Rica, San José, Costa Rica, 2003, p. 13-29.

ALFARO, Víctor. Estudio de los sistemas de Control PID. Articulo de Divulgación Universidad de Costa Rica, San José, Costa Rica, Enero 2008, p. 15-17.

ASTRÖM, K.J. and HÄGGLUND, T., Automatic Tuning of Simple regulators with specifications on Phase and Amplitude Margin, Automática, Vol. 20, no 5, 1984.

BARBALACE, A. et al. Performance Comparison of VxWorks, Linux, RTAI, and Xenomai in a Hard Real-Time Application. IEEE Transactions on Nuclear Science, Vol. 55, No. 1, February 2008. p 435-439.

BLICKLEY, G. Modern Control Started with Ziegler-Nichols Tuning. Control Engeeniring. E.U.A, Vol. 37, N°12 (2), October 1990.

BOHN, Christian and ATHERTON, Derek. An Analysis Package Comparing PID Anti-Windup Strategies. IEEE Control Systems. Vol 15. Issue 2. April 1999, p. 35-36.

BUCHER Roberto, MANNORI Simone and NETTER Thomas. RTAI-Lab tutorial: Scilab, Comedi, and real-time control. Scuola Universitaria Professionale della Svizzera Italiana (SUPSI), February 2008. p 1-47.

BUCHER Roberto, DOZIO, Lorenzo. and MANTEGAZZA Paolo. Rapid Control Prototyping with Scilab/Scicos and Linux RTAI. Article of divulgation, and Dipartimento di Ingegneria Aerospaziale del Politecnico di Milano (DIAPM), Milano, Italy, 2004. p 1-9.

BUNKS C, DELEBECQUE F, LE VEY G, and STEER S. Signal Processing With Scilab, Scilab Group, INRIA - Unitè de Recherche de Rocquencourt - Projet Meta2, Le Chesnay Cedex France, November 1998, p -1-205.

BURNS, Alan y WELLINGS, Andy. Sistemas de Tiempo Real y Lenguajes de Programación. Editorial Addison Wesley. Tercera Edición, Madrid – España. 2003. p 1-2.

CHEN, Cheng-Lian Enhanced PID Control: Cascade Control. PSE Laboratory Department of Chemical Engineering National TAIWAN University, Taiwan. May 2006, p. 1-39.

COOPER, Doug. An Implementation Recipe for Cascade Control [on line]. [Citado 2 Julio, 2008]. Disponible en internet: <URL:http://www.controlguru.com/2007/090207.html>.

DÍAZ, Javier. Sistemas Operativos de Tiempo Real. Artículo de divulgación, p 10-14, 2005.

DEPARTMENT OF CHEMICAL ENGINEERING UNIVERSITY OF THE NEGEV. Cascade Control. Article of divulgation, Beersheba, Israel, p. 245-262.

FLORES, Antonio. Diseño de Estructuras de Control Complejas. Articulo de divulgación, Departamento de Ciencias, Universidad Iberoamericana, México DF, México, Octubre 2000a. p. 1-8.

FLORES, Antonio. Control en Cascada. Articulo de divulgación, Departamento de Ciencias, Universidad Iberoamericana, México DF, México. Noviembre 2002b. p. 1-16.

FLÓREZ, Juan and DIAZ, Jaime. Industrial Series AWBT PID Controller in RTlinux. Article of Divulgation. University of Cauca, Popayán, Colombia. October 2007. p. 2-4.

FLÓREZ, Juan, CABEZAS Yonny y DÍAZ, Ermilso. Simulación y Control PI de una planta RC en Tiempo Real con RTAI-Lab. 4to IEEE Workshop Colombiano en Robótica y Automatización – CWRA 2008, Cali Colombia, Agosto 13-14 de 2008 (a). p 1-6.

FLÓREZ, Juan, CABEZAS Yonny y DÍAZ. Simulación y Control en Cascada de una planta POMTM en Tiempo Real con RTAI-Lab. Artículo de divulgación. Propuesto para el XIII Congreso Latinoamericano de Control Automático, CLCA 2008, Mérida - Venezuela, Agosto 2008 (2008b). p 1-6.

GUEVARA, Pedro; LOPEZ, Asdrúbal y MEDEL, José de Jesús. Importancia de los Sistemas de Tiempo Real. Artículo de divulgación. Centro de Investigación en Computación. México D.F, p 1-5, 2007.

HEONG, K.; Chang, G.; Li Y. "PID Control System Analysis, Design and Technology", IEEE Transactions on Control Systems Analysis Technology, Vol. 13, N°4, Julio 2005.

LAMANNA, R. Esquemas Generales de Estructuras de los Sistemas de Control Retroalimentados. Enero 2005. p. 1-30.

LÓPEZ ZAMARRÓN, Diego. Arquitectura de un Sistema Operativo de Tiempo Real. [Online]. Disponible en internet en http://www.ciclope.info/doc/rtos/rtos.php. 2004. Ultima visita Marzo 10 de 2008.

MAGALLÓN, Juan Antonio. Programación Paralela. Articulo de divulgación, Universidad de Zaragoza, Zaragoza-España, 2006. p 5.

MANNORI, Simmone, NIKOUKHAH, R. and STEER, S. Free and Open Source Software for Industrial Process Control Systems. INRIA-Rocquencourt, Domaine de Voluceau,, Le Chesnay Cedex, France, 2006. p 1-8.

MATTEI, Emanuele and LUDICIANI, Andrea. Real-time performance test comparison of VRTXsa, RTAI, and RTLinux. Articule of divulgation, Rome's University "La Sapienza". p 1-43. Disponible en internet en http://www.linuxdevices.com/news/NS5236356423.html

MAVAINSA. Control de Procesos, articulo de divulgación, Abril 2006, p 1-11. Disponible en Internet en http://www.mavainsa.com/documentos/9_control_procesos.pdf.

MIRANDA GÓMEZ, Oscar. Kernel de Tiempo Real para el Control de Procesos. Tesis de Maestría. Centro de Investigación y de Estudios Avanzados, IPN, México D.F. 2004. p 35-37.

NATIONAL INSTRUMENTS. DAC PCI-6023E/6024E/6025E User Manual. October 1998, p 1-138. Disponible en internet en http://www.ni.com/pdf/manuals/322072a.pdf

O'DWYER, Aiden. PI and PID controller tuning rules for time delay processes: a summary. Part 1: PI controller tuning rules, Proceedings of the Irish Signals and Systems Conference, National University of Ireland, Galway, June 1999, p. 1-8.

O'DWYER, Aiden. Part 2: PID controller tuning rules, Proceedings of the Irish Signals and Systems Conference, National University of Ireland, Galway, June 1999, p. 1-9.

PENG, Youbin, VRANCIC, Damir and HANUS, Raymond. Anti-Windup, Bumpless, and conditioned Transfer Techniques for PID Controllers. IEEE Control Systems. Vol 16, Issue 4, August 1996, p 48-57.

PROCTOR, Frederick. Introduction to Linux for Real-Time Control. Artículo de divulgación. National Institute of Standards and Technology - NIST, 2002. p 36–37.

RIPOLL Ismael. RTLinux versus RTAI. [Online] 2002. Disponible en internet en: http://www.linuxdevices.com/files/misc/ripoll-rtl-v-rtai.html.

ROJAS, Luis Felipe. Controladores PID Universales, Proyecto Eléctrico, Universidad de Costa Rica, Rodrigo Facio - Costa Rica, Noviembre 2007, p 14-18.

SAUCEDO F, Salvador. SIMULACION y DISEÑO DE SISTEMAS DE CONTROL EN CASCADA. Articulo de divulgación. Agosto 2001, p 2.

SCHLEEF D, HESS F and BRUYNINCKX H. Comedi: The Control and Measurement Device Interface handbook. Article of divulgation, July 2007. p 1-187

SECCHI, C., FANTUZZI, C and GIANOTTI, A. Control of an Industrial Manipulator by RTAI Linux. Article of divulgation, DISMI - University of Modena and Reggio Emilia, Reggio Emilia-Italy, 2000. p 1-5.

SHAW, John A. Modos del control en cascada [Online]. Disponible en internet en, http://www.learncontrol.com/pid/cascade2.html. Mayo 2006. Ultima consulta Julio 2008.

SILBERSCHATZ, Abraham y BAER Galvin, Peter. Sistemas Operativos. Editorial Addison Wesley Longman, 5ta Edición, México D.F, 1999. p 89-93.

SMITH, Carlos y CORRIPIO, Armando. Control Automático de Procesos. Editorial Limusa, Primera Edición, 1991. p. 21-30.

STROTHMAN, J. More than a century of measuring and controlling industrial process. InTech, E.U.A, June 1995.

VEIGA, Alejandro Luis. Sistemas Jerárquicos de Tiempo Real Para Adquisición de Datos y Control. Tesis de Maestría, Universidad de la Plata, La Plata – Argentina. 1999. p 28-29.

WALL, Kurt. Programación en Linux con Ejemplos, Editorial Prentice Hall, Primera Edición, Buenos Aires – Argentina. 2000. p 321-374.

Yun Li et al. PID Control Systems Analysis and Design. IEEE Control Systems Magazine. Feb 2006, p. 32-41.

ZIEGLER, J.G and NICHOLS, N.B. Optimum settings for automatic controllers. *Trans. ASME*. Vol. 64, no 8, 1942. p. 759 - 768.

ZULUAGA, Francisco Javier. Kernel de Tiempo Real Basado en Linux para una PDA. Tesis de Maestría. Centro de Investigación y de Estudios Avanzados, IPN, México D.F, 2005. p 16,19