

Simulación y Control de Procesos con Programación Gráfica

Por Ing. Igor Alvarado
BS ME, Kansas State University
igor.alvarado@ni.com y nilatam@usa.net
www.ni.com/latam

Director de Ventas
National Instruments - Andina y Caribe

1. Introducción

El modelaje y simulación se ha constituido en una poderosa herramienta para el diseño, análisis y optimización sistemas y procesos industriales. La disponibilidad de computadores personales cada vez más poderosos, de menor costo y de fácil uso, acompañados por software ó programas de aplicación y lenguajes de programación altamente flexibles, ha permitido la masificación del uso de diferentes técnicas de simulación y control de procesos. Este trabajo se concentra en la aplicación de las técnicas de modelaje, simulación y control de procesos haciendo uso de la tecnología PC y un lenguaje de programación 100% gráfico que permite la construcción de modelos mediante un paradigma de programación con “íconos interconectados por cables gráficos”, cuyo código es ejecutado en base al principio de “flujo de datos” por los “cables” que interconectan los diferentes “íconos” o funciones. Combinando este lenguaje de programación 100% gráfico con tarjetas de adquisición y control (entradas/salidas) para el PC, se puede entonces configurar un sistema de simulación y control capaz de manejar señales reales (analógicas y digitales, de entrada y de salida) y ejecutar modelos matemáticos de control de procesos en tiempo-real.

La simulación de sistemas se ha convertido entonces en una poderosa herramienta para la toma de decisiones que nos permite, entre otras cosas:

- Predecir el resultado de las acciones que se tomen sobre el proceso o sistema de control.
- Comprender porqué los eventos ocurren.
- Identificar áreas problemáticas antes de la implantación del sistema.
- Explorar los efectos de las modificaciones.
- Evaluar ideas y su viabilidad, e identificar sus ineficiencias.
- Estimular el pensamiento creativo y entrenar al personal.
- Optimizar los procesos (ahorros de energía, cuellos de botella, mejoras de los rendimientos, etc.).

Lo más temprano que se utilice la simulación de sistemas en un proyecto de automatización u optimización de procesos, mayor es el potencial de ahorro y efectividad que se logrará con dicho sistema. Esto se debe a que según estudios realizados, se ha podido confirmar que en un proyecto de automatización y control, más del 80% de los costos del sistema de control ya están comprometidos en la fase de diseño (ver Figura 1.1). Es decir, si se logra optimizar el sistema de control en su etapa de diseño, la reducción en los costos asociados con el mismo y su efectividad de maximiza.

Figura 1.1



Un mejor diseño se traduce entonces en un mejor producto, menores costos, mayor control, mayor seguridad, menor variabilidad del proceso, etc. Adicionalmente, a través de la simulación de sistemas se minimiza el ensayo y error, ya que se trabaja con modelos matemáticos de comportamiento predecible y controlable por los diferentes algoritmos de control que se apliquen al proceso real o simulado.

Para poder simular un sistema de control, se debe contar con un “modelo” matemático de la planta o proceso que se quiere controlar, el cual puede optimizarse vía interacciones múltiples, bajo múltiples escenarios, variables, parámetros que se evalúan individual ó simultáneamente.

En un modelo matemático, las entidades del sistema y sus atributos se representan mediante variables matemáticas. Las actividades se describen mediante funciones matemáticas que interrelacionan las variables. Los modelos matemáticos dinámicos normalmente se resuelven mediante métodos numéricos, y la simulación es una de las herramientas mas comunes.

Un modelo no es solo el sustituto de un sistema, sino también una simplificación del mismo. Para obtener un modelo se debe al menos:

- Determinar su estructura (entidades, atributos, actividades y fronteras) .
- Definir los datos (valores de los atributos y relaciones involucradas en las actividades).

Existe una estrecha interrelación entre el proceso de recolección de datos para el modelo, y el análisis de los mismos: las suposiciones relativas al sistema orientan la recolección de datos y el análisis de éstos confirma o refuta las suposiciones.

Los pasos básicos para modelar un sistema de control son:

- Definir el modelo matemático y programar el sistema de control.
- Seleccionar los parámetros y los valores iniciales.
- Ejecutar el modelo.
- Procesar los resultados (incluye la visualización y la investigación adicional).
- Interactuar con el modelo hasta lograr optimización.

Con el modelaje y simulación de los sistemas de control, se busca satisfacer la necesidad de controlar la fabricación de los productos mediante el mantenimiento de las variables de proceso en los valores más estables posibles. Los controladores comparan las variables de proceso con el punto de consigna (setpoint) y a partir de su diferencia o error calculan cual debe ser la señal de salida al elemento final de control.

Para controlar el proceso, se utiliza el controlador más apropiado:

- Relé (On-Off)

- Ratio
- PID,
- Fuzzy Logic,
- Otros (Cascada, Adaptivo, Algoritmos Genéticos, Redes Neuronales, etc.) .

Los procesos industriales son simulados de dos (2) formas: utilizando la teoría clásica de control o mediante el concepto de espacio de estado.

La arquitectura PC y la programación gráfica nos permiten realizar pruebas en tiempo-real con señales simuladas, reales (“hardware in the loop” testing) ó híbridas (simuladas ó reales).

El lenguaje de programación gráfica utilizado en este trabajo, incluye herramientas para el modelaje, análisis y simulación de sistemas dinámicos, y soporta sistemas lineales y no-lineales, discretos y continuos. También permite la combinación de estos dos (2) modelos basados en el tiempo, y permite que el sistema sea parcialmente simulado y/o parcialmente real (con señales reales).

Se demostrará en forma práctica el uso de un lenguaje gráfico y sus aplicaciones en el modelaje, simulación y control de procesos industriales.

2. La simulación de sistemas de control en la empresa:

Existen cuatro (4) áreas básicas en toda empresa manufacturera que se pueden beneficiar del modelaje y simulación de sistemas (ver Figura 2.1):

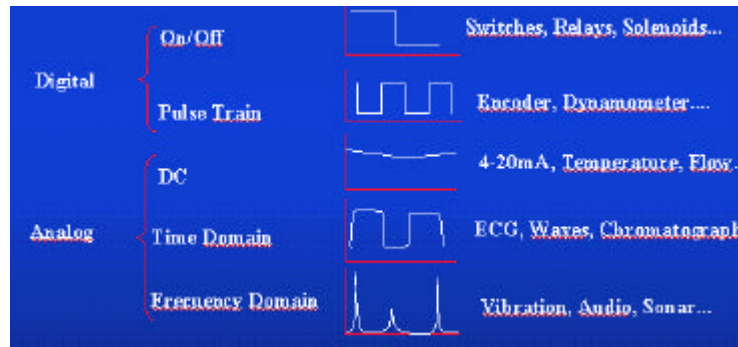
- Diseño (Ingeniería)
- Operaciones (Producción)
- Control de Calidad (Pruebas y Medidas)
- Mantenimiento

Figura 2.1



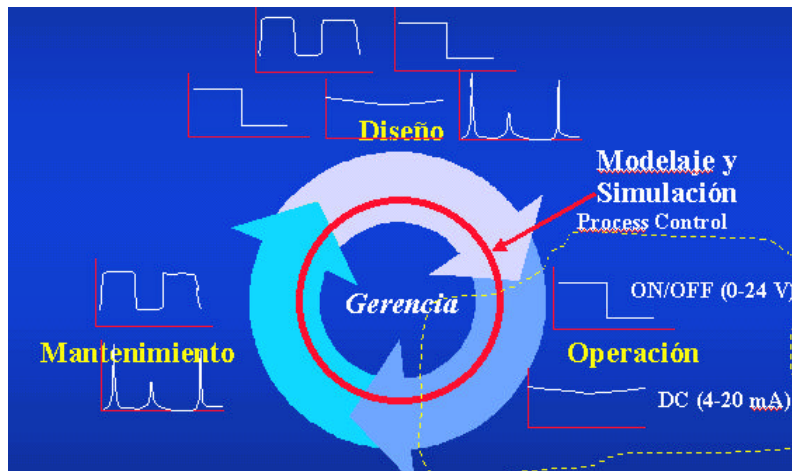
Para que un lenguaje de programación pueda ser utilizado exitosamente en aplicaciones de modelaje y simulación de sistemas de control, debe contar con una gran variedad de funciones matemáticas, gráficas y de procesamiento de señales. Tal lenguaje, debe ser fácil de aprender y usar, y debe permitir tanto el manejo de señales reales de tipo analógico y digital, como señales simuladas (ver Figura 2.2).

Figura 2.2



A estos tipos de señales eléctricas, debemos agregar las señales de video (analógico y digital) que generan las cámaras utilizadas en los sistemas de visión de máquinas y captura de imágenes en tiempo-real, y las señales de control de movimiento (analógica y digital) utilizadas para controlar motores de paso y servo. De esa forma, el lenguaje de simulación puede ser utilizado en cualquiera de las cuatro (4) áreas críticas antes mencionadas: ingeniería, operaciones, control de calidad y mantenimiento (ver Figura 2.3).

Figura 2.3



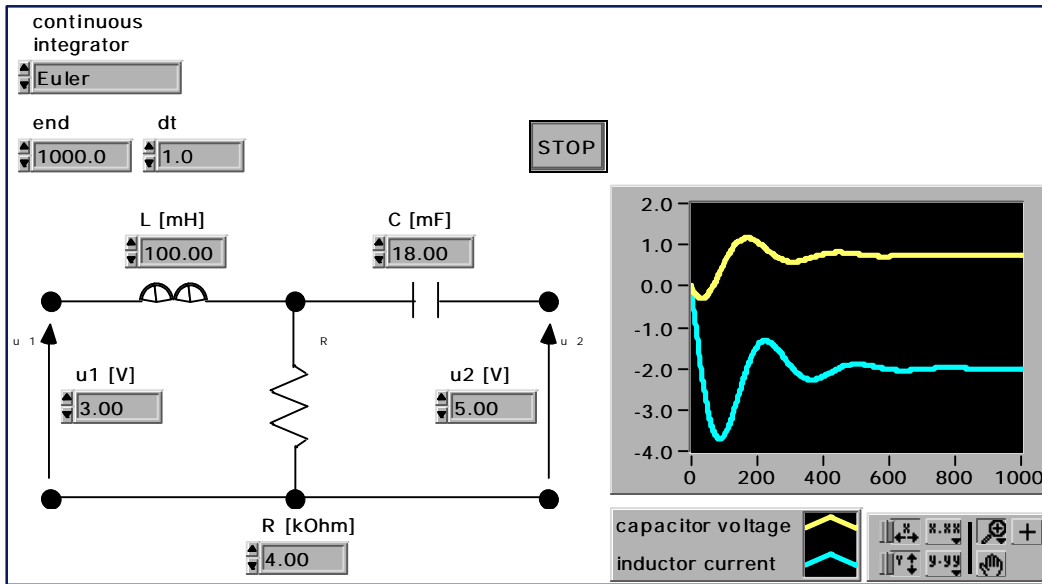
Los sistemas de simulación tradicionales, utilizan principalmente señales simuladas a través de funciones o datos pre-capturados, mientras que los lenguajes de programación permiten el uso de señales reales analógicas y digitales, junto a las simuladas.

El manejo de señales reales con los modelos de simulación, permiten que el modelo sea optimizado y ajustado en tiempo-real, de acuerdo a los patrones de comportamiento de las señales de entrada y salida del sistema. Es decir, el modelo puede ser ajustado en base a la respuesta del proceso o sistema real a las señales de estímulo generadas por el modelo mismo.

Esto hace que la transición del modelo/simulación hacia el sistema final sea rápida y efectiva, disminuyéndose el número de interacciones normalmente requeridas para adaptar o implantar el modelo en el proceso real.

Por ejemplo, se podría modelar y simular un circuito RLC completamente en software, y luego construir físicamente el circuito RLC y aplicarle el modelo de control previamente simulado (ver Figura 2.4).

Figura 2.4

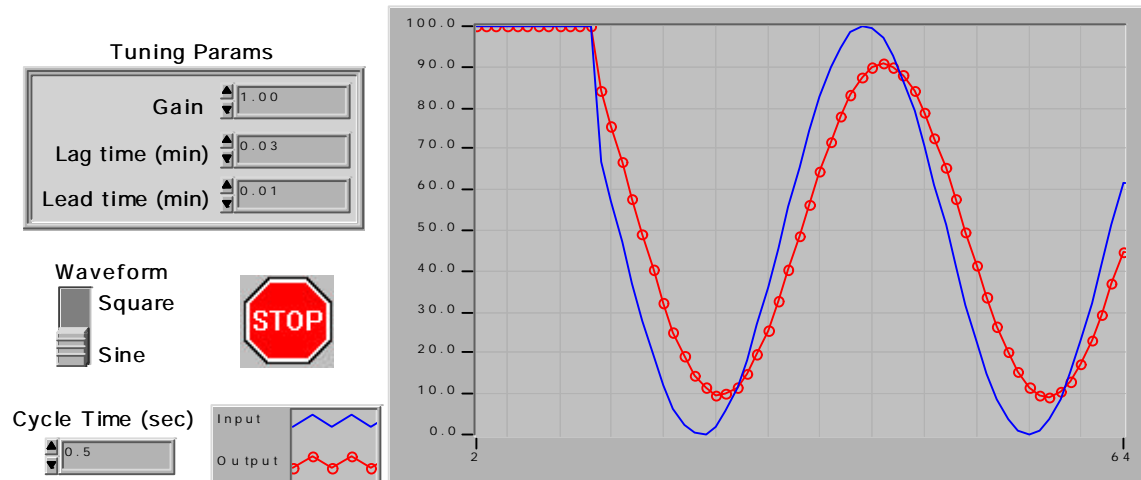


Bajo este esquema, la transición del modelo simulado al modelo físico, es casi transparente, y solo requiere de la sustitución de la “planta” o proceso simulado, por uno real. El sistema de control sería el mismo para ambos casos, y puede ser validado inmediatamente tanto con el modelo físico como con el proceso real.

Igualmente, si queremos aplicarle un controlador PID o un Lead-Lag a un proceso simulado (ver Figura 2.5), podemos primero aplicárselo al modelo simulado (función de transferencia, ecuación diferencial, ecuaciones de espacio de estado, etc.), y posteriormente al proceso real.

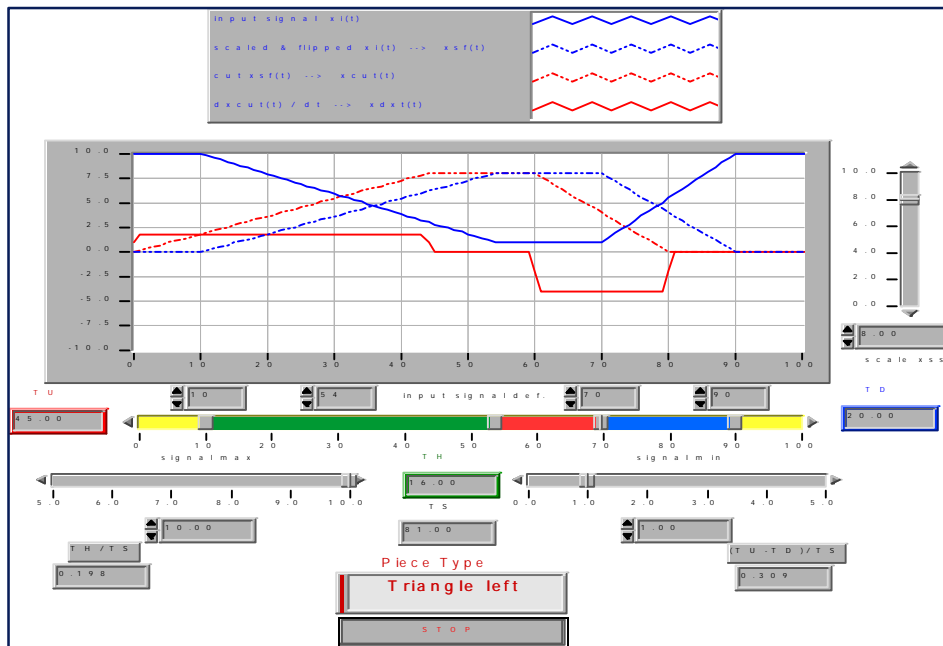
Figura 2.5

Demonstration of the function of the Lead/Lag block. Vary tuning parameters and observe response.



Si se demuestra que el controlador PID no es el adecuado, o si simplemente se desea evaluar otros controladores sobre el mismo sistema o proceso, se puede entonces aplicar un controlador de lógica difusa (ver Figura 2.6), u otros algoritmos de control (genéticos, redes neuronales, control adaptivo, etc.)

Figura 2.6



3. Programación gráfica:

Existen muchos lenguajes de programación en el mercado. Algunos de ellos utilizan la denominación “visual” en su nombre (Ej. Visual C, Visual Basic, etc.). Lo cierto es que la mayoría de estos lenguajes son no más del 50% visuales, ya que están basados en una combinación de objetos gráficos y texto. Sin embargo, si existen lenguajes de programación 100% gráficos, es decir, lenguajes que permiten programar y desarrollar una aplicación sin utilizar texto. En este trabajo se utilizó un lenguaje de programación que permite la programación sin texto, utilizando sólo íconos, cables gráficos y controles ó indicadores (objetos gráficos).

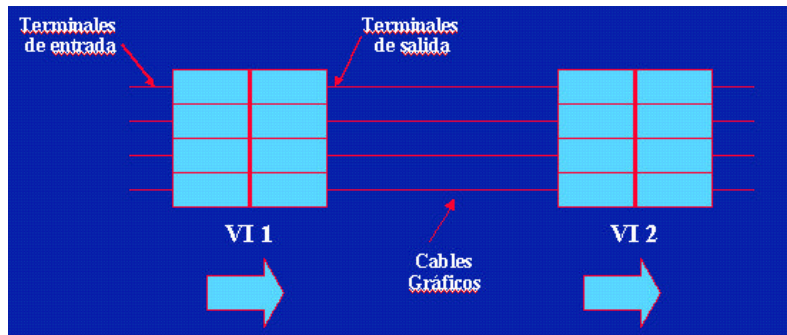
El principio o paradigma de programación utilizado por este lenguaje gráfico se basa en cuatro (4) elementos básicos:

- El uso de íconos, cables gráficos y controles-indicadores.
- Programación en base a un diagrama de bloques
- El uso de dos (2) ventanas de programación: panel de controles y panel del diagrama
- Ejecución del programa en base al flujo de datos, en forma paralela

La programación con diagramas de bloques se basa en íconos o bloques gráficos que se interconectan entre sí a través de “cables gráficos” (ver Figura 3.1). Los datos “fluyen” entre los bloques o funciones a través de los “cables gráficos”. Cada bloque o función cuenta con terminales de entrada en el lado izquierdo, y terminales de salida, en el lado derecho.

Cada bloque se ejecuta cuando todos los datos de entrada son recibidos en los terminales de entrada, permitiendo la ejecución paralela de más de un bloque en un mismo programa. Cada programa se denomina “Instrumento Virtual” ó VI (según sus siglas en el idioma inglés), y puede incluir uno ó mas sub-Vis como una especie de sub-rutina.

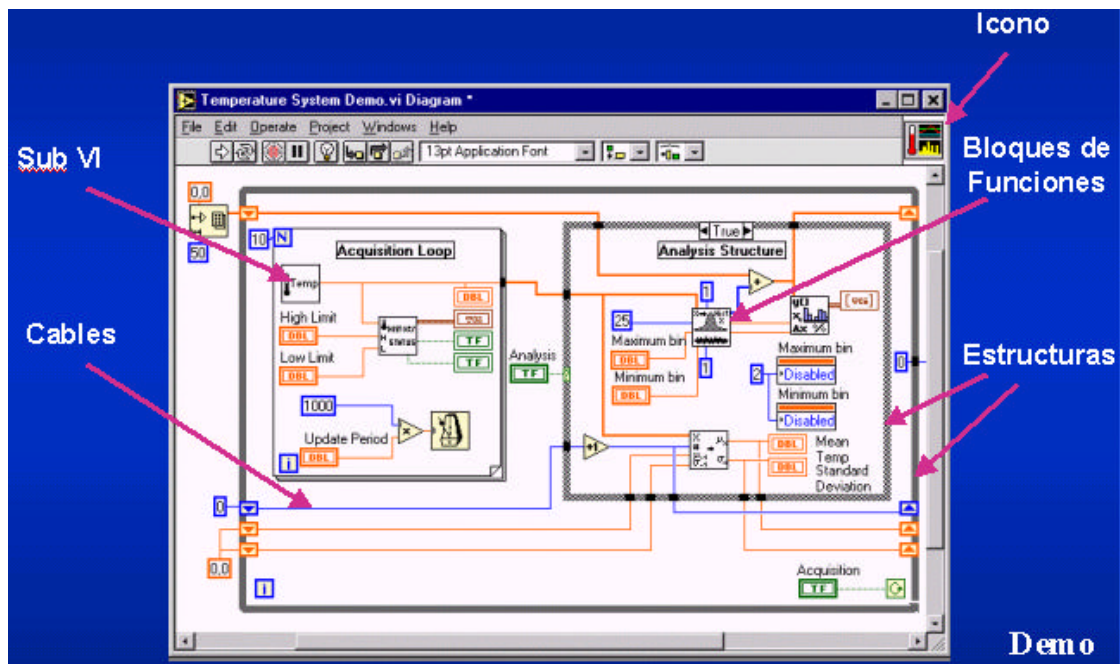
Figura 3.1



De esta forma, se puede desarrollar una aplicación completa (ver Figura 3.2), basada en un lenguaje de programación altamente intuitivo, fácil de aprender, pero a la vez, tan poderoso y rápido como cualquier otro lenguaje.

A cada tipo de dato (entero, punto-flotante, cadena de caracteres, etc.) se le asigna un color específico, lo que permite su rápida identificación en el código fuente.

Figura 3.2



El lenguaje de programación gráfica utilizado en este trabajo cuenta con un compilador de alta eficiencia, que permite crear ejecutables (EXE) para la posterior distribución del modelo.

4. Simulación de sistemas-Conceptos básicos:

Un “sistema” es un conjunto de objetos reunidos en alguna interacción o interdependencia regular. Los “sistemas” están compuestos por entidades, atributos y actividades. El “estado” de un sistema describe la condición actual de sus entidades, atributos y actividades. Su progreso se estudia siguiendo los cambios del “estado” del sistema.

Un “modelo” es una descripción lógica de cómo un sistema, proceso ó componente se comporta. La simulación incluye el diseño de un “modelo” de un sistema, proceso o componente, y la realización de experimentos sobre el mismo. El propósito de los experimentos es determinar cómo el sistema real se desempeña, y pronosticar el efecto de los cambios sobre el mismo en el tiempo.

Se utiliza la simulación para responder tales preguntas como:

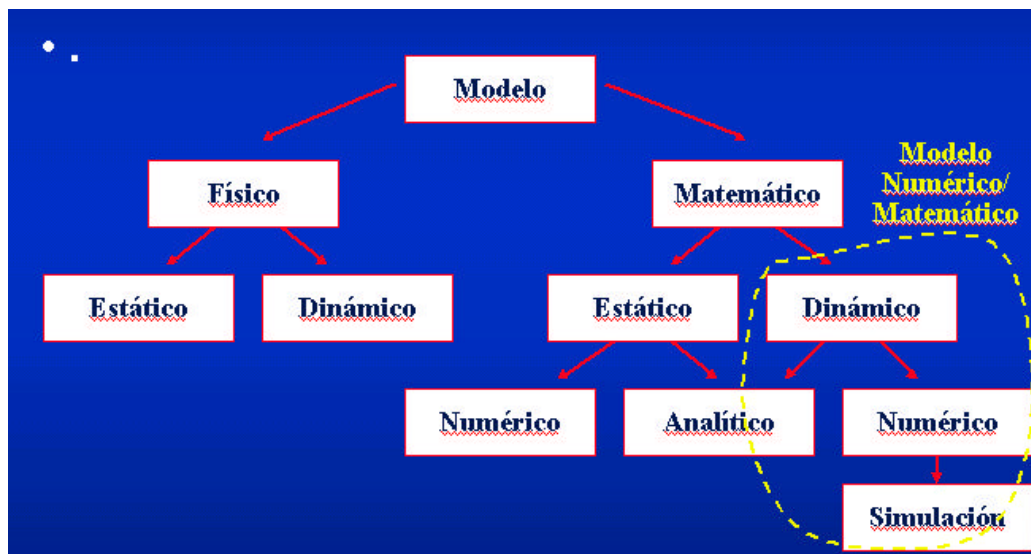
- Resultará este cambio en el proceso en una mayor calidad, eficiencia ó producción?
- Podemos estabilizar el sistema en un menor tiempo aplicando este algoritmo de control?
- Se reduce el consumo de materia prima realizando estos cambios al sistema de control?
- Se reduce la variación del proceso utilizando este nuevo algoritmo?
- Etc.

Existen actividades endógenas que sólo ocurren dentro del sistema. También existen actividades exógenas que sólo ocurren fuera del sistema (en el entorno). Si el sistema es un sistema “cerrado”, entonces se dice que no cuenta con actividad exógena. Si el sistema es “abierto”, entonces si cuenta con actividad exógena. Los cambios en actividades exógenas (del entorno) pueden afectar el sistema.

Si el resultado de una actividad se puede describir completamente en términos de su entrada, se denomina “determinística”. Si los efectos de la actividad varían aleatoriamente en diferentes salidas, se denomina “estocástica”. Si la ocurrencia de una actividad es aleatoria, se le atribuye al entorno o medio ambiente y por lo tanto es exógena. Se utilizan distribuciones de probabilidad para describir el carácter aleatorio de una actividad.

Los modelos son estáticos o dinámicos. Los modelos dinámicos se sub-dividen en continuos y discretos (ver Figura 4.1).

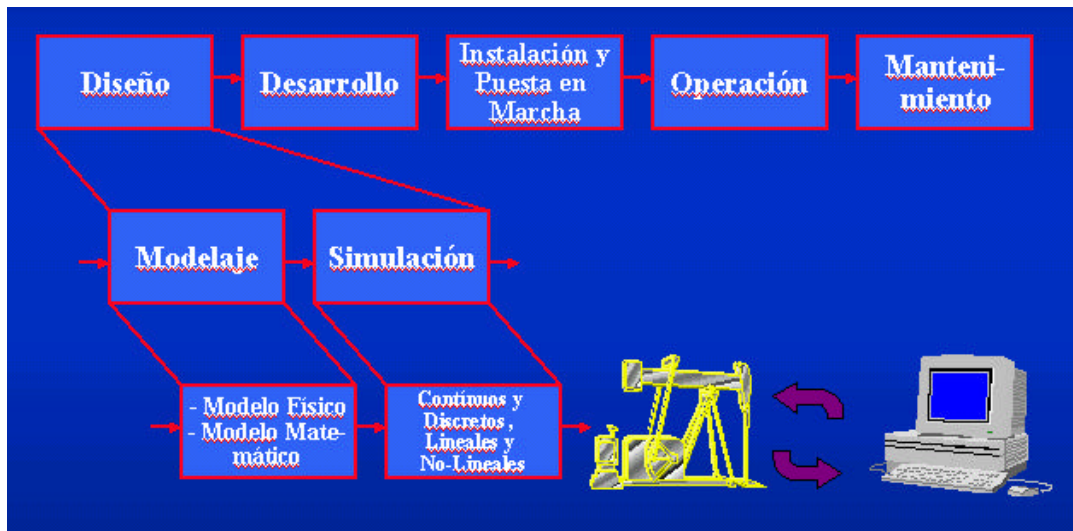
Figura 4.1



Los sistemas discretos son aquellos en los que los cambios ocurren en forma discontinua. Se describe en base a los eventos que producen los cambios en el estado del sistema, y a veces se simplifican considerando que los cambios ocurren en forma continua.

Hay pocos sistemas totalmente discontinuos o totalmente discretos (ver Figura 4.2). Sin embargo, siempre predomina un tipo de cambio (suave ó discontinuo), por lo que se pueden definir como continuos o discretos.

Figura 4.2



En un modelo matemático, las entidades del sistema y sus atributos se representan mediante variables matemáticas. Las actividades se describen mediante funciones matemáticas que interrelacionan las variables. Los modelos matemáticos dinámicos normalmente se resuelven mediante métodos numéricos, y la simulación es una de las herramientas más comunes.

Un modelo no es solo el sustituto de un sistema, sino también una simplificación del mismo. Para obtener un modelo se debe al menos:

- Determinar su estructura (entidades, atributos, actividades y fronteras) .
- Definir los datos (valores de los atributos y relaciones involucradas en las actividades).
- Definir interrelación: las suposiciones relativas al sistema orientan la recolección de datos y el análisis de éstos confirma o refuta las suposiciones.

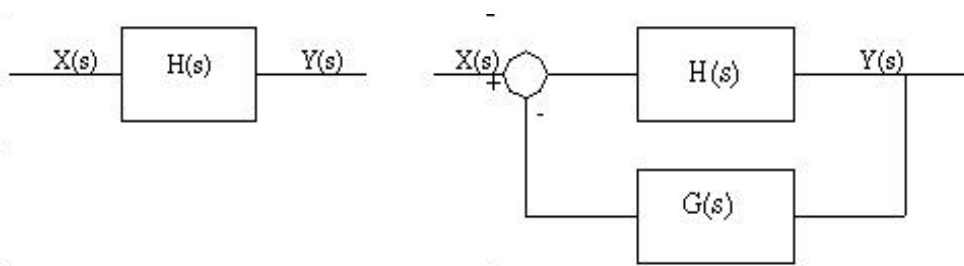
Los pasos básicos para modelar un sistema de control son:

- Definir el modelo matemático y programar el sistema de control.
- Seleccionar los parámetros y los valores iniciales.
- Ejecutar el modelo (simulación).
- Procesar los resultados (incluye la visualización y la investigación adicional).
- Interactuar con el modelo hasta lograr optimización.

Con el diseño y análisis de los sistemas de control, su modelaje y simulación, se busca satisfacer la necesidad de controlar la fabricación de los productos mediante el mantenimiento de las variables de proceso en los valores más estables posibles.

Los controladores comparan las variables de proceso con el punto de consigna (setpoint) y a partir de su diferencia o error calculan cual debe ser la señal de salida al elemento final de control (ver Figura 4.3). Se utiliza el controlador más apropiado: Relé (On-Off), Ratio, PID, Fuzzy y otros . Los procesos industriales son simulados de dos (2) formas: utilizando la teoría clásica de control o mediante el concepto de espacio de estado. Se realizan pruebas en tiempo-real con señales simuladas, reales (“hardware in the loop” testing) ó híbridas.

Figura 4.3



5. Programación gráfica para la simulación y control de procesos

Para ayudar en el diseño de los sistemas de control, el lenguaje que hemos gráfico utilizado para la simulación, permite realizar la transformación entre las diferentes representaciones matemáticas del sistema:

- Ecuaciones de espacio-estado, a función de transferencia
- Función de transferencia, a ecuaciones de espacio-estado.
- Cálculo de polos y ceros a partir de la función de transferencia
- Cálculo de parámetros de interés tales como: índice de amortiguamiento, frecuencias naturales, ganancia DC, polos y ceros para la determinación de la estabilidad del sistema.
- Etc.

El lenguaje de programación utilizado cuenta con una librería de simulación y control (Control and Simulation Toolkit, GSIM), le permite utilizar el diagrama de polos y ceros, y moverlos en forma gráfica para lograr la estabilidad del sistema, y calcular la ganancia de retroalimentación $G(s)$ para el nuevo sistema de lazo cerrador (ver Figura 4.3).

Una vez que se diseña el sistema de control, el lenguaje de programación gráfica le permite analizarlo en forma totalmente simulada o con señales reales capturadas o generadas en tiempo-real, a través de diagramas de Bode (ver Figura 5.1), Nyquist (ver Figura 5.2), y de Ubicación de Raíces (Root Locus, ver Figura 5.3) para el análisis del sistema de control y su respuesta en el dominio de frecuencia. El VI (Instrumento Virtual) de Ubicación de Raíces (Root Locus) también genera un archivo texto con información adicional tal como los ángulos de las asíntotas.

Figura 5.1

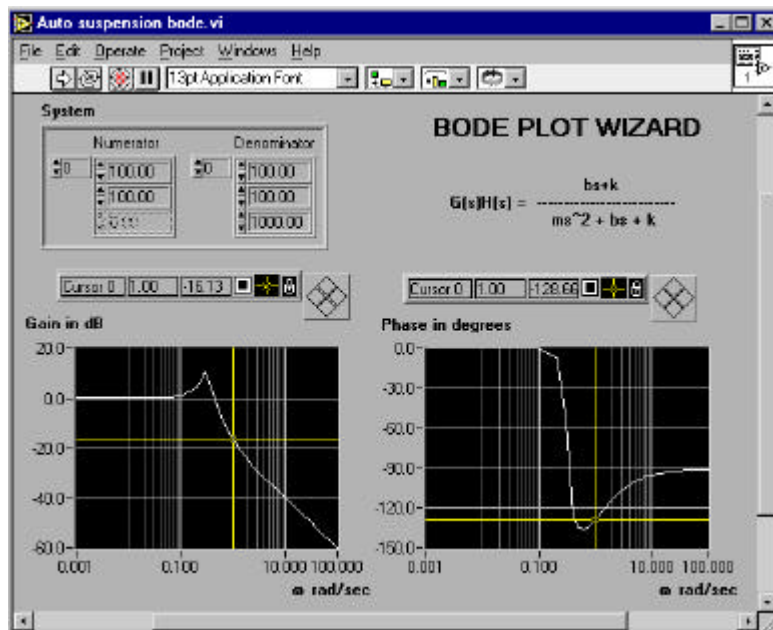


Figura 5.2

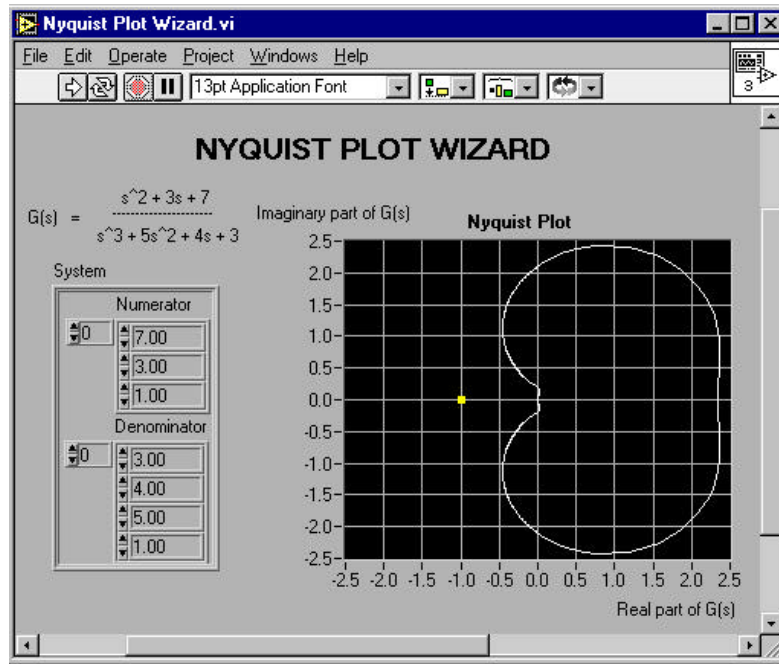
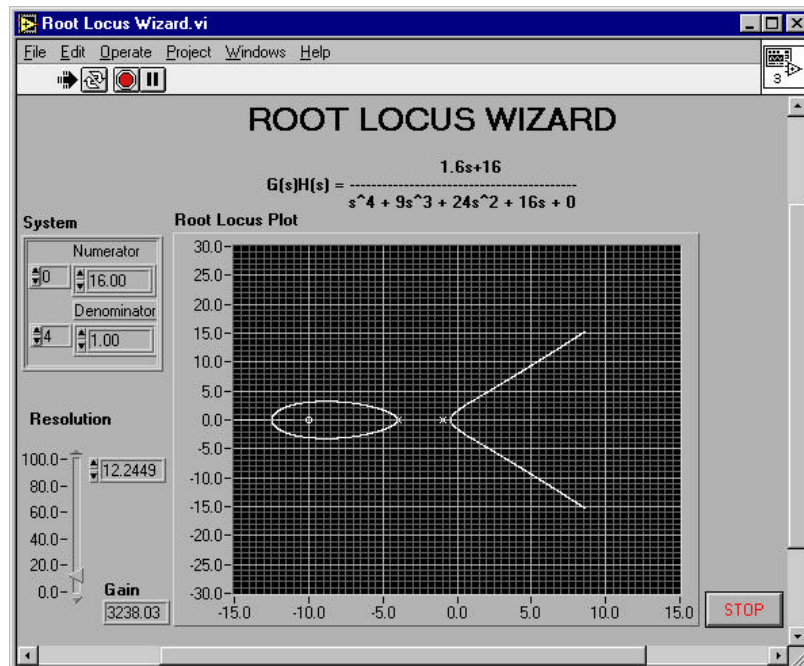


Figura 5.3



El lenguaje gráfico utilizado también cuenta con herramientas para el modelaje, análisis y simulación de sistemas dinámicos.

Este lenguaje soporta sistemas lineales y no-lineales, discretos y continuos, y permite la combinación de estos dos (2) modelos basados en el tiempo, y permite que el sistema sea parcialmente simulado y/o parcialmente real (con señales reales).

Al utilizarse junto al toolkit GSIM, tiene entonces las siguientes características:

- Simulación y control en tiempo-real
- Conexión directa a las tarjetas DAQ
- Despliega los resultados en forma gráfica
- Soporta rutinas pre-definidas (Ej. PID, Fuzzy, etc..)
- Ejecución de acciones mientras corre el VI
- Maneja ambos, problemas discretos y continuos
- Totalmente “abierto” (modificable)
- Ejecución y compilación rápida

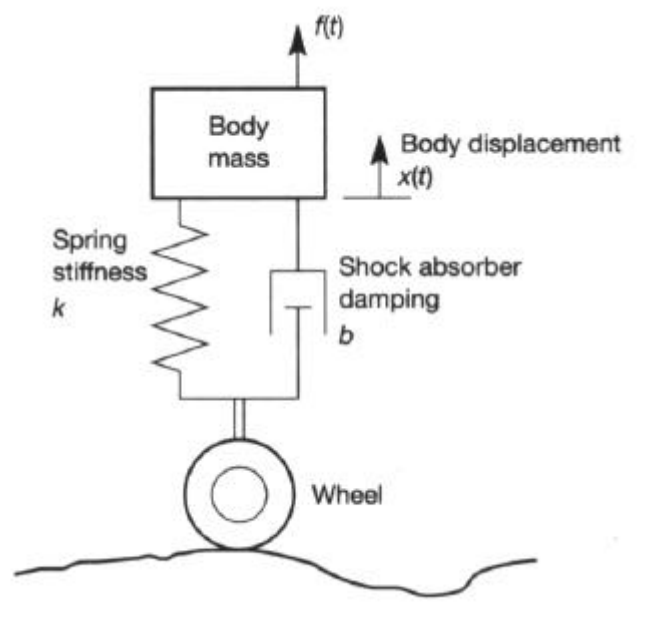
Además, incluye:

- Tres (3) integradores continuos (Euler, Adams y Runge-Kutta).
- Tres (3) integradores discretos (Euler-forward, Euler-backward y trapezoidal)
- Representaciones polo-zero, estado-tiempo y funciones de transferencia.
- Herramientas de análisis gráfico (Nyquist, Bode, Root-Locus).
- GSIM Manager y GSIM Synchronizer

6. Aplicaciones prácticas y ejemplos:

Supongamos que deseamos simular y diseñar un sistema de suspensión de un vehículo (ver Figura 6.1).

Figura 6.1



En este modelo, se asumen valores constantes para m , k , b , donde:

- m = masa del vehículo
- k = constante del resorte
- b = factor de amortiguamiento.

Según la segunda ley de Newton:

$$\sum F(t) = m \frac{d^2 x(t)}{dt^2}$$

La ecuación diferencial resultante para el sistema de suspensión es entonces:

$$m \frac{d^2 x(t)}{dt^2} + b \frac{dx(t)}{dt} + kx(t) = b \frac{du(t)}{dt} + ku(t)$$

Se asume que el resorte imparte una fuerza proporcional $f(t)$ al desplazamiento relativo entre la rueda y el vehículo $x(t)$. Para resolver esta ecuación diferencial, el sistema de suspensión del vehículo puede entonces representarse en una ó más de las siguientes formas:

- Espacio de Estado: para sistemas MIMO o para información sobre el estado interno de sistemas de control (MIMO = Multiple Input, Multiple Output)
- Funciones de transferencia: para expresiones algebraicas comunes (Hs)
- Polos-Zeros: para el análisis detallado del comportamiento dinámico de un sistema.

Mediante el sistema de ecuaciones espacio-estado, se puede convertir una ecuación de un orden superior, a un grupo de ecuaciones diferenciales de primer orden (SISO = Single Input, Single Output):

$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu}$$

$$y = \mathbf{Cx} + Du$$

Las ecuaciones espacio-estado se desarrollan definiendo el sistema en términos del estado del sistema identificado por variables de estado. El número de variables de estado en el vector \mathbf{X} se determina por el orden de la ecuación diferencial. De esta forma, el sistema puede ser completamente descrito a través de un grupo de ecuaciones diferenciales de primer orden.

Las ecuaciones resultantes son:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} k & b \\ m & m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0]u$$

Los valores de m, k, y b pueden ser incluidos directamente en estas ecuaciones, y así obtener un resultado.

Como alternativa, se puede utilizar la transformada de Laplace, mediante el cual se transforman las ecuaciones diferenciales en expresiones algebraicas. La transformada de Laplace es utilizada en la teoría de control clásica. La integral es representada por el símbolo $1/s$, y la derivada por el símbolo s .

A partir de la siguiente ecuación:

$$m \frac{d^2x(t)}{dt^2} + b \frac{dx(t)}{dt} + kx(t) = b \frac{du(t)}{dt} + ku(t)$$

Se obtiene entonces:

$$(ms^2 + bs + k)X(s) = (bs + k)U(s)$$

Utilizando el formato de la transformada de Laplace:

$$H(s) = \frac{Y(s)}{U(s)} = \frac{b_n s^n + b_{n-1} s^{n-1} + \dots + b_1 s + b_0}{a_m s^m + a_{m-1} s^{m-1} + \dots + a_1 s + a_0}$$

Se obtiene la siguiente función de transferencia:

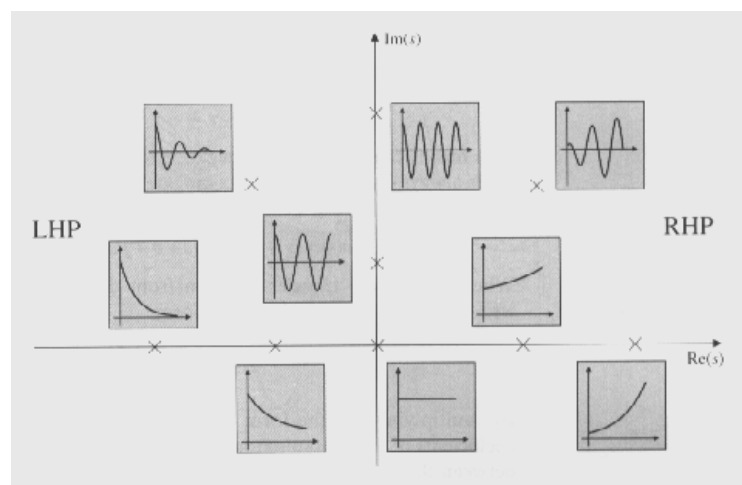
$$H(s) = \frac{X(s)}{U(s)} = \frac{bs + k}{ms^2 + bs + k}$$

Se utiliza la transformada de Laplace y las funciones de transferencia para relacionar la entrada de un sistema a la salida del mismo, en el dominio de Laplace. También se utiliza para describir sistemas lineales con tiempo constante (no variable).

Por otro lado, el método de “polos y ceros” es una representación de la función de transferencia donde las raíces del polinomio del numerador se denominan zeros y las raíces del denominador se denominan polos.

Los polos y zeros pueden ser reales o complejos. Muy útil para el análisis de estabilidad y rendimiento del sistema de control (ver Figura 6.2).

Figura 6.2



La representación en polos y ceros tiene el siguiente formato:

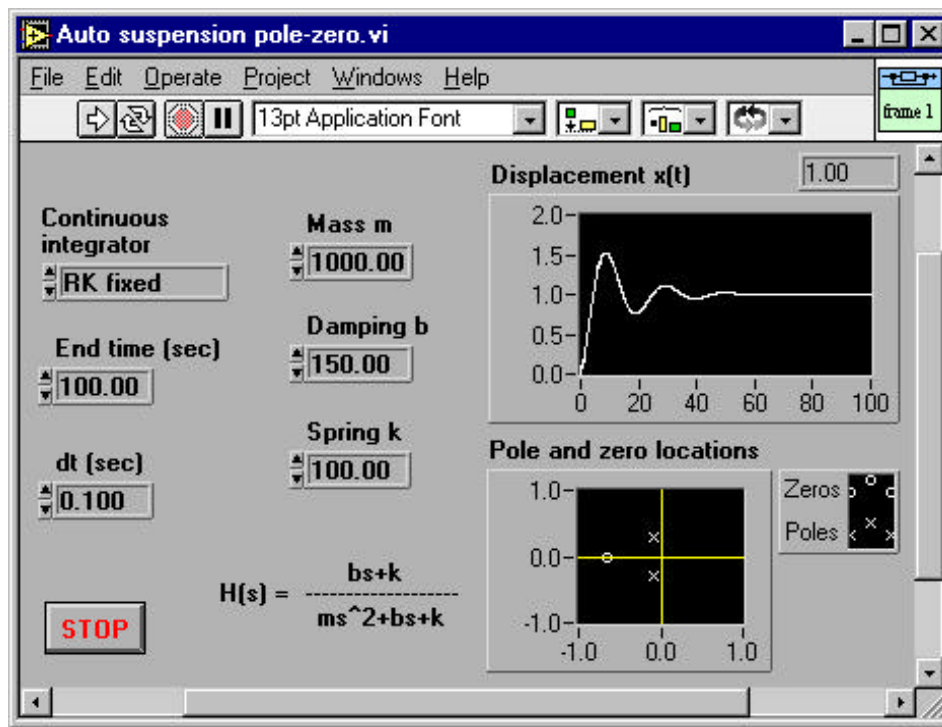
$$H(s) = \frac{K(s - z_1)(s - z_2)\dots(s - z_n)}{(s - p_1)(s - p_2)\dots(s - p_m)}$$

Utilizando la transformada de Laplace y la función de transferencia del sistema:

$$H(s) = \frac{X(s)}{U(s)} = \frac{bs + k}{ms^2 + bs + k}$$

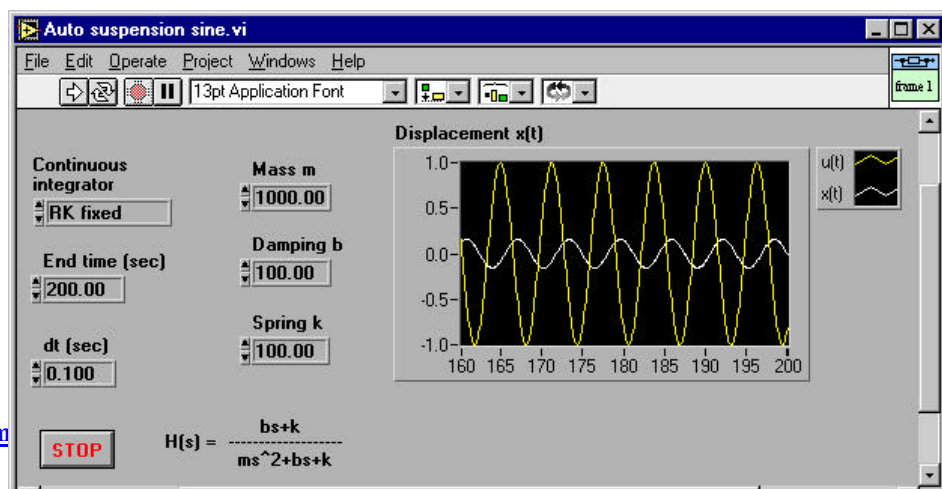
se le asigna valores a las variables, m, b y k, y se obtiene el siguiente resultado (ver Figura 6.3):

Figura 6.3



En la Figura 6.3, podemos observar que el sistema logra estabilizarse rápidamente (desplazamiento), con la ubicación de sus polos y ceros en el cuadrante izquierdo. En la Figura 6.4, podemos observar que la salida del sistema siempre tiene la misma frecuencia, pero con amplitud y fase diferentes.

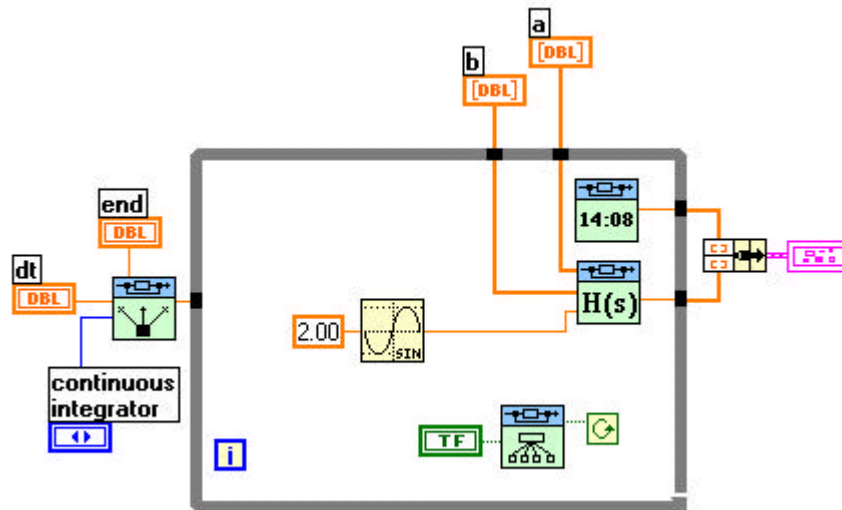
Figura 6.4



De esta forma, hemos demostrado que podemos evaluar el comportamiento del sistema (estabilidad, desempeño, etc.) en forma dinámica, utilizando el kit de herramientas de simulación y control que acompañan el lenguaje de programación gráfica.

El código fuente del programa anterior (ver Figura 6.5), consiste simplemente en unos cuantos íconos o funciones que permiten resolver la función de transferencia de acuerdo a los coeficientes de “s”, en la forma de coeficientes del “numerador”, y coeficientes del “denominador”.

Figura 6.5



Como se puede observar, se utiliza una función de seno ($\sin(2)$) como entrada al sistema. Los nodos “a” y “b” representan los coeficientes “s” de los polinomios del numerador y denominador, respectivamente. Se utiliza un integrador Runge-Kutta, con una tasa $dt = 0.1$ seg., y una duración $end = 100$ msec. Los íconos o funciones utilizados son:



Inicializador del lazo



Función de transferencia



Administrador del lazo

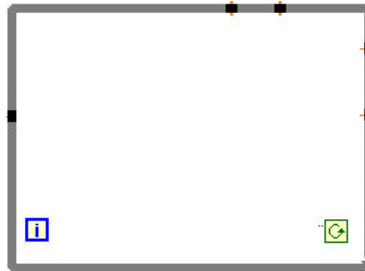


Temporizador (señal de tiempo para gráfico XY)



Generador de onda sinusoidal para estimular el sistema.

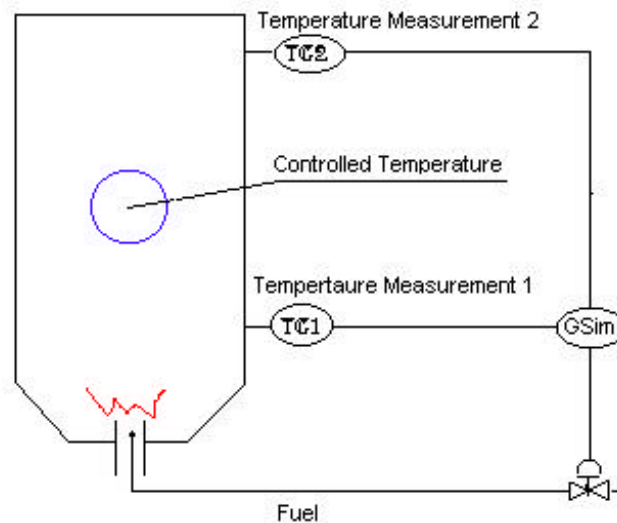
Los otros íconos ó nodos son simplemente las representaciones gráficas de los controles e indicadores incluidos en el panel frontal oó interfáz hombre-máquina. Todos ellos se ejecuntan dentro de un lazo “while” representado por el siguiente símbolo :



En resumen, podemos desarrollar un modelo del sistema de suspensión de vehículos utilizando este lenguaje gráfico, y evaluar los resultados en forma dinámica.

Veámos ahora un ejemplo diferente. En este caso, se desea controlar la temperatura en un horno (ver Figura 6.6). El problema es que se desea controlar la temperatura en el centro del horno, y los sensores de temperatura sólo pueden ser instalados en la superficie o paredes del tanque.

Figura 6.6



La filosofía ó estrategia de control a aplicar consiste en ajustar el caudal de combustible $F(s)$ de acuerdo a las temperaturas $T1(s)$ y $T2(s)$, para así controlar la temperatura en el centro del tanque (Controlled Temperature). En la Figura 6.6, el simbolo GSIM representa el controlador simulado en nuestro lenguaje gráfico. Es casi imposible controlar este proceso con un algoritmo PID debido a que ni $T1$ ni $T2$ pueden ser utilizadas como señales de retroalimentación. Una alternativa a este problema consiste en utilizar datos de campo, obtenidos mediante mediciones directas en el horno. En base a esta información, se realizanda una “identificación del sistema” o “caracterización”, y se definen las siguientes funciones de transferencia:

$$\frac{T_1(s)}{F(s)} = \frac{0.4}{s+1} \quad \text{y} \quad \frac{T_2(s)}{F(s)} = \frac{0.27e^{-5s}}{8s+1}$$

Considerando los coeficientes de transferencia de calor, propiedades de los materiales, escalas físicas, etc., podemos establecer una relación entre la temperatura central $T(s)$ y las temperaturas en las paredes del horno $T_1(s)$ y $T_2(s)$:

$$T(s) = \frac{0.38e^{-3s}}{10s+1} T_1(s) + \frac{0.62}{2s+1} T_2(s)$$

Finalmente, se obtiene la siguiente función de transferencia que relaciona $T(s)$ y $F(s)$:

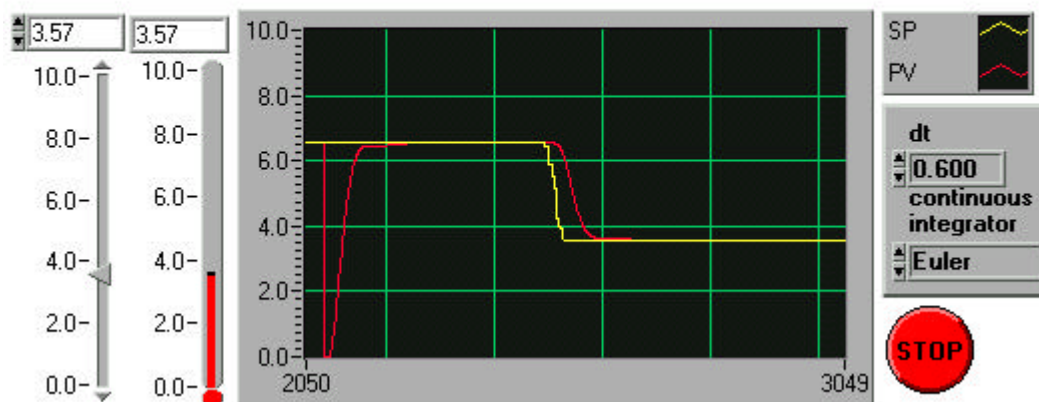
$$\frac{T(s)}{F(s)} = \frac{0.15(2s+1)(8s+1)e^{-3s} + 0.124(10s+1)(s+1)e^{-5s}}{(10s+1)(s+1)(2s+1)(8s+1)}$$

Esta función es simplificada, obteniéndose la siguiente función:

$$\frac{T(s)}{F(s)} = \frac{0.275e^{-4.5s}}{(3s+1)(8s+1)}$$

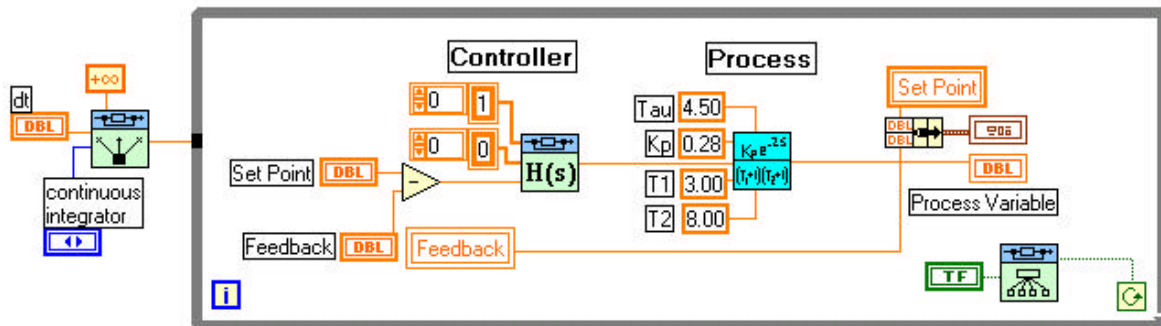
Con esta función de transferencia simplificada del proceso, se diseña el controlador adecuado y se construye el modelo del proceso/controlador para la simulación. El panel frontal de este modelo, y su respuesta, se muestran en la Figura 6.7.

Figura 6.7



El código fuente del modelo muestra en la Figura 6.8. En el, se definen los valores de Tau, Kp, T1 y T2, del proceso simulado (Process). El controlador (Controller) utiliza la función de transferencia H(s) indicada en la Figura 6.8.

Figura 6.8



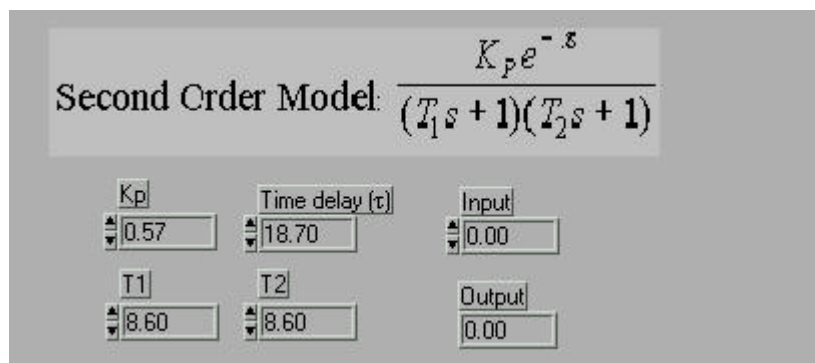
El gráfico de la Figura 6.7 indica que el controlador logra estabilizar la temperatura (PV ó Process Variable) en el centro del horno rápidamente, llevándola a coincidir con el valor del punto de consigna (SP ó Setpoint).

El modelo del proceso es en sí mismo una sub-rutina, representada por el ícono :



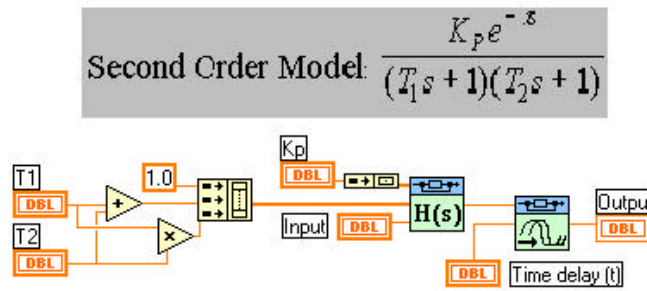
Este ícono o función, es un sistema de 2do. orden que incluye el siguiente panel frontal (Figura 6.9):

Figura 6.9



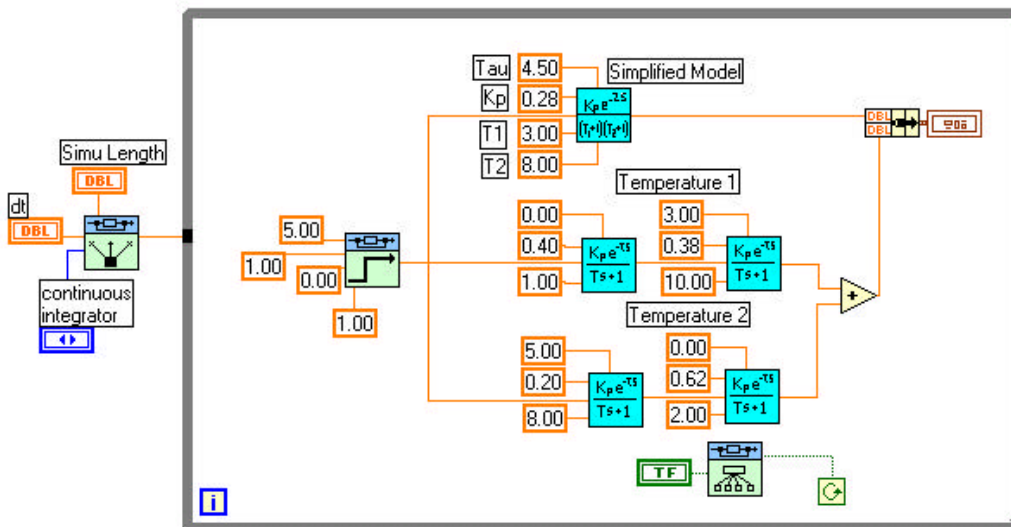
El código fuente de esta sub-rutina es el siguiente (ver Figura 6.10):

Figura 6.10



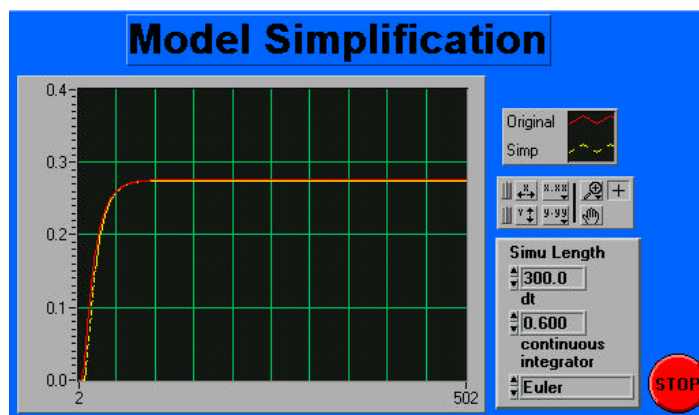
Para llegar a este modelo o función simplificado del proceso, se utilizó el siguiente programa (ver Figura 6.11).

Figura 6.11



Con este sencillo programa, podemos verificar que el modelo simplificado del proceso es equivalente al modelo original, según se indica en la Figura 6.12. Se utiliza una función de pulso o escalón para estimular el sistema y evaluar su respuesta. Las líneas rojas y amarillas nos indican que el comportamiento de ambos modelos es similar ante la señal de estímulo de tipo escalón.

Figura 6.12



En resumen, el controlador toma el valor del punto de consigna (setpoint, SP) y los coeficientes de la función de transferencia $H(s)$, y calcula la salida requerida para controlar la temperatura en el proceso simulado. Se utiliza un integrador EULER, a una tasa dT de 0.6 seg.

El proceso a controlar está representado en el sistema de 2do. Orden indicado en la figura 6.10. Este sistema es entonces utilizado como modelo matemático del proceso a controlar, tomando las temperaturas $T1$ y $T2$, la ganancia Kp y la constante de tiempo t (tau) y el valor de entrada (input) generado por el controlador, generándose una salida (output) que es retroalimentada al mismo controlador (controller) en forma continua, tal y como se indica en la Figura 6.8.

La Figura 6.7 nos indica que el controlador logra mantener la temperatura o variable de proceso (línea roja, PV) alrededor del punto de consigna (línea amarilla, SP), que la repuesta del mismo es rápida, y que la estabilidad de la variable del proceso, alrededor del SP, es casi instantánea.

7. Conclusiones:

Como hemos demostrado, el lenguaje gráfico utilizado junto al kit de Simulación y Control (GSIM) constituyen una poderosa herramienta que nos permite construir modelos matemáticos de procesos y sistemas continuos, discretos, lineales y no lineales, para luego simularlos en forma dinámica y evaluar su comportamiento, sin necesidad de utilizar modelos físicos.

Una vez realizada la simulación con el modelo matemático utilizando señales simuladas, este lenguaje gráfico permite que se sustituyan las señales simuladas por señales reales (voltaje, corriente, etc.) y de esa manera poder evaluar el comportamiento del sistema o del controlador ante señales reales.

Finalmente, se puede sustituir el modelo matemático tanto del controlador como del proceso a controlar, por los reales, e implantar el sistema de control deseado, previa evaluación y optimización basada en la simulación con modelos matemáticos.

Las ventajas de un lenguaje de programación gráfica son amplias. Entre muchas otras cosas, nos permite simular con modelos matemáticos los procesos y sistemas de control, pero también nos permite implantarlos para controlar el proceso real a través de tarjetas de adquisición de datos y control con entradas y salidas analógicas y digitales. El paso de la simulación a la implantación es casi transparente para el usuario.

La programación gráfica permite que tanto los programadores experimentados, como los novatos, puedan desarrollar programas poderosos sin necesidad de ser un verdadero experto en programación.

De las opciones de utilizar un simulador versus un lenguaje de simulación, la que hemos utilizado en este trabajo representa una especie de híbrido que nos ofrece la flexibilidad de un lenguaje con la sencillez de un simulador.

Referencias:

- National Instruments Sales Conference, , Austin, Texas, 1999
- National Instruments NIWEEK 99 Proceedings, , Austin, Texas, 1999
- LabVIEW User Manual
- Simulation and Control Toolkit for G Manual