Jornada sobre Técnicas de Modelado y Simulación

El Software PowerDEVS

Ernesto Kofman y Federico Bergero

Laboratorio de Sistemas Dinámicos y Procesamiento de la Información
FCEIA - Universidad Nacional de Rosario.
CIFASIS – CONICET. Argentina

Organización de la Presentación

1. Simulación de Sistemas Discretos y Continuos con PowerDEVS
2. Simulación en Tiempo Real
3. Práctica con PowerDEVS

Organización de la Presentación

1. Simulación de Sistemas Discretos y Continuos con PowerDEVS
2. Simulación en Tiempo Real
3. Práctica con PowerDEVS

Formalismo DEVS

Discrete EVent System specification (DEVS)

DEVS es un formalismo general para modelar y simular sistemas de eventos discretos.

Un modelo DEVS puede ser visto como un autómata que procesa una serie de eventos de entrada y genera una serie de eventos de salida.

El formalismo DEVS es muy general. Permite representar cualquier clase de sistemas discretos, y, mediante aproximaciones, sistemas continuos y sistemas híbridos.

PowerDEVS

PowerDEVS es un entorno de simulación de modelos DEVS de propósito general, orientado a la simulación de sistemas híbridos, realizado y mantenido por nuestro grupo de investigación en la FCEIA.

Cuenta con dos partes principales:

Entorno gráfico Es la interfaz que ve el usuario regular de PowerDEVS. Permite describir modelos DEVS en forma gráfica y sencilla.

Motor de simulación Este módulo es el núcleo de toda la herramienta. El código generado por la interfaz, luego es compilado junto con el motor de simulación obteniéndose el programa que simula el modelo DEVS.

PowerDEVS

PowerDEVS es un entorno de simulación de modelos DEVS de propósito general, orientado a la simulación de sistemas híbridos, realizado y mantenido por nuestro grupo de investigación en la FCEIA.

Cuenta con dos partes principales:

Entorno gráfico Es la interfaz que ve el usuario regular de PowerDEVS. Permite describir modelos DEVS en forma gráfica y sencilla.

Motor de simulación Este módulo es el núcleo de toda la herramienta. El código generado por la interfaz, luego es compilado junto con el motor de simulación obteniéndose el programa que simula el modelo DEVS.

Características de PowerDEVS

- Posee una interfaz gráfica donde pueden especificarse modelos DEVS (acoplando DB).
- Amplia librería para la simulación de sistemas continuos (métodos de QSS).
- Implementado en C++.
- Corre sobre Windows y Linux.
- Soporte para tiempo real.
- Conexión con Scilab.
- Software libre.

Características de PowerDEVS

- Posee una interfaz gráfica donde pueden especificarse modelos DEVS (acoplando DB).
- Amplia librería para la simulación de sistemas continuos (métodos de QSS).
- Implementado en C++.
- Corre sobre Windows y Linux.
- Soporte para tiempo real.
- Conexión con Scilab.
- Software libre.

Características de PowerDEVS

- Posee una interfaz gráfica donde pueden especificarse modelos DEVS (acoplando DB).
- Amplia librería para la simulación de sistemas continuos (métodos de QSS).
- Implementado en C++.
- Corre sobre Windows y Linux.
- Soporte para tiempo real.
- Conexión con Scilab.
- Software libre.

Características de PowerDEVS

- Posee una interfaz gráfica donde pueden especificarse modelos DEVS (acoplando DB).
- Amplia librería para la simulación de sistemas continuos (métodos de QSS).
- Implementado en C++.
- Corre sobre Windows y Linux.
- Soporte para tiempo real.
- Conexión con Scilab.
- Software libre.

Características de PowerDEVS

- Posee una interfaz gráfica donde pueden especificarse modelos DEVS (acoplando DB).
- Amplia librería para la simulación de sistemas continuos (métodos de QSS).
- Implementado en C++.
- Corre sobre Windows y Linux.
- Soporte para tiempo real.
- Conexión con Scilab.
- Software libre.

Características de PowerDEVS

- Posee una interfaz gráfica donde pueden especificarse modelos DEVS (acoplando DB).
- Amplia librería para la simulación de sistemas continuos (métodos de QSS).
- Implementado en C++.
- Corre sobre Windows y Linux.
- Soporte para tiempo real.
- Conexión con Scilab.
- Software libre.

Características de PowerDEVS

- Posee una interfaz gráfica donde pueden especificarse modelos DEVS (acoplando DB).
- Amplia librería para la simulación de sistemas continuos (métodos de QSS).
- Implementado en C++.
- Corre sobre Windows y Linux.
- Soporte para tiempo real.
- Conexión con Scilab.
- Software libre.

Ejemplo Modelo Atómico – Generador

El siguiente modelo genera eventos con números al azar entre 0 y 1. El tiempo transcurrido entre eventos sucesivos es también aleatorio (entre 0 y 1).

$$G =< X, Y, S, \delta_{\text{int}}, \delta_{\text{ext}}, \lambda, ta >$$
$$Y = S = \mathbb{R}^{+}$$
$$\delta_{\text{int}}(s) = RND[0, 1]$$
$$\lambda(s) = RND[0, 1]$$
$$ta(s) = s$$

Modelo en PowerDEVS:

Ejemplo Modelo Atómico – Procesador

Un procesador recibe trabajos representados por números reales positivos, que indican el tiempo que demora en procesarse dicho trabajo. Transcurrido dicho tiempo, el procesador emite un evento con el valor “1” indicando la finalización. Si recibe un trabajo mientras está procesando, el nuevo trabajo es descartado.

$$P =< X, Y, S, \delta_{\text{int}}, \delta_{\text{ext}}, \lambda, ta >$$

$$X = Y = \mathbb{R}^+$$

$$S = \mathbb{R}^+ \times \{true, false\}$$

$$\delta_{\text{ext}}(s, e, x) = \delta_{\text{ext}}((\sigma, busy), e, x) = \begin{cases} (\sigma - e, true) & \text{si } busy = true \\ (x, true) & \text{en otro caso} \end{cases}$$

$$\delta_{\text{int}}(s) = \infty$$

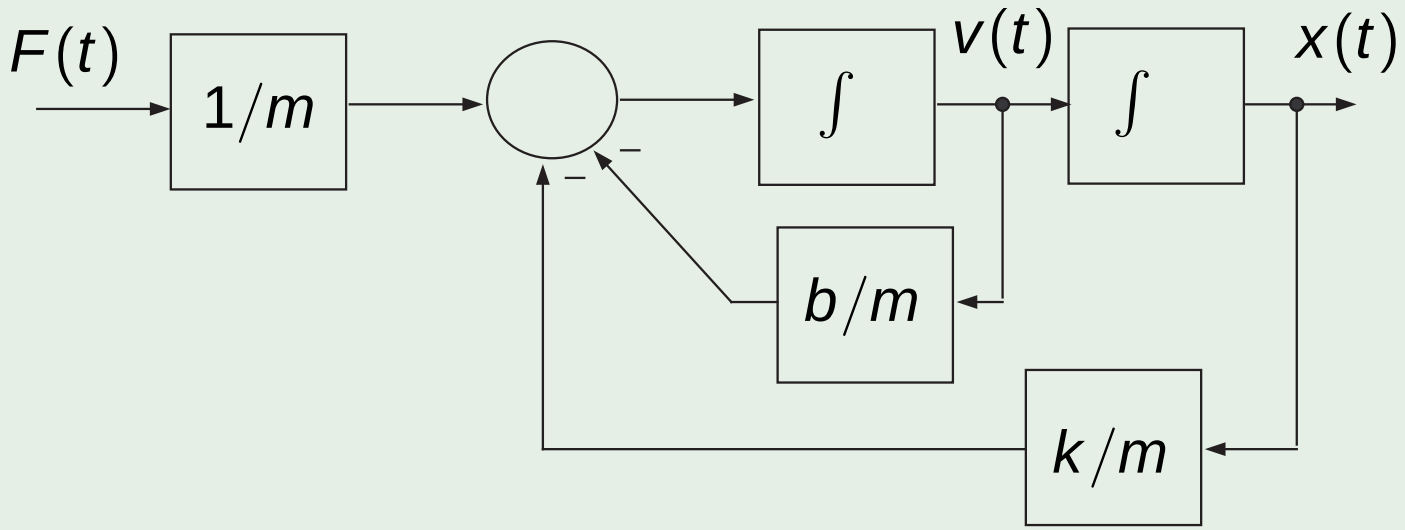
$$\lambda(s) = 1$$

Ejemplo Modelo Acoplado

Generador + Procesador

Ejemplo – Sistema Continuo

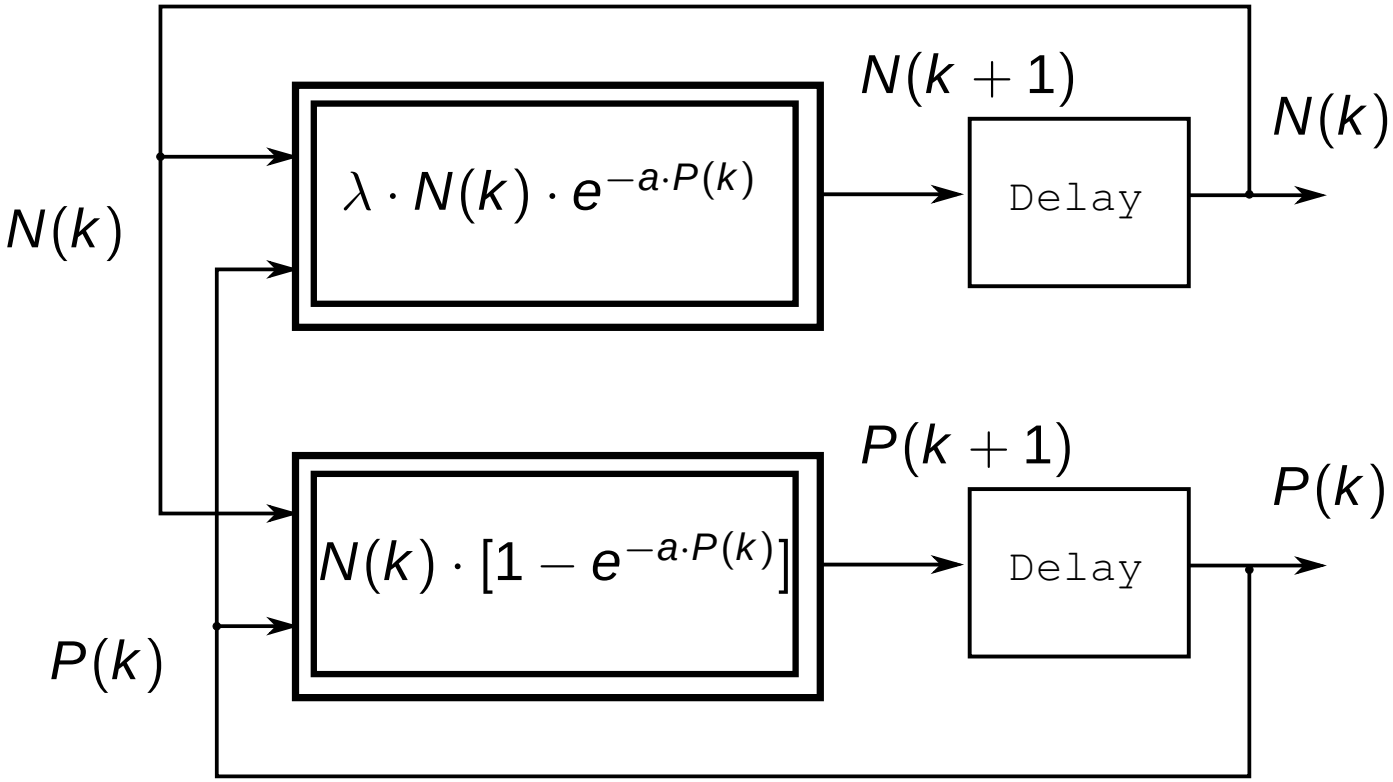
Sistema Masa Resorte



$$\dot{x}(t) = v(t)$$

$$\dot{v}(t) = -\frac{k}{m}x(t) - \frac{b}{m}v(t) + \frac{1}{m}F(t)$$

Ejemplo – Sistema de Tiempo Discreto



$$N(t_{k+1}) = \lambda \cdot N(t_k) \cdot e^{-a \cdot P(t_k)}$$
$$P(t_{k+1}) = N(t_k) \cdot [1 - e^{-a \cdot P(t_k)}]$$

Nicholson–Bailey

Organización de la Presentación

1 Simulación de Sistemas Discretos y Continuos con PowerDEVS

2 Simulación en Tiempo Real

3 Práctica con PowerDEVS

PowerDEVS en Tiempo Real

Simulación en Tiempo Real

Sistemas de tiempo real

Son sistemas en los cuales las operaciones tienen una restricción temporal.

- Los resultados obtenidos no sólo deben ser correctos sino que deben ser “entregados” en el momento correcto.
- Deben responder a estímulos cumpliendo también restricciones temporales.
- Las aplicaciones típicas son Hardware in the Loop y Man in the Loop.

Módulos desarrollados

Módulos desarrollados

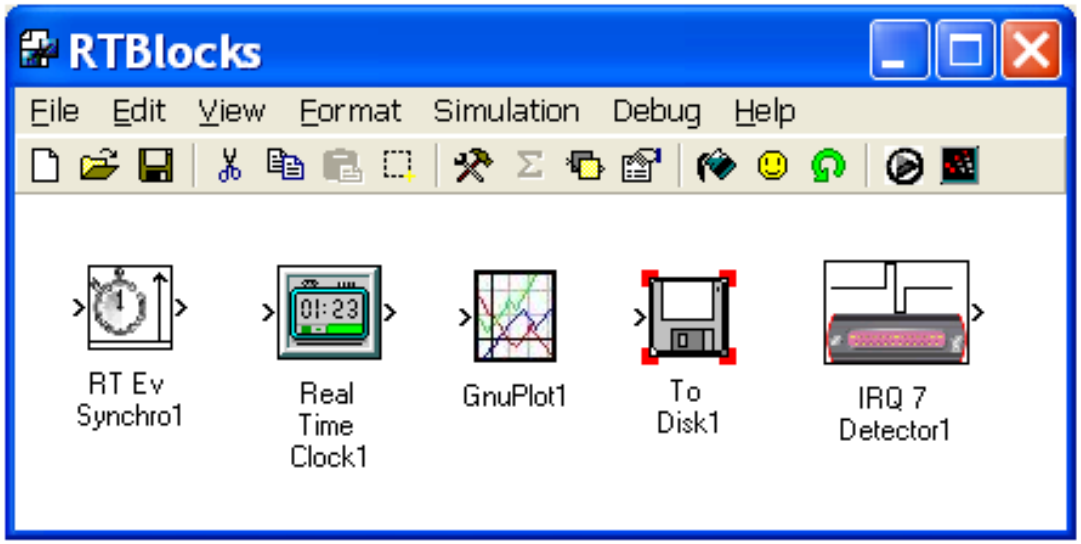
Para poder realizar simulaciones en tiempo real, se agregaron varios módulos al motor de simulación de PowerDEVS

- Módulo de sincronización
- Módulo de interrupciones
- Módulo de archivos (en tiempo real no puede utilizarse el file system de Linux)
- Módulo de medición de tiempo real (RTAI provee un reloj mucho más preciso que el de Linux)

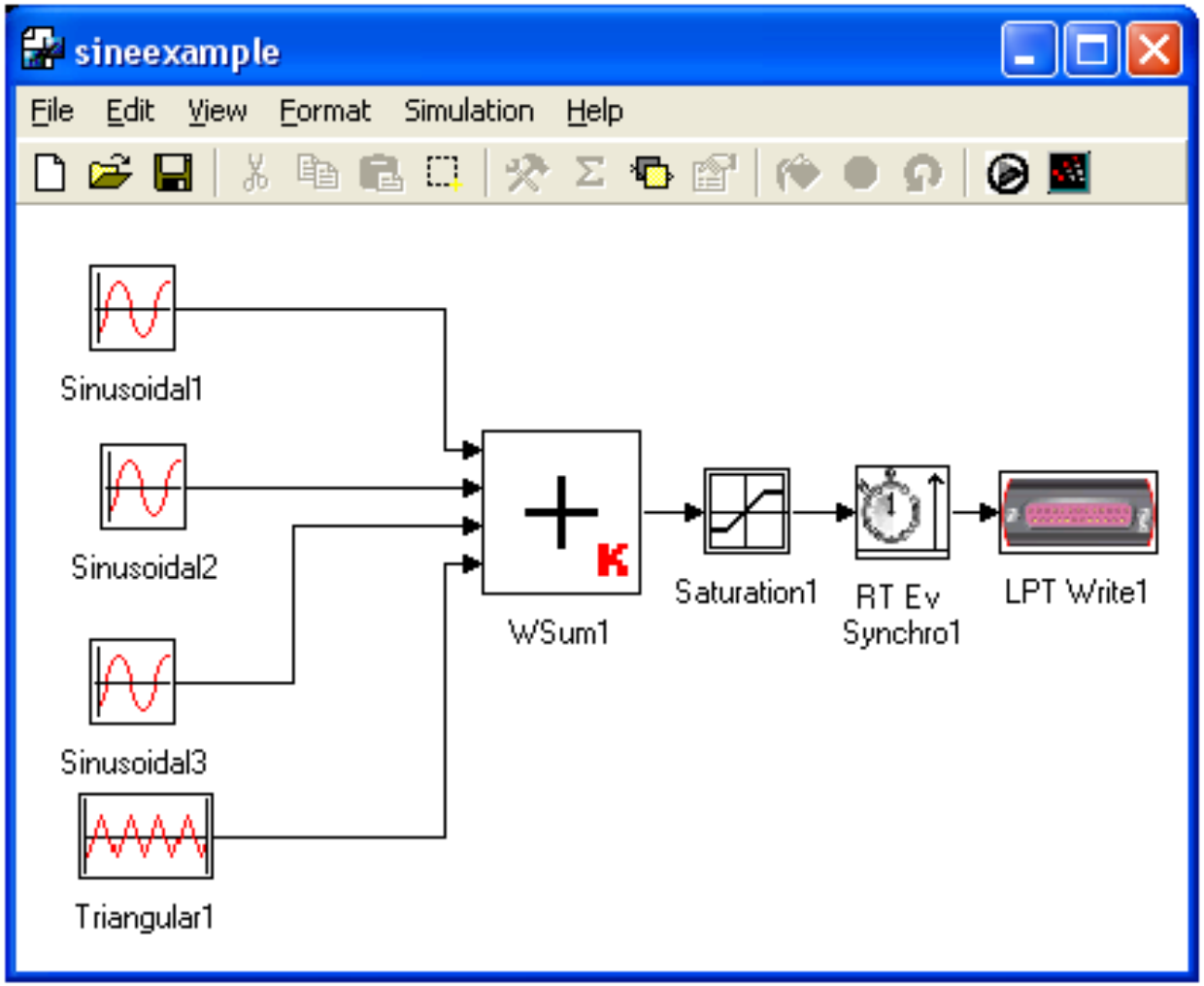
Modelos agregados a la librería

Para hacer uso de los nuevos servicios agregados al motor, expandimos las librerías de PowerDEVS con nuevos bloques para incluir directamente en los modelos.

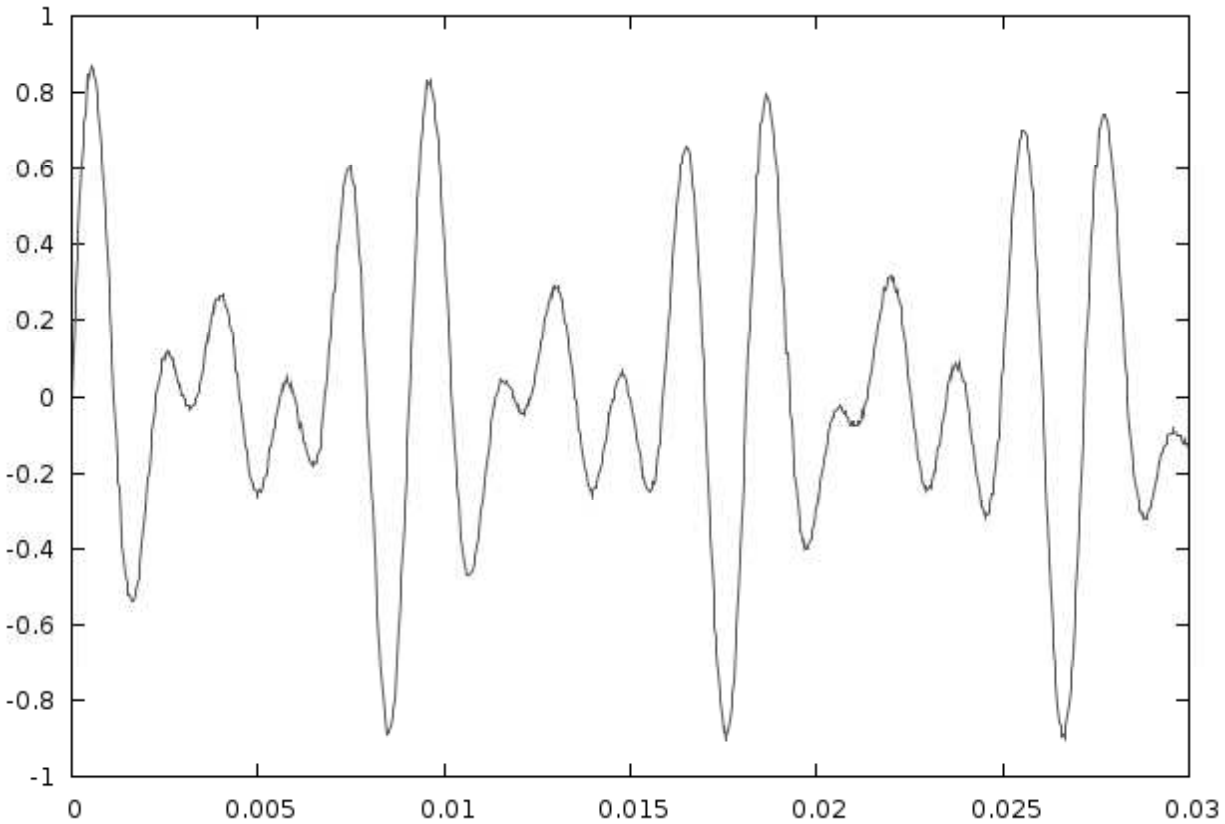
- RTWait
- RTClock
- GNUPlot
- ToDisk
- IRQDetector



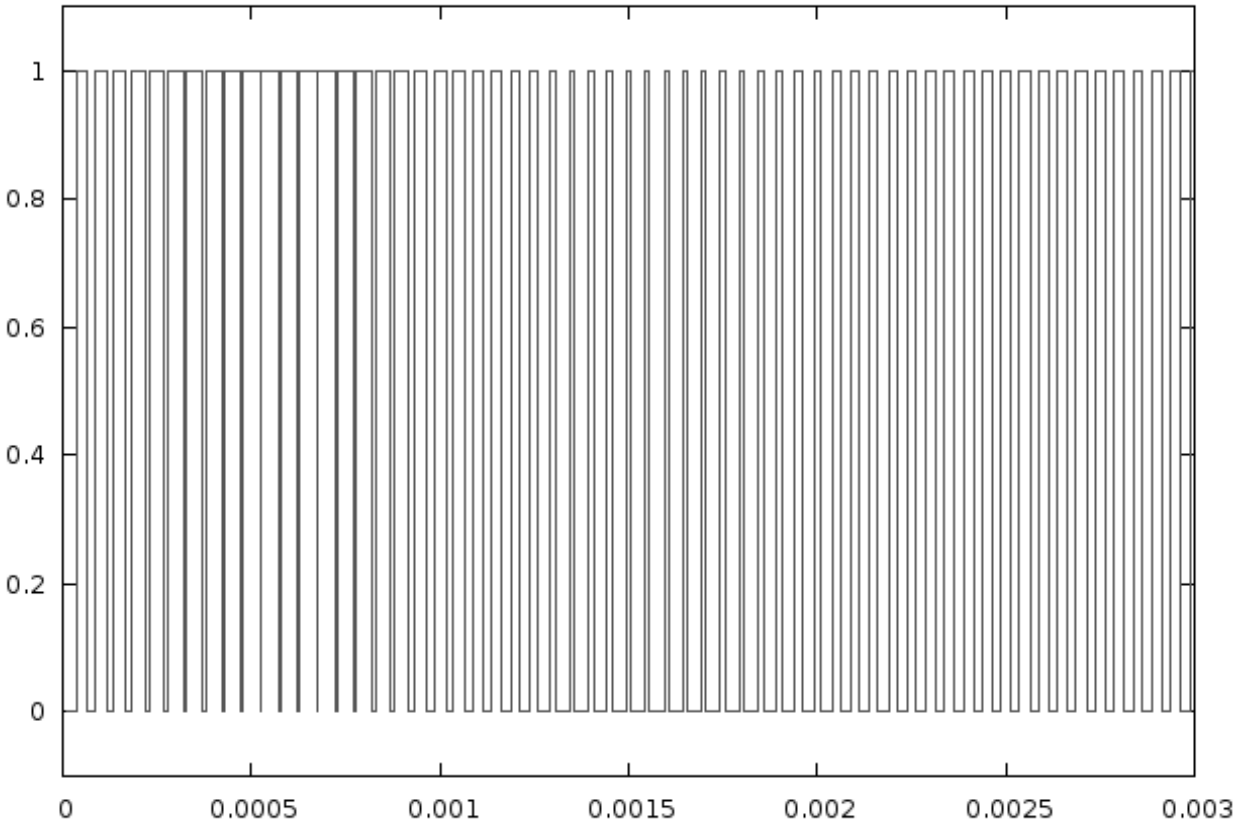
Sonido por paralelo - Modelo DEVS



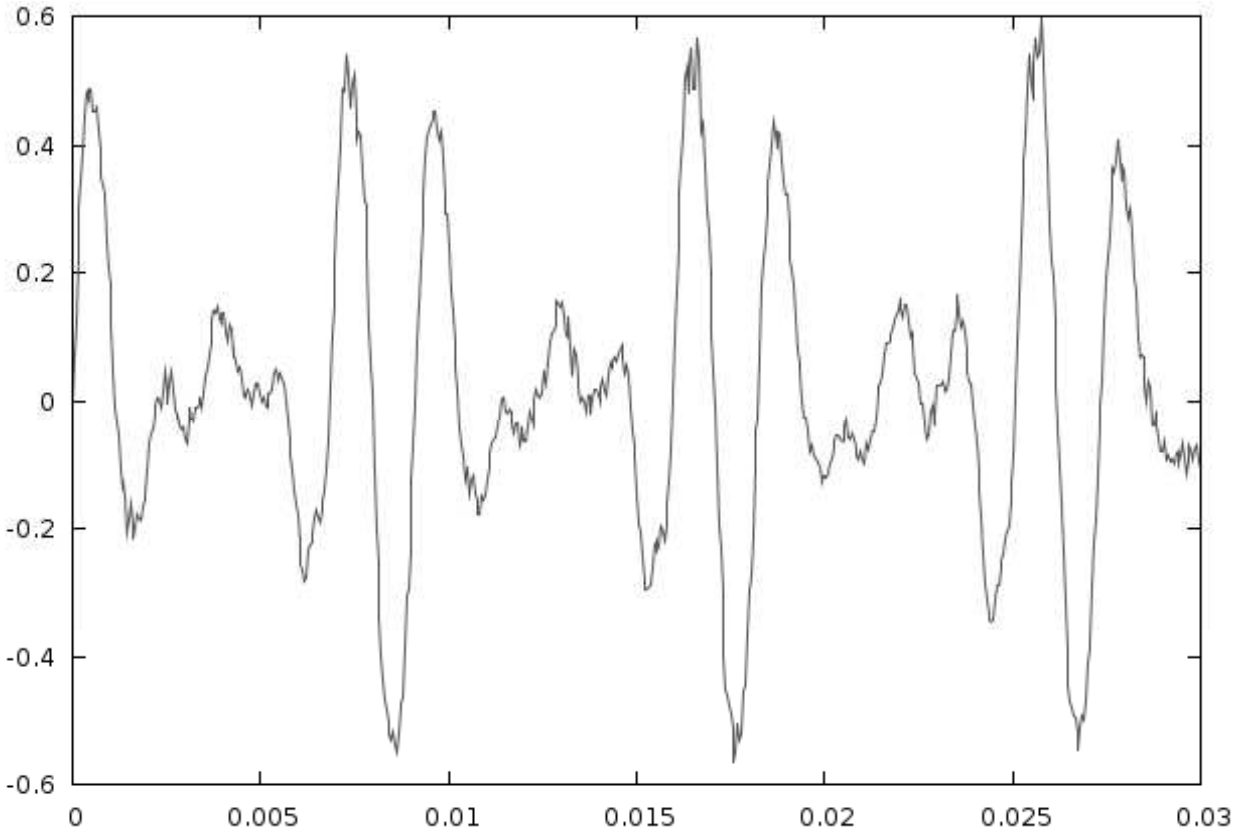
Sonido por paralelo - Señal original



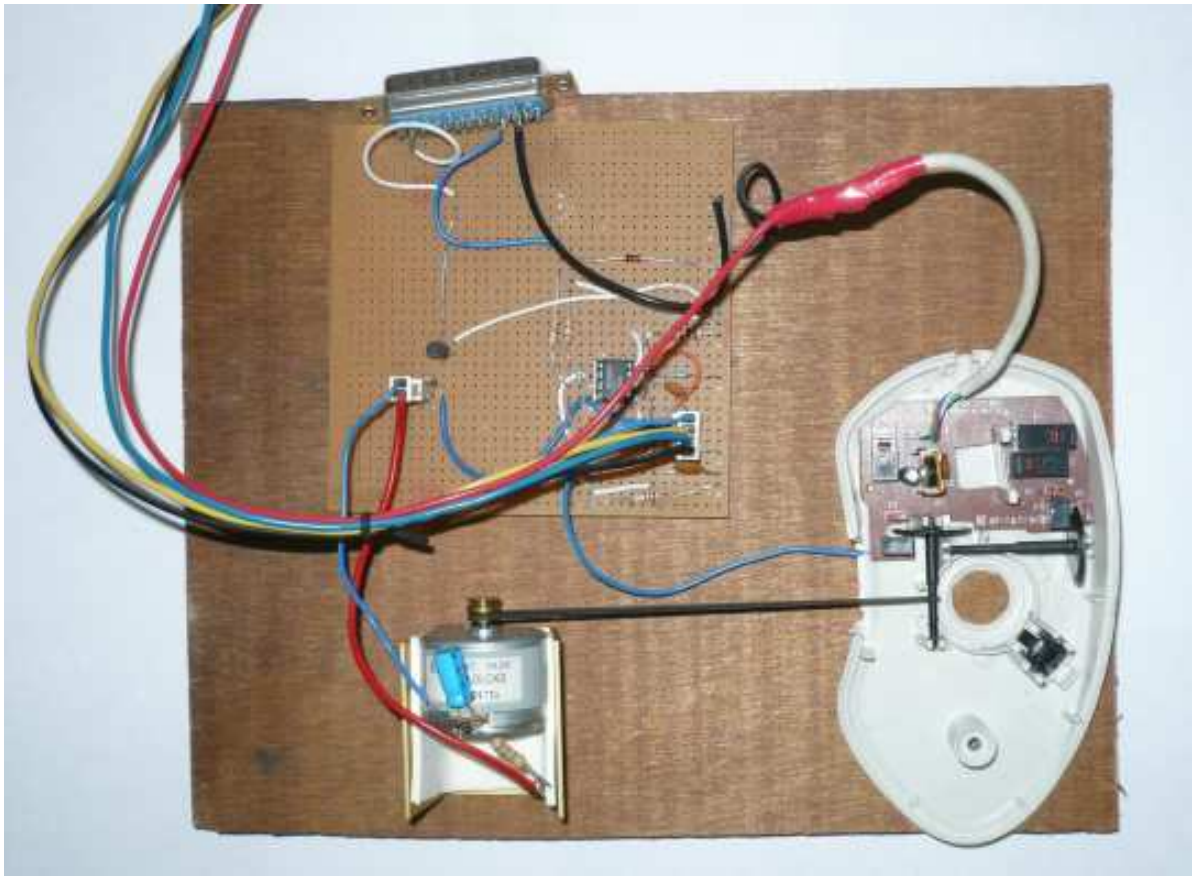
Sonido por paralelo - Señal modulada por PWM



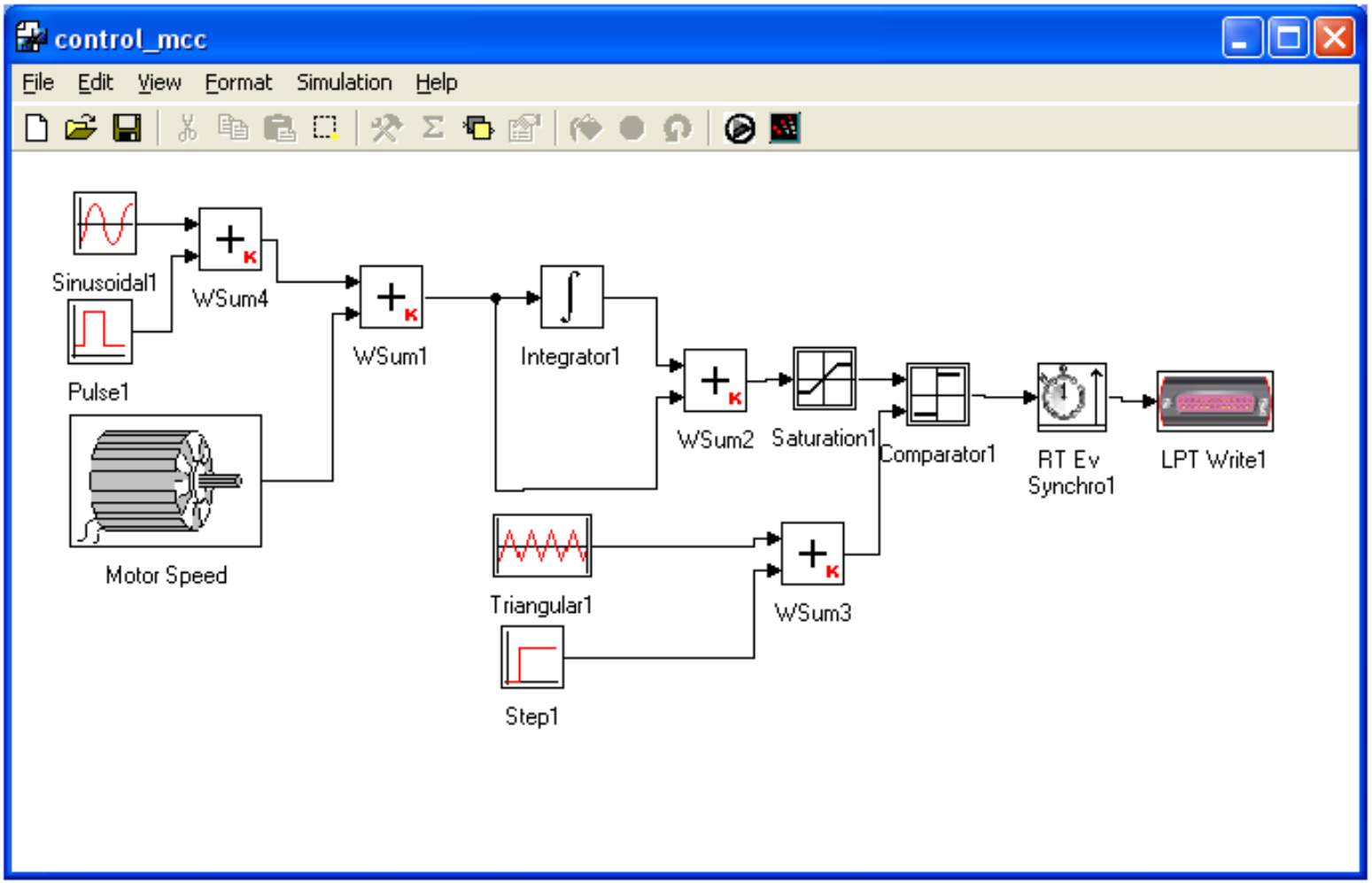
Sonido por paralelo - Señal capturada por un mic.



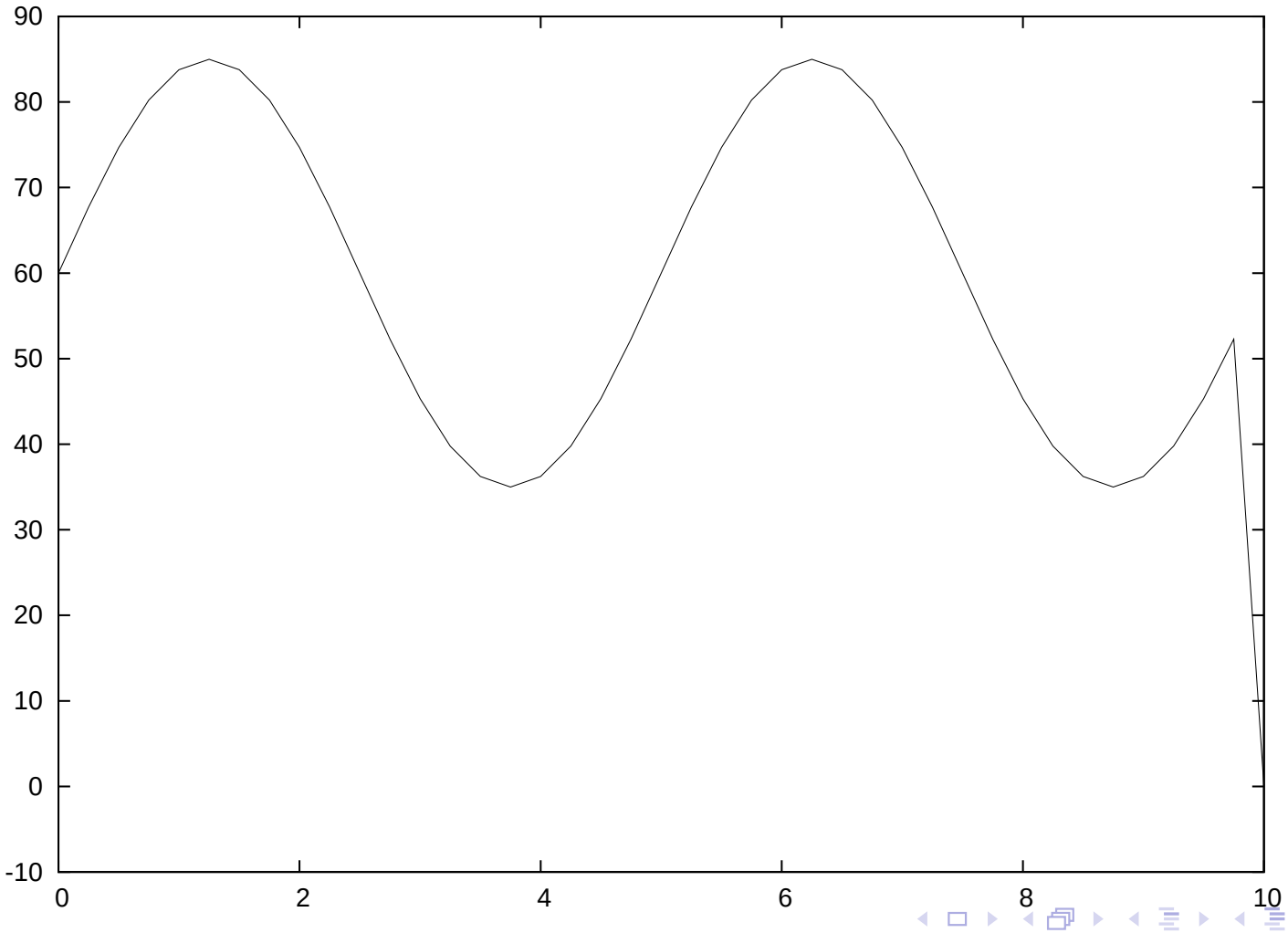
Control de un motor - Dispositivo



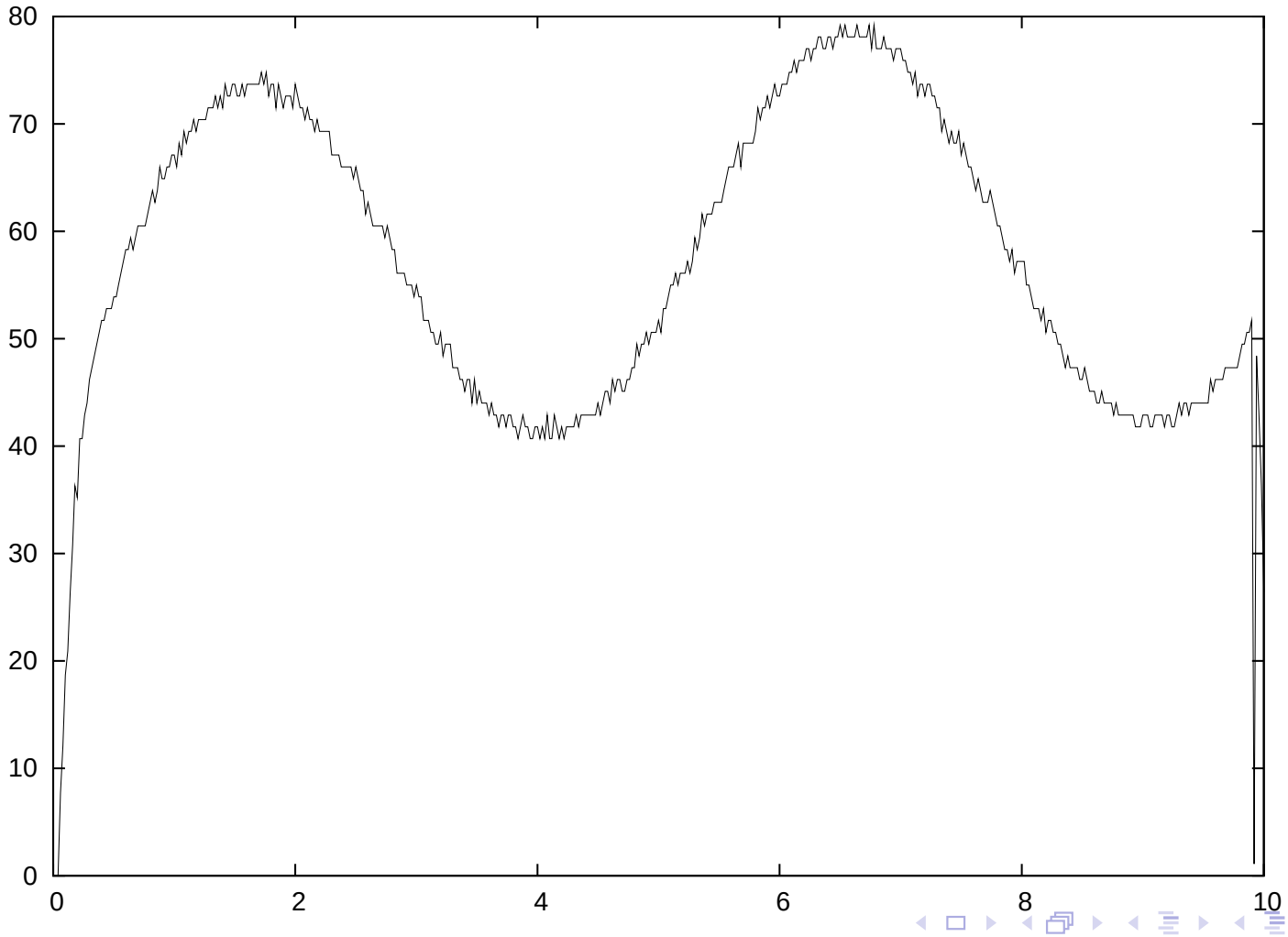
Control de un motor - Modelo DEVS



Control de un motor - Velocidad de referencia



Control de un motor - Velocidad medida



PowerDEVS – Latencia

- Utilizando la herramienta desarrollada obtenemos latencias de simulación del orden de los 2000 ηs a 5000 ηs.
- En cuanto a la latencia de interrupción (respuesta a estímulos externos) obtenemos una latencia cercana a 15000 ηs

PowerDEVS – Latencia

- Utilizando la herramienta desarrollada obtenemos latencias de simulación del orden de los 2000 ηs a 5000 ηs.
- En cuanto a la latencia de interrupción (respuesta a estímulos externos) obtenemos una latencia cercana a 15000 ηs

Organización de la Presentación

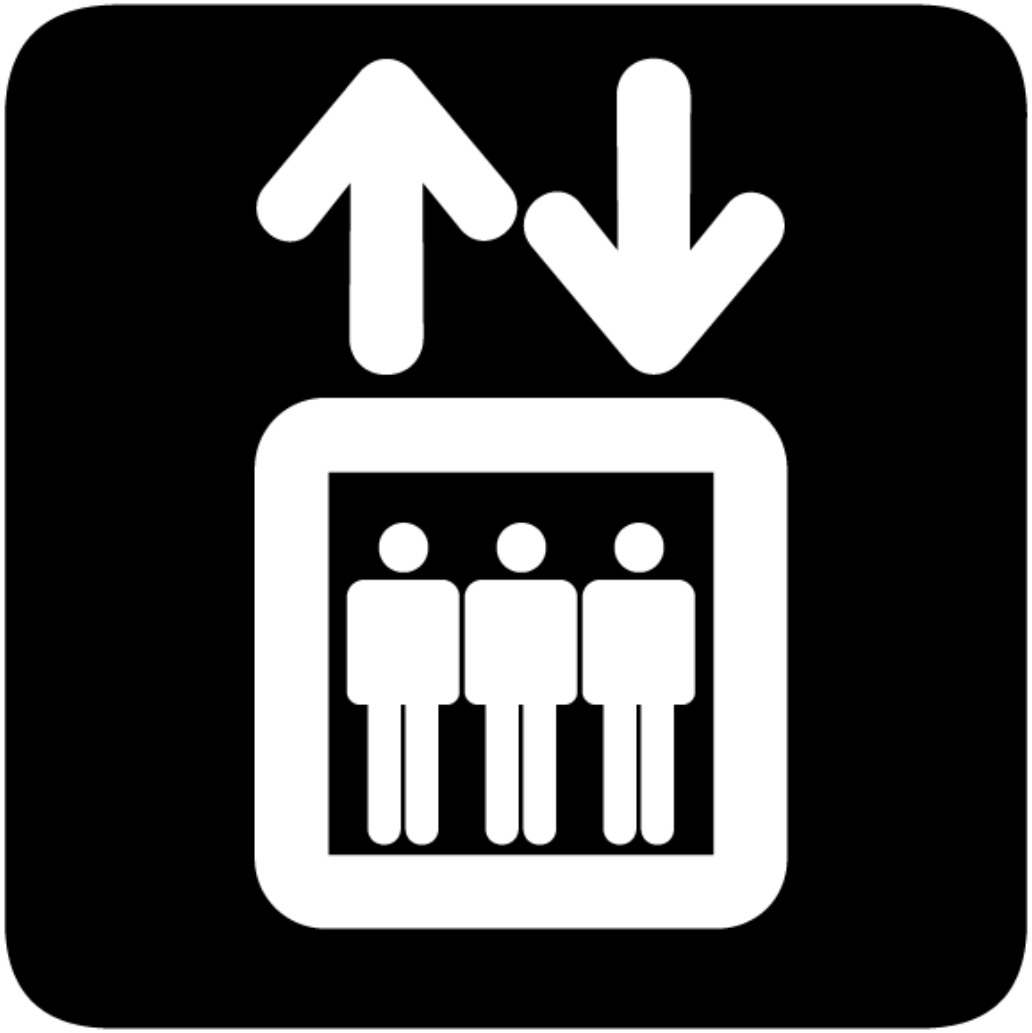
1 Simulación de Sistemas Discretos y Continuos con PowerDEVS

2 Simulación en Tiempo Real

3 Práctica con PowerDEVS

Práctica con PowerDEVS

Ejemplo - Ascensor



Ejemplo - Ascensor

- Simularemos un modelo de un ascensor de un edificio de 5 pisos, incluyendo los pedidos, el control y el ascensor.
- Los pedidos vienen en momentos aleatorios y desde pisos aleatorios.
- Los pedidos no deben perderse.
- Inicialmente, el ascensor tarda un tiempo constante en subir o bajar un piso.
- El ascensor tiene sensores que notifican cuando se llega a un piso.

Ejemplo - Ascensor

- Simularemos un modelo de un ascensor de un edificio de 5 pisos, incluyendo los pedidos, el control y el ascensor.
- Los pedidos vienen en momentos aleatorios y desde pisos aleatorios.
- Los pedidos no deben perderse.
- Inicialmente, el ascensor tarda un tiempo constante en subir o bajar un piso.
- El ascensor tiene sensores que notifican cuando se llega a un piso.

Ejemplo - Ascensor

- Simularemos un modelo de un ascensor de un edificio de 5 pisos, incluyendo los pedidos, el control y el ascensor.
- Los pedidos vienen en momentos aleatorios y desde pisos aleatorios.
- Los pedidos no deben perderse.
- Inicialmente, el ascensor tarda un tiempo constante en subir o bajar un piso.
- El ascensor tiene sensores que notifican cuando se llega a un piso.

Ejemplo - Ascensor

- Simularemos un modelo de un ascensor de un edificio de 5 pisos, incluyendo los pedidos, el control y el ascensor.
- Los pedidos vienen en momentos aleatorios y desde pisos aleatorios.
- Los pedidos no deben perderse.
- Inicialmente, el ascensor tarda un tiempo constante en subir o bajar un piso.
- El ascensor tiene sensores que notifican cuando se llega a un piso.

Ejemplo - Ascensor

- Simularemos un modelo de un ascensor de un edificio de 5 pisos, incluyendo los pedidos, el control y el ascensor.
- Los pedidos vienen en momentos aleatorios y desde pisos aleatorios.
- Los pedidos no deben perderse.
- Inicialmente, el ascensor tarda un tiempo constante en subir o bajar un piso.
- El ascensor tiene sensores que notifican cuando se llega a un piso.

Ejemplo - Ascensor

Modelo del generador de pedidos:

- el generador de pedidos emite eventos con números del 0 al 4 indicando el piso al que hay que ir.
- El tiempo entre pedidos es un número real (aleatorio) entre 0 y 20.

Modelo de la cola de pedidos:

- La cola recibe los eventos del generador por el primer puerto de entrada y los guarda.
- Recibe las notificaciones de que el ascensor está disponible por el segundo puerto de entrada y en este caso envía el primer pedido al ascensor.
- Cuando recibe un evento del generador y sabe que el ascensor está disponible, envía el pedido inmediatamente.

Ejemplo - Ascensor

Modelo del control del ascensor:

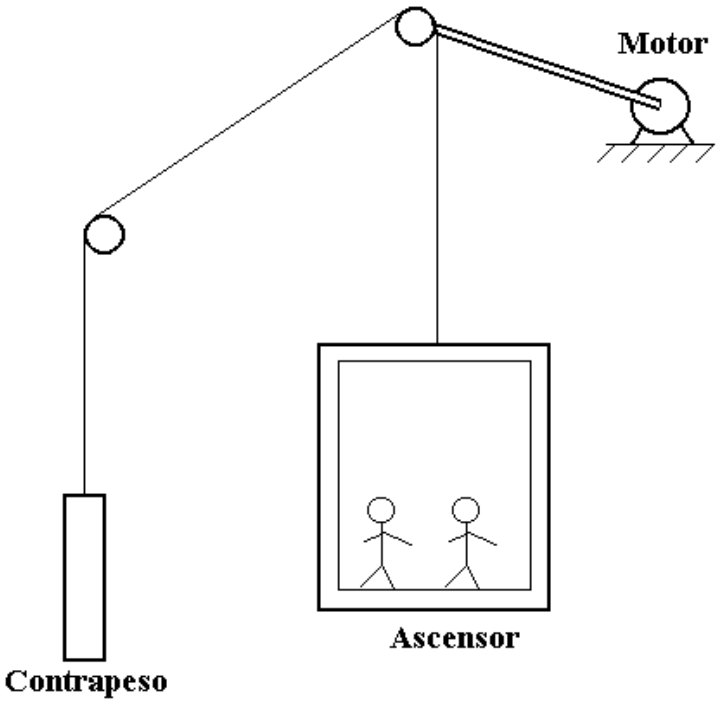
- Este modelo recibe los pedidos desde la cola y las señales de los sensores que indican cuando el ascensor pasa por cada piso.
- Por otro lado, envía una señal al ascensor que vale 1 si debe subir, -1 si debe bajar y 0 si debe detenerse.
- Además, envía una señal a la cola cuando el ascensor está disponible dos segundos después que éste arribó al piso pedido.

Modelo del ascensor:

- El ascensor recibe las señales del control (1,0 o -1).
- Sube o baja con una velocidad constante de 0.5 m/s.
- Cada vez que llega a un piso (que consideraremos de 1m de altura) emite un evento indicando el valor del piso alcanzado.

Ejemplo - Ascensor

Un modelo más detallado del ascensor es considerar que el mismo está impulsado por un motor.



$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = \frac{1}{m_a + m} \cdot (-b \cdot x_2 + \frac{k_m}{r} \cdot x_3 - m \cdot g)$$

$$\dot{x}_3 = \frac{1}{L_a}(-\frac{k_m}{r} \cdot x_2 - R_a \cdot x_3 + x_4)$$

$$\dot{x}_4 = \frac{1}{\tau}(-x_4 + U_m \cdot u_c(t))$$

Ejemplo - Ascensor

Modelo de los sensores:

- Podemos complementar el modelo continuo del ascensor con modelos de los sensores, que reciban la altura del ascensor y emitan un evento (con el número del piso correspondiente) cada vez que este pase por un piso.
- Para esto podemos utilizar un bloque de la librería 'Hybrid' de PowerDEVS que detecta cruces de las señales por valores predeterminados.

Luego, puede reemplazarse el modelo discreto del ascensor por el modelo continuo y los sensores, sin modificar ni el control, ni la cola, ni el generador.

Este nuevo modelo constituye un Sistema Híbrido.