DESARROLLO DE LABORATORIOS VIRTUALES, INTERACTIVOS Y REMOTOS UTILIZANDO EASY JAVA SIMULATIONS Y MODELOS SIMULINK

G. FARIAS[*], F. ESQUEMBRE [†], J. SANCHEZ[*], S. DORMIDO[*], H. VARGAS[*], S. DORMIDO-CANTO[*], R. DORMIDO[*], N. DURO[*], M. CANTO[*].

[*]*Departamento de Informática y Automática, UNED. Ciudad Universitaria, C.P. 28040, Madrid, España*

[†]*Departamento de Matemáticas,Universidad de Murcia. Campus de Espinardo, C.P. 30071, Murcia, España*

E-mails: {gfarias,hvargas}@bec.uned.es, fem@um.es, {jsanchez,sdormido,sebas,raquel,nduro, macanto}@dia.uned.es

Resumen— Este trabajo describe como utilizar Easy Java Simulation (Ejs) y el servidor Java Internet Matlab (Jim), para construir laboratorios virtuales agregando interactividad a modelos Simulink remotos. Ejs es un paquete de software gratuito diseñado para crear simulaciones interactivas en Java, mientras que Jim es una extensión de Ejs diseñada para soportar el enlace entre Ejs y un modelo Simulink remoto. Ejs ya proporciona la conexión para utilizar modelos Simulink de forma local, la principal ventaja de esta combinación, es la reutilización de modelos Simulink y el aprovechamiento de las capacidades gráficas de Ejs para aumentar la interactividad de éstos. La extensión desarrollada permite, con muy pequeños cambios en las simulaciones anteriores, la utilización de modelos Simulink en un servidor remoto. Con ello es posible ejecutar un laboratorio virtual, desarrollado con Ejs, desde un computador que no posea Matlab.

Palabras-clave— Educación en Automática, Laboratorio virtual, Interactivo, Laboratorio Web, Ejs, Simulink, Matlab.

1 Introducción

Las nuevas tecnologías de la información y la comunicación (TIC) están transformando todos los aspectos de la sociedad. El paradigma de la educación tradicional no es ajeno a ello, pasando de ser una etapa de preparación a un proceso de constante formación.

Un proceso continuo de enseñanza será más exitoso mientras más flexible sea respecto de la hora y el lugar en que se realice la instrucción. Desde esta perspectiva la educación a distancia se presenta como un pilar fundamental del nuevo paradigma, ya que es la acción de entregar conocimientos desde profesores a alumnos que pueden estar física o temporalmente separados (UNESCO).

Las TIC y sus novedosas características de comunicación (como Internet, la video conferencia o la mensajería instantánea) y de aprendizaje (como la simulación o la tele-presencia), presentan sin duda un enorme potencial para el desarrollo de la educación a distancia.

En la Ingeniería de Control, un laboratorio es un elemento central en el proceso de aprendizaje tradicional. En este caso, es claro que la interacción entre el alumno y un elemento de un laboratorio, como una planta, proporciona un conocimiento que es difícil de asimilar si no es mediante la utilización del mismo. Además la Ingeniería de Control requiere del estudio constante y profundo de una enorme cantidad de sistemas y modelos de diferentes tecnologías. Es por ello que la simulación, se convierte en un apoyo esencial para el aprendizaje de los mismos, mediante las diversas herramientas de análisis y diseño suministrados por la teoría.

Por otra parte, Internet proporciona los medios necesarios para el intercambio de información entre computadores ubicados en cualquier parte del planeta. Esta característica permite el desarrollo de simulaciones distribuidas, es decir, el cálculo computacional requerido puede ejecutarse en diferentes computadores conectados a Internet, consiguiendo con ello una versión remota de la simulación.

La Ingeniería de Control, debe entonces aprovechar las capacidades facilitadas por las herramientas de simulación e Internet, para reproducir y soportar las principales acciones causa-efecto entre el estudiante y un modelo de la planta, mediante un laboratorio virtual interactivo, ya sea en un formato local o remoto. En los últimos años se han producido interesantes resultados de investigación en este campo (Aktan B., *et al.* 1996; Gillet D. *et al.* 2005).

La interactividad es un aspecto fundamental cuando se diseñan laboratorios virtuales para ser usados con propósitos pedagógicos en el campo de la Ingeniería de Control. Es la interactividad y la calidad de respuesta del sistema, las que permiten al estudiante explorar diferentes configuraciones cuando se intenta adquirir no sólo la teoría, si no además los aspectos estratégicos e intuitivos de los modelos analizados (Dormido S. 2004).

Más aún, la interactividad debería permitir al usuario visualizar, en tiempo de ejecución (*on the fly*), la evolución de los diferentes aspectos y respuestas del sistema, ante cualquier cambio introducido en los parámetros. Esta observación inmediata,

del cambio gradual del sistema como respuesta a la interacción del usuario, es lo que realmente ayuda a los estudiantes a obtener un útil y práctico acercamiento a los fundamentos del control de sistemas (Sánchez J. *et al.* 2005)

Matlab/Simulink (Mathworks), es un paquete de software clásico que proporciona facilidades para la construcción de modelos de forma gráfica mediante diagramas de bloques, por lo que se ha convertido en una herramienta ampliamente utilizado en la industria y la enseñanza de gran parte de las ingenierías. Sin embargo, la interactividad proporcionada por los modelos Simulink, dista bastante del tipo de interactividad que se ha mencionado anteriormente.

Es en este aspecto donde Ejs (Esquembre F. 2005) puede resultar muy útil. Ejs es una herramienta software gratuita diseñada para el desarrollo de laboratorios virtuales interactivos. Este paquete proporciona un mecanismo propio para la descripción de sistemas y modelos dinámicos propios de la Ingeniería de Control. Además, Ejs provee de una enorme cantidad de elementos visuales parametrizables e interactivos, que permiten la rápida construcción de la vista de un laboratorio virtual.

La integración de modelos Simulink, en los laboratorios virtuales desarrollados en Ejs, ha sido parte de un trabajo constante y fructífero (Sánchez J. *et al.* 2005, Dormido S. *et al.* 2005). Los resultados obtenidos permiten, de forma muy sencilla al usuario, agregar a un modelo Simulink un alto nivel de interactividad.

Jim (*Java Internet Matlab*), es un paquete escrito en Java orientado a extender las capacidades de Ejs para manipular modelos Simulink. La extensión permite al usuario de Ejs, ejecutar la simulación sin la necesidad de contar con Matlab/Simulink en su equipo, ya que Ejs establece un enlace de red con un equipo remoto, soportado por Jim, que sí posee Matlab/Simulink.

La estructura del presente artículo es la siguiente: La sección 2 describe de forma breve el entorno Ejs. Posteriormente, en la sección 3, se resumen los pasos requeridos para utilizar los modelos Simulink en Ejs. En el apartado 4, se describe el servidor Jim. A continuación en la sección 5 se presenta un ejemplo de conexión remota. Finalmente se analizarán las principales conclusiones y futuros trabajos.

2 Ejs: Easy Java Simulations

Ejs es una herramienta de software gratuita diseñada para la creación de simulaciones interactivas en Java (Sánchez J. *et al* 2004, Esquembre F. 2005). El usuario al que está dirigida Ejs son estudiantes, profesores e investigadores de ciencias e ingeniería, que poseen un conocimiento básico de programación de computadores, pero que no disponen de una gran cantidad de tiempo para crear una simulación gráfica con un elevado grado de interactividad.

Las simulaciones en Ejs son estructuradas en dos partes, el *Modelo* y la *Vista*. En el modelo se describe el comportamiento del sistema mediante variables y código Java o ecuaciones diferenciales ordinarias. Mientras que la vista provee el aspecto gráfico o visual de la simulación. La interfaz de usuario de Ejs se presenta en la Figura 1.

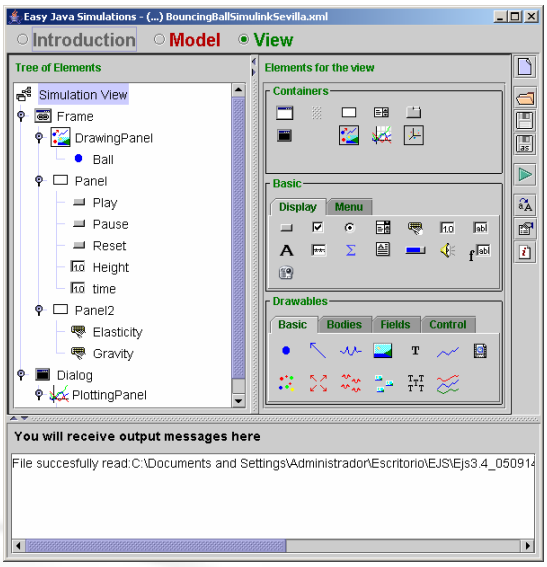

Figura 1. Interfaz de usuario de Ejs para la creación de la vista de una simulación. La estructura de árbol a la izquierda corresponde a la vista de la Figura 6.

Ejs se puede utilizar de forma independiente para crear simulaciones interactivas, ya sea como aplicación o como applet, sin embargo, también es posible utilizar Ejs para agregar interactividad a modelos Simulink (Dormido S. *et al.* 2005).

3 Easy Java Simulations y Simulink

La comunicación entre Ejs y Matlab/Simulink se realiza mediante la librería JMatlink (Müller S. 2005), que se basa en la Interfaz Nativa de Java y la librería Engine en lenguaje C proporcionada por Matlab. En la Figura 2 se muestra el esquema anterior.

El procedimiento para la conexión de un modelo Simulink y Ejs es simple y se resume en los siguientes cuatro pasos:

- Seleccionar (o modificar) el modelo Simulink.
- Crear y conectar variables Ejs con el modelo.
- Controlar la ejecución del modelo.
- Definir la vista y la interactividad con el usuario.

Al seleccionar el modelo Simulink, se debe considerar la potencialidad interactiva que éste posee. En general se recomienda que las variables que serán controladas u observadas por el usuario sean señales explícitas de entrada, valores de parámetros o salida de algún bloque en el modelo. Esto permitirá realizar la conexión con Ejs de forma sencilla y directa.

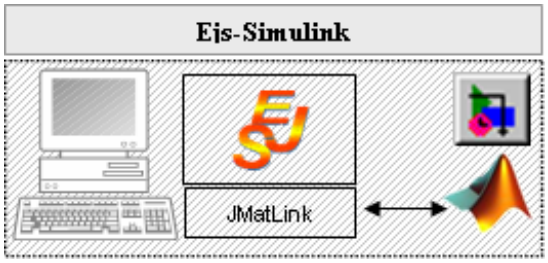

Figura 2. Conexión Ejs y Matlab/Simulink

Respecto a lo anterior, si se requiere aumentar la interactividad de un modelo Simulink para utilizarlo con Ejs, es posible que sea necesario modificar ligeramente la estructura del mismo de acuerdo con las recomendaciones mencionadas.

En la Figura 3 se observa el modelo seleccionado, en este caso corresponde al que proporciona Simulink para simular una pelota que rebota (aunque este modelo no es del campo de la Ingeniería del Control el procedimiento con un modelo que sí lo fuese sería análogo). Obsérvese que en este caso no se requiere una modificación del modelo ya que las variables de interés (gravedad, elasticidad, posición y velocidad) están ya dadas de forma explícita.

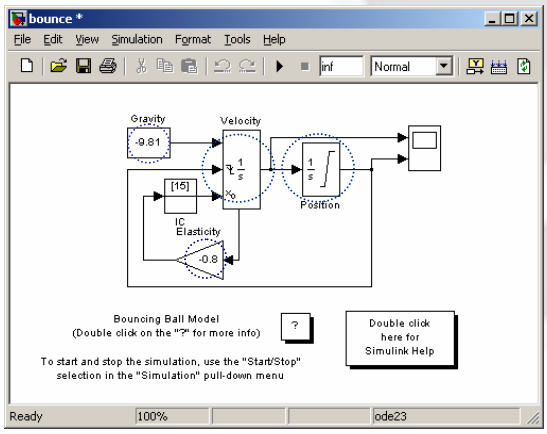

Figura 3. Modelo Simulink seleccionado. Las variables de interés están rodeadas por un círculo.

En la parte *Modelo* de Ejs, el usuario debe especificar en la *Tabla de Variable*s los nombres de las variables, sus dimensiones, el tipo, y si está conectada o no a una señal de un bloque del modelo Simulink. Para realizar la conexión, Ejs proporciona al usuario un cuadro de dialogo en donde se listan los bloques y parámetros disponibles del modelo. Ejs también permite realizar la conexión utilizando el modelo Simulink. En la Figura 4 es posible observar la indicación a Ejs del modelo seleccionado (*bounce.mdl*) y las variables conectadas al modelo.

Para que Ejs controle la ejecución sincronizada del modelo Simulink, basta especificar en la subsección *Evolución*, dentro de la sección *Modelo*, el siguiente método proporcionado por Ejs:

_external.step (int pasos);

Donde *pasos* es una variable entera.

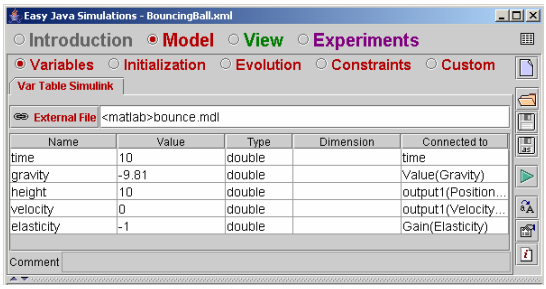

Figura 4. Interfaz de Ejs en donde se observa el modelo seleccionado y las variables de interés conectadas.

El control de la simulación está sincronizado, debido a que en cada paso de evolución se actualizan las variables de entrada en el modelo de acuerdo con sus valores en Ejs. En primer lugar se ejecuta un paso de integración en Simulink. A continuación se obtienen los valores de las variables de salida del modelo y finalmente se actualizan los valores de la vista en Ejs. Estas etapas se ejecutan repetitivamente para que la simulación evolucione.

La variable *pasos* indica a Ejs cada cuantos pasos de integración del modelo Simulink se actualizarán los valores de las variables de Ejs conectadas al modelo. En general el valor de pasos será igual a 1, aunque puede ser mayor, si se desea que la evolución de la simulación sea más rápida.

Finalmente, el cuarto paso consiste en la definición de la vista, y por tanto de la interactividad, ya que los elementos visuales, están íntimamente relacionados con la capacidad interactiva de la simulación. Para crear la vista, el diseñador debe utilizar los elementos gráficos disponibles proporcionados por Ejs y que se observan en el lado derecho de la Figura 1. Cada elemento de la vista proporciona al diseñador un conjunto de parámetros de configuración, como los que se observan en la Figura 5, para el elemento que representa la pelota.

En este caso las variables de interés como posición y velocidad están asociadas con el estado de la pelota, mientras que la elasticidad y gravedad pueden manipularse mediante dos deslizadores tal como se aprecia en la Figura 6.

Además obsérvese, en la misma imagen, la existencia de un conjunto de botones que permiten controlar la ejecución de la simulación. También puede distinguirse una ventana más pequeña al lado derecho, que presenta la evolución de la altura de la pelota respecto al tiempo.

Figura 5. Propiedades del elemento de la vista que representa la pelota. El texto sombreado corresponde al método *_external.resetIC()*.

Un aspecto importante para la interactividad es la posibilidad de reiniciar la simulación en un estado diferente, en este caso, por ejemplo que la pelota comience desde una altura definida por el usuario final. Para hacer esto Ejs proporciona al diseñador un método que modifica y reinicia a un valor arbitrario la salida de un bloque integrador del modelo Simulink. El método suministrado por Ejs es el siguiente:

_external.resetIC();

Con lo anterior, el usuario final podrá arrastrar la pelota a una altura diferente, y al soltarla, la simulación comenzará a ejecutarse nuevamente pero, en un nuevo estado. En la Figura 5 se observa, en texto sombreado, la incorporación del método anterior en las propiedades de la pelota.

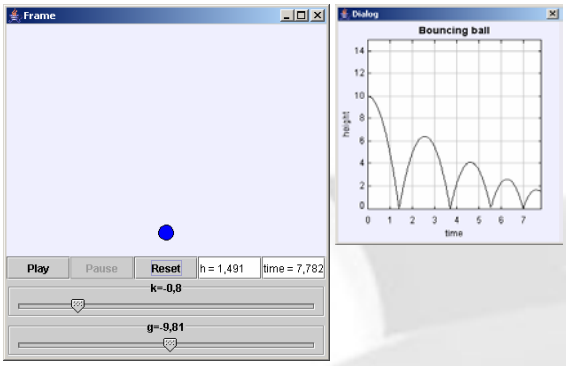

Figura 6. Interfaz gráfica de la simulación creada por Ejs.

Una descripción más detallada del procedimiento de conexión puede encontrarse en (Dormido S. *et al.* 2005).

4 Jim: Java Internet Matlab

Para extender las capacidades de conexión de Ejs y los modelos Simulink, se decidió desarrollar Jim, un paquete software que permite conectar un modelo Simulink remoto con una aplicación de Ejs.

Es importante destacar que Jim es una aplicación escrita en Java que se conecta con una versión de Matlab local, y no con la utilización de Matlab Web-Server.

Al igual que Ejs, la librería JMatlink se utiliza para la conexión entre Matlab y Jim.

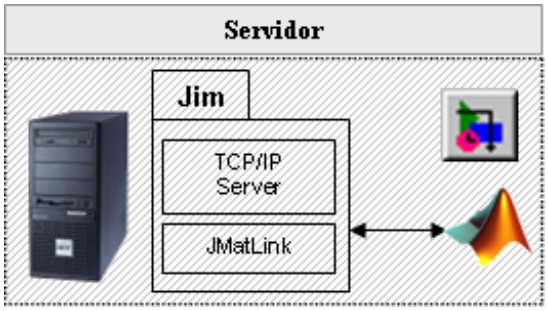

Figura 7. Servidor Java Internet Matlab

La comunicación entre Jim y Ejs es del tipo cliente/servidor y se resuelve mediante la utilización de sockets TCP/IP. El lado del servidor está soportado por Jim mientras que Ejs corresponde al lado del cliente. La Figura 7 presenta un esquema del servidor.

A continuación se describe brevemente la interfaz de usuario y las opciones proporcionadas por el servidor Jim.

3.1 Interfaz de usuario de Jim

En la Figura 8 se presenta la interfaz de usuario de Jim. En la interfaz se distinguen las siguientes opciones disponibles para la comunicación entre Jim, Ejs y Simulink:

- *Puerto de servicio*: Número del puerto en el que se establecerá la comunicación entre Ejs y Jim.
- *Tamaño del buffer de entrada*: Límite máximo del buffer de entrada disponible para el enlace TCP/IP.
- *Tamaño del buffer de salida*: Límite máximo del buffer de salida disponible para el enlace TCP/IP.
- *Directorio de trabajo*: Carpeta donde se alojarán los modelos Simulink propios del usuario remoto.
- *Máximo número de sesiones*: Límite superior de sesiones de Matlab a ejecutar en el servidor.
- *Permitir funciones S.O.*: Esta opción habilita al usuario remoto a utilizar funciones de Matlab (como DOS o el operador !) que afectan al sistema operativo.
- *Permitir archivos externos*: Permite al usuario remoto utilizar un modelo de Simulink propio.
- *Archivo de bitácora*: Si esta habilitado se guardará en un archivo información de las acciones realizadas por los usuarios externos.

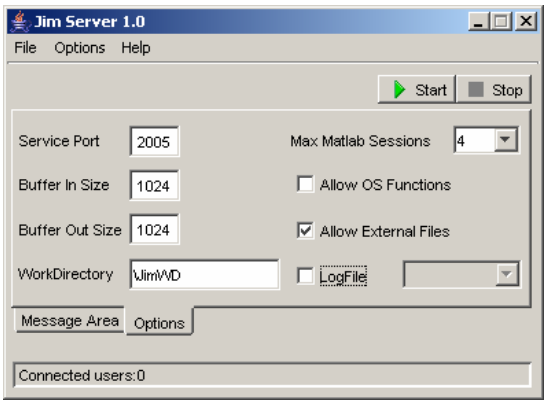

Figura 8. Interfaz de usuario de Jim.

Jim puede ser detenido en cualquier momento por el administrador del servidor remoto para realizar una re-configuración, sin embargo esto obligará a reiniciar la conexión con Jim al usuario remoto.

3.2 Protocolos de comunicación de Jim

Debido a los retardos inherentes en las comunicaciones por red, se decidió desarrollar una forma de realizar el enlace entre Jim y Matlab diferente a la utilizada entre Ejs y Matlab.

La comunicación original es de tipo **síncrona**, mientras que la desarrollada para la comunicación a través de la red se denomina **asíncrona**.

Como se describió en la sección 3, en la comunicación síncrona, cada paso de evolución de la simulación, tanto en la vista como en el modelo Simulink, está controlada directamente por Ejs. Es decir, el enlace es bidireccional en todo paso de evolución. Esto permite observar en la simulación el estado actual del modelo Simulink. La elección de este tipo de comunicación hará que la ejecución de la simulación en forma remota sea similar a la local, pero con los retardos propios de la red.

Para atenuar los retardos introducidos por la red, la versión asíncrona no mantiene una coordinación constante entre el modelo Simulink y la vista de la simulación. Este "descuido" se basa en la *idea clave* de que el usuario no interactúa con la simulación en todo momento, es decir, el usuario no introduce cambios en los parámetros de la simulación en cada instante, porque si se realiza uno, requerirá obviamente de un cierto tiempo para observar la respuesta del sistema antes de volver a hacer otro.

Debido a lo anterior, la comunicación entre la simulación y Simulink será unidireccional en la mayoría de los pasos de evolución, presentando una bidireccionalidad sólo en los instantes en que el usuario interactúa con la simulación. En el caso unidireccional, Jim enviará a Ejs los valores de las variables de salida del modelo, mientras que Ejs enviará a Jim los valores de las variables de entrada del modelo en el caso de que el usuario realice una interacción con la simulación.

5 Ejs, Jim y Simulink

En la Figura 9 se presenta el esquema de conexión entre Ejs, Jim y Simulink, obsérvese que el cálculo se realiza en el lado del servidor, mientras que el lado del cliente puede ser una aplicación o un applet creado por Ejs.

Uno de los aspectos considerados para el uso de un modelo Simulink remoto, es minimizar los cambios requeridos por una simulación existente que utiliza la versión local. Esto se ha logrado en forma absoluta en el caso del enlace síncrono, mientras que en el asíncrono sólo se requieren realizar un cambio más.

El único cambio requerido por una simulación local existente, en un enlace síncrono, es que el usuario especifique solamente la dirección IP y el puerto del servidor remoto. Obsérvese dicha modificación en la Figura 10, que se realiza con la simulación creada en la sección 3.

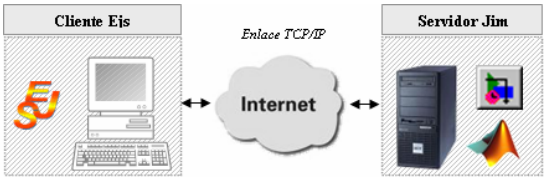

Figura 9. Conexión entre Ejs, Jim y Matlab

Una vez especificado el enlace remoto, no es necesario realizar ninguna modificación más a la simulación. Incluso, si el modelo Simulink no se encuentra en el servidor remoto, Ejs se encarga de enviárselo cuando sea necesario.

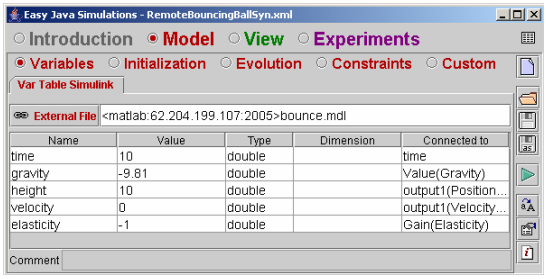

Figura 10. Establecimiento de un enlace remoto síncrono.

Para establecer un enlace asíncrono, entre una simulación local existente, y un servidor remoto, se debe cambiar la palabra clave *matlab* por *matlabAS*, seguida por la dirección IP y el puerto, por lo que en la Figura 10 se debería escribir "*<matlabAS:62.204....*" en lugar de "*<matlab:62.204.....*".

Otra modificación necesaria para establecer el enlace sincrónico, es utilizar el siguiente método para informar a Ejs que se han realizado cambios en las variables de entrada del modelo Simulink:

_external.synchronize();

En el caso del modelo de la pelota que rebota presentado en la sección 3, el método anterior debe utilizarse para detectar la interacción del usuario con los deslizadores tal como se observa en la Figura 11. En este caso el texto sombreado corresponde al método anterior, y es definido en una de las propiedades del elemento de la vista que controla la elasticidad.

Respecto de las propiedades del elemento de la vista que representa la pelota no se requiere ninguna modificación, porque Ejs lo hace implícitamente cuando se ejecuta el método *_external.resetIC()* al cambiar la altura de la pelota.

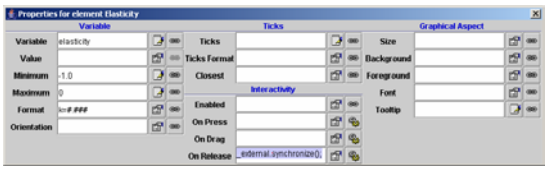

Figura 11. Propiedades del elemento de la vista que controla la elasticidad de la pelota que rebota. El texto sombreado corresponde al método _external.synchronize();

6 Conclusiones

El paradigma educacional esta en un proceso de cambio profundo, debido principalmente a la irrupción de las tecnologías de la información. La educación a distancia se beneficia del progreso de las nuevas tecnologías, debido a que aminoran las dificultades creadas a partir de la separación física o temporal entre alumnos, profesores y material de estudio.

El desarrollo de laboratorios virtuales y remotos está enfocado en este esfuerzo, ya que proporciona al usuario o alumno un conjunto de herramientas que simulan la interacción que tendría al utilizar un sistema existente. Este hecho es especialmente relevante en la Ingeniería de Control, donde la experimentación con una planta real, proporciona un aprendizaje difícil de obtener sin la utilización de la misma.

Ejs es una herramienta que presenta enormes posibilidades para la creación de laboratorios virtuales, ya que entrega al usuario un conjunto de elementos visuales interactivos, que se integran fácilmente en un modelo que, por ejemplo, podría ser descrito mediante ecuaciones diferenciales.

También se ha comentado, como Ejs incorpora de forma simple, en el diseño de un laboratorio virtual, un sistema descrito como un modelo en Simulink. Para ello el usuario sólo debe realizar unos cuantos pasos para obtener una simulación con una riqueza visual y gráfica verdaderamente interactiva.

El aspecto central de este trabajo, consistió en describir la extensión incorporada a Ejs que permite a un usuario ejecutar una simulación que utiliza un modelo Simulink, sin el requisito de tener instalado Matlab/Simulink en su equipo, debido a que Ejs realiza una conexión a través de la red con un computador externo, soportado por el servidor Jim, que si posee Matlab/Simulink.

Como se observó los cambios requeridos para ejecutar de forma remota una simulación local, son realmente mínimos, en cualquiera de las dos formas (síncrona o asíncrona) proporcionadas por Ejs y Jim.

El desarrollo de ésta nueva característica en Ejs, abre posibilidades para el diseño de laboratorios virtuales y/o remotos en computadores que no posean Matlab/Simulink.

Esta extensión, además permite la distribución del cálculo requerido por la simulación, por lo que incluso, en una red con un ancho de banda elevado, es posible observar que la versión remota de la simulación se ejecuta de forma más rápida que la versión local.

Agradecimientos

Los autores quieren agradecer la financiación de este trabajo a la CICYT en el marco del proyecto DPI2004-01804.

Referencias

UNESCO, www.unesco.org/education/educprog/lwf/doc/portfolio/definitions.htm

Aktan B., Bohus C., Crowl L., Shor M., (1996) "Distance Learning Applied to Control Engineering Laboratories", IEEE Transactions on Education, vol. 39, nº 3, Agosto, pág. 320-326.

Guillet D., Nguyen A., Rekik Y. (2005). Collaborative Web-Based Experimentation in Flexible Enginnering Education, IEEE Transactions on Education, vol. 48, nº 4.

Dormido, S. (2004) "Control learning: Present and future" IFAC Annual Control Reviews, vol. 28, pp 115-136.

Sánchez J., Dormido S., Esquembre F. (2005) "The learning of control concepts using interactive tools". Computer Applications in Engineering Education , vol. 13, nº 1, pp 84-98.

The MathWorks Inc., "Matlab", www.mathworks.com

Esquembre F. " Easy Java Simulations", http://fem.um.es/Ejs

Dormido S., Esquembre F., Farias G., Sánchez J. (2005). Adding interactivity to existing Simulink models using Easy Java Simulations, *44th IEEE* CDC/ECC, Sevilla (Spain).

Sánchez J., Dormido S., Pastor R., Esquembre F., (2004)"Interactive learning of control concepts using Easy Java Simulations", Plenary Lecture, IFAC Workshop Internet Based Control Education IBCE'04, Grenoble (France), september.

Esquembre F., (2005) "Creación de Simulaciones Interactivas en Java", Pearson Educación S.A., Madrid.

Müller S. (2005), www.held-mueller.de/JMatLink/