UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Tecnura

http://revistas.udistrital.edu.co/ojs/index.php/Tecnura/issue/view/725

DOI: http://dx.doi.org/10.14483/udistrital.jour.tecnura.2015.4.a08

ESTUDIO DE CASO

Herramienta de emulación de sistemas dinámicos a través de internet

Emulation tool of dynamic systems via internet

Daniel Ruiz Olaya[1], Edinson Franco Mejía[2]

Fecha de recepción: 29 de septiembre de 2014 **Fecha de aceptación:** 24 de agosto de 2015

Como citar: Ruiz Olaya, D., & Franco Mejía, E. (2015). Herramienta de emulación de sistemas dinámicos a través de internet. Revista Tecnura, 19(46), 103-113. doi:10.14483/udistrital.jour.tecnura.2015.4.a08

Resumen

Los laboratorios de experimentación para los cursos de formación superior en sistemas de control pueden llegar a ser costosos, ya sea en su adquisición, operación o mantenimiento. Un recurso alternativo para disponer plataformas de experimentación son los laboratorios remotos; sin embargo, no siempre es posible obtener sistemas complejos. Una solución a este problema son los laboratorios de emulación remotos. En este artículo se describe el desarrollo de una aplicación web para la emulación de sistemas dinámicos utilizando una herramienta software de prototipado rápido de control de libre distribución basada en Linux/RTAI. La aplicación está enfocada especialmente a la experimentación con sistemas dinámicos que no están disponibles fácilmente en un laboratorio, donde el modelo es configurado por el usuario. Se presenta el diseño de la interfaz de usuario y el motor de procesamiento de los datos enviados. Se verificaron en el servidor de aplicación los tiempos de latencia del sistema operativo en tiempo real y la capacidad del sistema para reproducir señales similares a las de un sistema real a partir de un modelo emulado. El modelo de un evaporador fue utilizado como ejemplo para probar la funcionalidad de la aplicación. Una de las ventajas de la aplicación es la metodología de trabajo, que está basada en el desarrollo de bloques en Scicos, esto permite al usuario reutilizar esos parámetros y el código que implementó para construir un bloque sobre el Toolbox Scicos en el entorno Linux/RTAI/ScicosLab; además, se requiere solamente un navegador web y la máquina virtual de Java.

Palabras clave: emulación de sistemas dinámicos, herramienta de experimentación, RTAI, simuladores en tiempo real.

Abstract

The experimentation laboratories for the studies of control system courses can become expensive, either in its acquisition, operation or maintenance. An alternative resource have been the remote laboratories. However, not always is possible to get complex systems. A solution to this matter are the remote emulation laboratories. In this paper describes the development of a Web application for the emulation of dynamic systems using a free-distribution software tool of rapid control prototyping based on Linux/RTAI. This application is focused especially for the experimentation with dynamic systems that are not

[1] Ingeniero electrónico, Magíster en Ingeniería con énfasis en Automática de la Universidad del Valle. Cali, Colombia. Contacto: daniel.ruiz.olaya@correounivalle.edu.co

[2] Ingeniero electricista, Magíster en Ingeniería con Énfasis en Automática, Doctor en Ingeniería. Docente e investigador de la Universidad del Valle. Cali, Colombia. Contacto: edinson.franco@correounivalle.edu.co

available easily in a laboratory where the model have been configured by the user. The design of the front-end and the back-end are presented. The latency times of the real-time operating system and the ability of the system to reproduce similar signals to a real system from an emulated model were verified. An example, to test the functionality of the application the model of an evaporator was used. One of the advantages of the application is the work methodology which is based on the development of blocks in Scicos. This allows the user to reuse those parameters and the code that was implemented to build a block on the Scicos toolbox with the Linux/RTAI/ScicosLab environment. Moreover, only a web-browser and the Java Virtual Machine are required.
Keywords: experimentation tool, emulation of dynamic systems, real-time simulators, RTAI.

INTRODUCCIÓN

Las herramientas de experimentación en la enseñanza de los sistemas de control son un factor importante para el aprendizaje de los estudiantes de ingenierías. Aprovechando las ventajas que actualmente ofrecen las tecnologías informáticas (TIC) se han desarrollado y empleado diferentes recursos como los laboratorios remotos y los laboratorios virtuales como una alternativa a los prototipos de plantas físicas; las TIC permiten que los nuevos recursos desarrollados beneficien la interactividad y promuevan el autoaprendizaje en los estudiantes (Dormido, 2004). No siempre es posible reproducir un escenario de un problema real de control, por costos o acceso a la herramienta; una alternativa para acercar los estudiantes a aspectos prácticos de los sistemas de control son los experimentos basados en emulación. En este contexto, una emulación es un "proceso computacional de un modelo que simula la operación de un sistema real o propuesto" (Macias, Guridi & Ortiz, 2007). Por tanto, se busca que el modelo a emular sea lo más similar posible al sistema real, esto se logra considerando modelos detallados del sistema que consideran componentes no lineales, la mínima cantidad de simplificaciones, los tiempos y las señales tal como se presentan en la realidad.

La implementación de esta aplicación debe realizarse dentro de un sistema operativo de tiempo real con la capacidad de conectarse con un dispositivo externo para realizar tareas de "Hardware-in-the-Loop"; para realizar esto se utilizó una herramienta de prototipado rápido de control (RCP, por sus siglas en inglés). El uso de RCP está enfocado principalmente a realizar tareas de implementación de estrategias de control sobre una planta física (Franco, Maya & Ramos, 2004; Franco, Jaramillo, Maya & Ramos, 2000).

A nivel comercial existen diferentes alternativas para probar en tiempo real una estrategia de control, tales como dSPACE®, el software LabView® de National Instruments, xPCTarget® de MathWorks, QUARC® y RCP Toolkit® de Quanser, OPAL-RT, entre otros; estos sistemas se caracterizan por ser muy flexibles, con altas velocidades de procesamiento y, en general, con costos de licencia y hardware elevados.

Una alternativa que ha estado en constante desarrollo ha sido un sistema conformado por distintos paquetes software de libre distribución como RTAI, Rtai-Lab y ScicosLab; esta forma de experimentación usualmente se realiza localmente. En los últimos años se reportan en la literatura desarrollos basados en RTAI/Rtai-Lab para construir laboratorios remotos (Basso, Romagnoli, & Innocenti, 2008; Bucher & Balemi, 2008; Santos, 2009) donde el controlador de la planta viene preestablecido y en algunos casos con la posibilidad de suministrar un diseño realizado por el usuario a través de algún software adicional como Matlab® o Scilab®.

Janik y Zakova (2012) utilizaron este entorno RTAI/Rtai-Lab/Scilab para construir un laboratorio

de control remoto para una planta de levitación magnética, realizaron modificaciones a la metodología de implementación de los experimentos dando la posibilidad al usuario de diseñar su controlador en línea; el diseño y el proceso de generación del ejecutable requiere de diferentes interacciones por parte del usuario, para ello se emplea un editor gráfico como Scicos; además, presentan una solución para automatizar ese proceso de compilación a través de un *script* que carga en la ejecución no-interactiva de SciLab (ejecución del programa sin GUI) los diferentes módulos de la herramienta Scicos, RTAI y el macro de generación de código C (el cual fue modificado para que se ejecute de forma automática). En la plantilla desarrollada por Janik y Zakova (2012) se encuentra fija la estructura de bloques de entrada y salidas en Scicos, a la cual el usuario introduce la estructura de controlador.

En este trabajo se extiende lo reportado en diferentes investigaciones (Franco et al., 2004; Basso et al., 2004; Bucher & Balemi, 2008; Santos, 2009; Janik & Zakova, 2012) para permitir que los usuarios (estudiantes y profesores) implementen modelos de sistemas dinámicos *online* utilizando la RCP basada en RTAI/Rtai-Lab/ScicosLab, y en una etapa posterior realicen los respectivos análisis o diseñen estrategias para controlar la planta a partir de especificaciones de diseño.

Para el desarrollo de la herramienta no se requiere modificar el macro generador de código C, y en la generación del ejecutable en tiempo real la aplicación no requiere ejecutar ScicosLab para compilar el proyecto. Este enfoque permite diversificar los contextos y aplicaciones en donde los sistemas de control son utilizados (Fernández, Ramírez, & Orozco, 2010); por tanto, se ha establecido este tipo de experimentación como una herramienta alternativa de apoyo a la formación con metodología de aprendizaje basado en proyectos (ApP), la cual se está empleando en los cursos de sistemas de control en la Universidad del Valle (Fernández, Ramírez, & Orozco, 2012).

Este documento está conformando por cinco secciones. En la sección dos se describen brevemente las herramientas empleadas, en la sección tres se presenta la metodología para el desarrollo de la interfaz de usuario de configuración y el motor de procesamiento en el lado del servidor, en la sección cuatro, los resultados de las pruebas realizadas y en la sección 5, las conclusiones.

HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO DE LA APLICACIÓN

En la implementación de esta aplicación se utilizaron lenguajes como HTML, CSS y JavaScript para el desarrollo de la interfaz gráfica de configuración, y para el procesamiento del formulario se utilizó el lenguaje PHP. La herramienta se integró a la Plataforma de Experimentación Remota en Ingeniería (s. f.).

En el servidor de aplicación se utilizó como plataforma el "RealTime Suite" (Guiggiani, Marta, Basso, Vassalli, & Difato, 2011). La plataforma está conformada por varias aplicaciones software, entre las cuales está la extensión que le proporciona las características de tiempo real duro al sistema operativo Linux, denominada *Real-Time Application Interface for Linux* (RTAI, 2014), versión 3.8.1; RTAI-LAB (Bucher, 2004) es una herramienta que se incorpora a las aplicaciones de diseño de sistemas de control asistido por computador (CACSD) adicionando bloques que generan código en tiempo real, además integra un generador de código que permite compilar y generar el ejecutable en tiempo real para el entorno Linux/RTAI. Dentro de las herramientas CACSD se utilizó Scicos/ScicosLab v4.4.1 (ScicosLab, 2014), un "toolbox" que permite diseñar en diagramas de bloques el algoritmo de simulación; también se utilizó la aplicación servidor RTAI-XML (RTAI-XML, 2014), que establece la comunicación remota (a través de la red TCP/IP) con el proceso en tiempo real desarrollado y un cliente, para el intercambio de datos y parámetros con el proceso.

METODOLOGÍA

Interfaz de la herramienta

El proceso que debe realizar un usuario para realizar una emulación en esta aplicación se divide en dos etapas: la primera consiste en realizar la configuración del experimento y la segunda, en la interacción con el experimento.

La interfaz de usuario de configuración se diseñó para que sea intuitiva, con una secuencia de navegación establecida y con bajos requerimientos de software, ya que solo requiere de un navegador web. En la Figura 1 se ilustra el flujo que debe seguir un usuario para configurar una simulación, que consiste en tres pasos secuenciales.

El primer paso consiste en definir la cantidad y tipo de entradas y salidas del modelo (Figura 2),

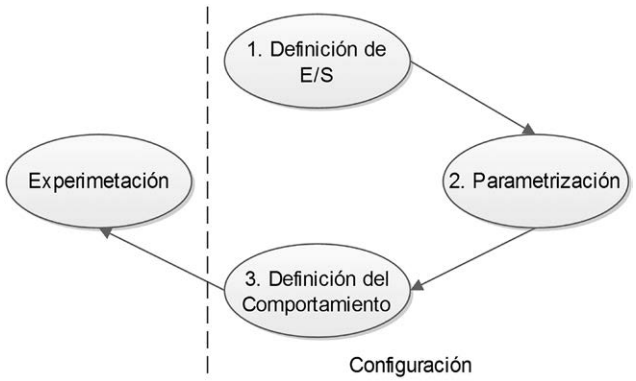

Figura 1. Flujo de experimentación en la aplicación

Fuente: elaboración propia.

el segundo, en definir los valores iniciales para el experimento (Figura 3), en el tercer paso, el usuario debe definir el comportamiento del modelo que desea emular a través de la representación en

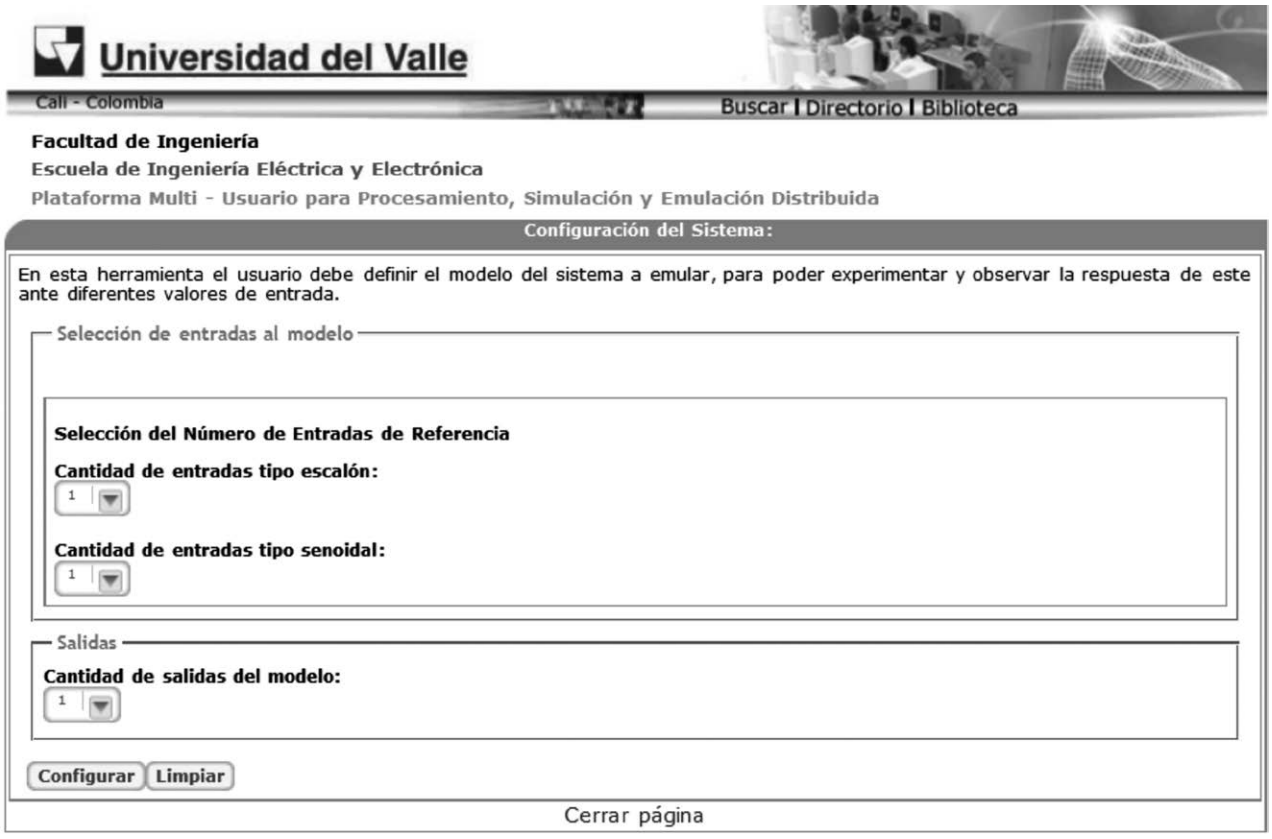

Figura 2. Interfaz para la selección de las entradas y salidas del modelo

Fuente: elaboración propia.

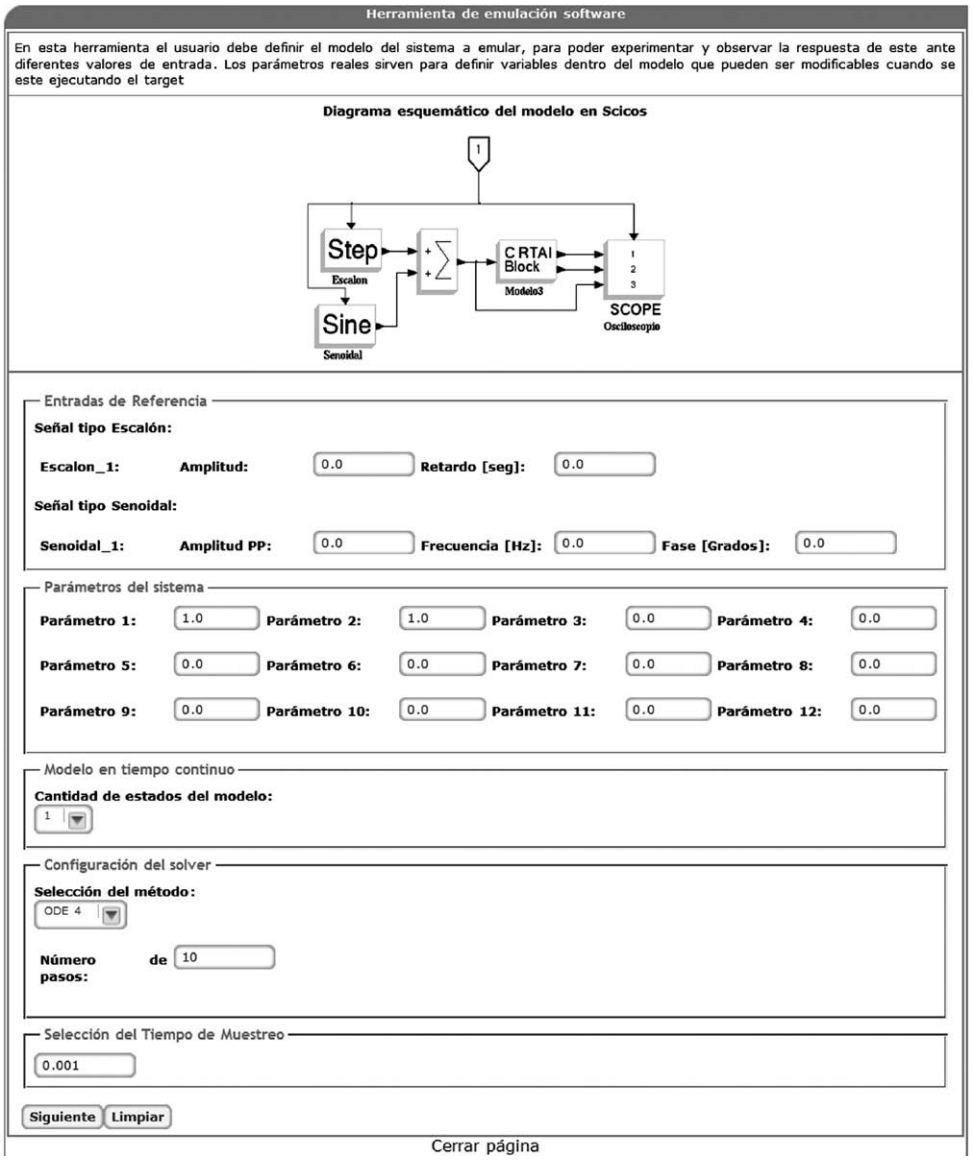

Figura 3. Interfaz para la parametrización del modelo

Fuente: elaboración propia.

variables de estados del proceso (Figura 4); al finalizar el tercer paso de configuración, el usuario debe enviar los datos del formulario y el motor de la herramienta descrito al servidor para la respectiva compilación y generar el proceso en tiempo real. En caso de que ocurra un error en el proceso de configuración, el usuario no podrá realizar la experimentación hasta que los corrija y los vuelva a enviar; si no ocurre error alguno, el usuario podrá proceder a la etapa de experimentación; en cada etapa es programada una función JavaScript para verificar el correcto ingreso de los datos tanto en tipo como en sus valores.

Para la definición del comportamiento del modelo, que debe estar representado en variables de estado, el usuario debe tener conocimientos básicos en lenguaje C y en la función computacional de Scicos. Esta metodología es similar a cuando se

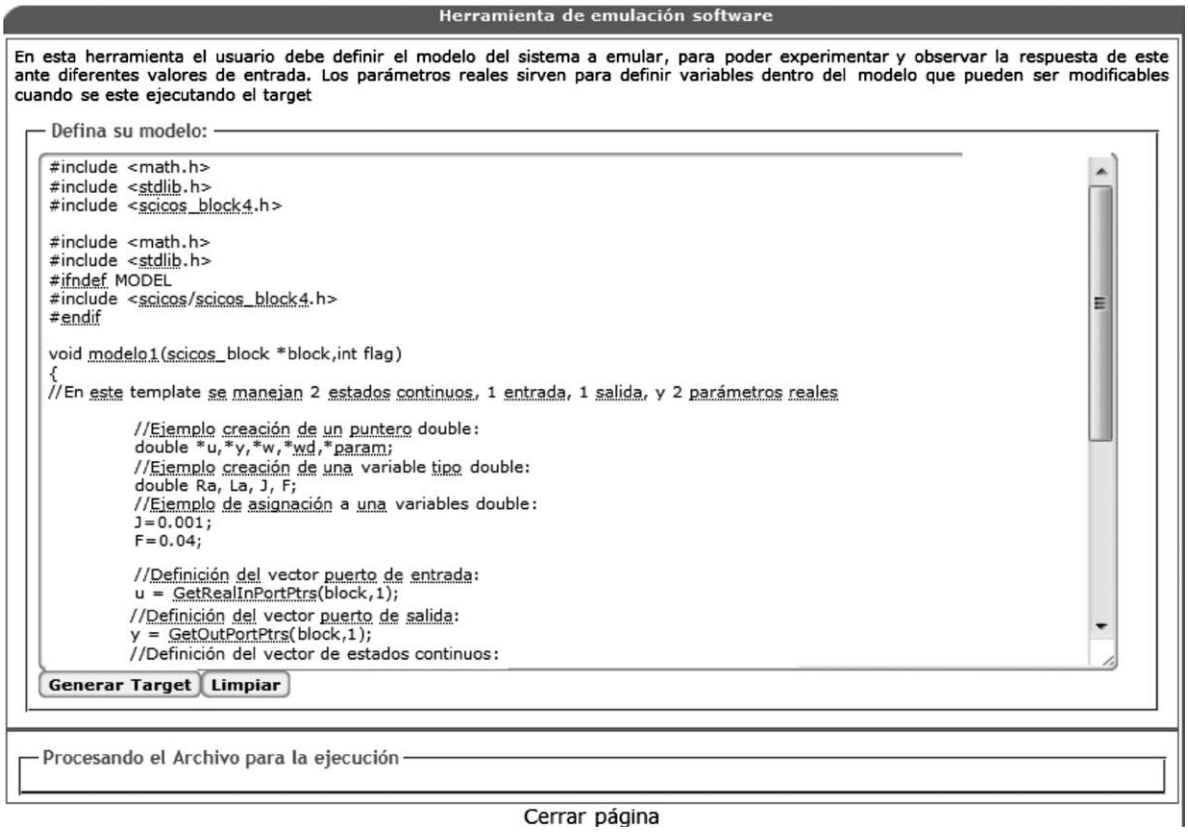

Figura 4. Interfaz para la implementación del comportamiento del modelo

Fuente: Elaboración propia.

desarrolla un nuevo bloque en Scicos, en donde se define la función de interfaz del bloque y la interfaz de comportamiento.

La interfaz de usuario para la interacción con el experimento es proporcionada por el proyecto jRtaiLab (RTAI-XML, 2014), el cual permite establecer una conexión con el servidor RTAI-XML, así como visualizar y modificar los parámetros del experimento en línea.

Motor de la Herramienta

La Figura 5 representa el proceso que se realiza en el lado servidor, el cual es implementado en un *script* que ejecuta de forma automática la compilación de los archivos para generar el ejecutable en tiempo real y la habilitación del experimento en caso de ser exitosa la compilación.

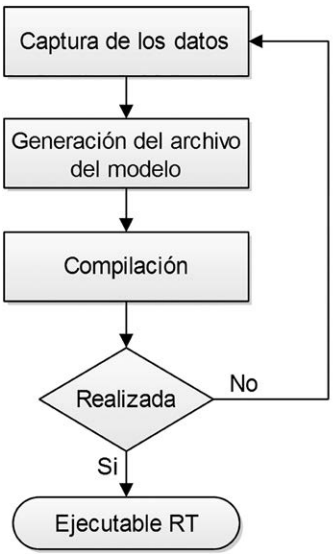

Figura 5. Flujo de tareas que se realiza en el servidor

Fuente: Elaboración propia.

En la captura de la información suministrada por el usuario a través del formulario descrito en la sección previa se usaron las funciones que proporciona PHP para acceder a los datos.

La creación del ejecutable en tiempo real está basada en la forma como el generador de código modificado presentado en Bucher (2004) realiza este trabajo.

En la Figura 6 se presenta el proceso de generación de un ejecutable en tiempo real utilizando la RCP descrita previamente. Esta herramienta funciona de forma local. La metodología de trabajo consiste en desarrollar el algoritmo como un diagrama de bloques en Scicos, y una vez terminado se seleccionan todos los bloques, a excepción del reloj, y se transforma en un superbloque. Finalizado el diseño, se utiliza el macro RTAICodegen.sci para compilar el superbloque y enlazar las respectivas librerías y funciones necesarias para crear el ejecutable. Como resultado de la compilación se crean varios archivos, representados como los bloques de color gris oscuro de la Figura 6, denominados "código", "CBlocks", "rtmain", "common" y "Makefile". El archivo "código" contiene la representación del algoritmo desarrollado; el archivo "CBlocks" contiene la función computacional para el bloque genérico que se utiliza para escribir el comportamiento del modelo a implementar; el archivo "rtmain" viene integrado a RTAI-Lab y realiza dos tareas principales: manejo de la tarea de tiempo real y comunicación con aplicaciones cliente para monitoreo y cambio de parámetros; el archivo "Makefile" contiene las instrucciones para compilar los archivos a través del compilador de código C de Linux y generar el ejecutable en tiempo real.

Los pasos anteriores son realizados de forma local a través de varias interacciones gráficas hechas por el usuario; por tanto, es necesario automatizar el proceso de creación del algoritmo y el proceso de compilación y generación del ejecutable en tiempo real. Dado que la herramienta tiene como objetivo la implementación de modelos matemáticos de plantas o procesos, se desarrolló en Scicos un modelo que sirve como base para posteriormente compilar de forma local el algoritmo y obtener los archivos generados por el macro generador de código. Estos archivos son utilizados por la aplicación web; con los datos proporcionados por el usuario se modifican los archivos "código" y "CBlocks" para actualizar las características del modelo. Para evitar ejecutar ScicosLab en el proceso de compilar el proyecto, se automatizó la compilación y la generación del ejecutable en tiempo real, basándose en el archivo "Makefile", el cual utiliza los archivos *.c antes mencionados y posteriormente se genera el ejecutable en tiempo real a partir de los archivos *.o que resultan de la compilación.

Por último, realizada la compilación, si es exitosa, queda disponible el ejecutable para ser utilizado a través de la interfaz de experimentación mencionada en la sección anterior.

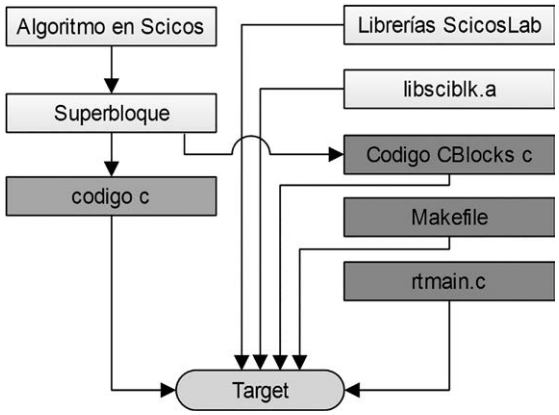

Figura 6. Proceso de generación de un ejecutable en tiempo real de forma local

Fuente: Elaboración propia

PRUEBAS Y RESULTADOS

En la validación de la herramienta de experimentación se realizaron tres tipos de pruebas. La primera prueba consistió en verificar en el servidor de aplicación los tiempos de respuesta del sistema operativo en tiempo real ante diversas interrupciones.

En la segunda prueba se verificó la capacidad del sistema para reproducir señales similares a las de un sistema real a partir de un modelo emulado. Finalmente, se verificó el funcionamiento de la aplicación a través de la emulación de un modelo matemático no lineal de un evaporador.

En la verificación del servidor de aplicación se utilizaron las pruebas que tiene disponible RTAI (en el espacio del kernel), las cuales permiten conocer los retardos de procesamiento, que son de tres tipos: "latency", "switches" y "Preempt".

La prueba de "latency" verifica el rendimiento general del sistema, mide la diferencia en tiempo entre el tiempo del interruptor en espera y el tiempo cuando una tarea es llamada por el planificador; el rendimiento en tiempo real está limitado por la latencia máxima que puede lograr, en este caso se obtuvo un tiempo aproximadamente de 10us sin ningún "overrun".

La prueba de "switches" proporciona información sobre la cantidad de tiempo máxima que RTAI necesita para deshabilitar las interrupciones; se obtuvo un tiempo máximo de 817 ns para la ejecución de 10 tareas; también se verificó que el tiempo de conmutación es menor que el tiempo de latencia máximo, una condición que es requerida por el sistema.

La prueba de "Preempt" es una utilidad de tensión que verifica los programadores en tiempo real bajo carga de procesamiento, este software combina la tarea de calibración de latencia con una tarea rápida y lenta para tener dos niveles de preferencia "preemption"; se obtuvo un retardo máximo de 13 us para esta prueba.

Para verificar la capacidad del emulador en la reproducción de señales similares a las de un sistema real se implementaron dos casos: el primero usando un servomotor del cual se disponían las señales tiempo real, mientras en el segundo caso se usó el modelo de un evaporador. El servomotor se seleccionó por ser ampliamente usado en formación en Control, además presenta no linealidades, su modelo es muy conocido, ha sido ampliamente validado y esta fácilmente disponible en el laboratorio; el comportamiento de la señal del sistema real y la producida por el modelo emulado fueron comparadas para comprobar la efectividad del emulador. Para el caso del servomotor, en la Figura 7 se presenta el comportamiento dinámico del sistema y la comparación de ambas señales, así como también el escalón de entrada con una variación de un 10% sobre el rango máximo de entrada del sistema; el error relativo medio (MRE) obtenido es de 1,63% en la comparación de las señales; la exactitud de la respuesta de la señal emulada es determinada por la exactitud del modelo.

Para el caso del modelo del evaporador, se definieron en la ecuación: (1) las señales de control, (2) las variables de estado, (3) las entradas de perturbación y (4) las señales de salida.

$$u^T = \begin{bmatrix} m_{vent} & m_{jent} \end{bmatrix} \tag{1}$$

$$x^T = \begin{bmatrix} h & C_{jsal} \end{bmatrix} \tag{2}$$

$$d^T = \begin{bmatrix} K_\lambda & C_{jent} \end{bmatrix} \tag{3}$$

$$\frac{dh}{dt} = \frac{m_{jent} - K_v\sqrt{h}\rho - K_\lambda m_{vent}}{\rho A} \tag{4}$$

Donde m_{vent} es el flujo másico de vapor de entrada, m_{jent} es el flujo másico de jugo de entrada, h es el nivel de jugo en los tubos de la calandria, C_{jsal} es la concentración del jugo de salida, K_λ es la constante de ganancia estática y C_{jent} es la concentración del jugo de entrada.

El modelo se basa en las ecuaciones de balance de masa, componente de masa y energía de Polo, Galiano, & Chica (2004) e Ipanaqué, De Keyser, Dutta, Oliden, & Manrique (2012), a partir de las cuales se obtuvieron las ecuaciones (5) y (6).

$$\frac{dh}{dt} = \frac{m_{jent} - K_v\sqrt{h}\rho - K_\lambda m_{vent}}{\rho A} \tag{5}$$

$$\frac{dC_{jsal}}{dt} = \frac{m_{jent}C_{jent} - (m_{jent} - K_\lambda m_{vent})C_{jsal}}{\rho A h} \tag{6}$$

Al implementar el modelo se definieron en la interfaz de selección (ver Figura 2) dos entradas

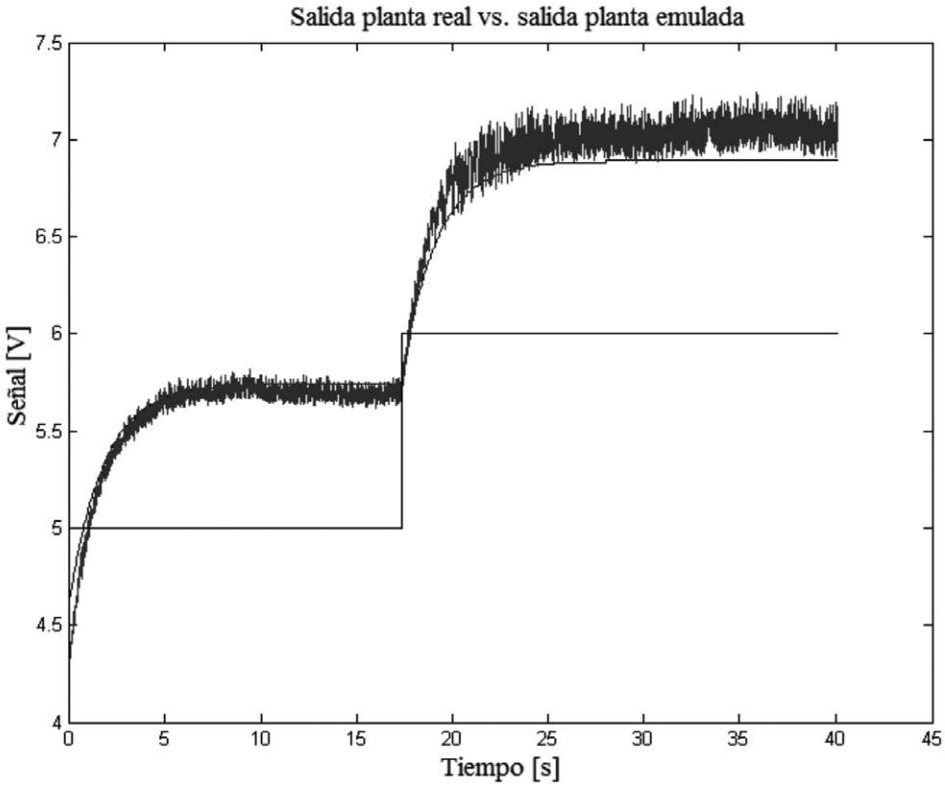

Figura 7. Comparación de la señal medida en la planta y la señal emulada

Fuente: elaboración propia.

tipo escalón [m_{vent} m_{jent}] y dos salidas [h C_{jsal}]. En la interfaz de parametrización (ver Figura 3) se establecieron los valores iniciales para los flujos másicos de entrada, se utilizaron dos parámetros para las variables de disturbio asignándole sus valores iniciales respectivos, y finalmente se seleccionaron dos estados; el resto de datos para esta interfaz se dejó por defecto. Con lo anterior se tiene que las variables de entrada y de perturbación se puedan modificar *online* y obtener la respuesta ante diversos valores sin necesidad de volver a compilar el modelo. En la interfaz de la Figura 4 se implementó el comportamiento del sistema, que está determinado por las ecuaciones (5) y (6) y la utilización de los parámetros de la interfaz de parametrización.

Las respuestas en lazo abierto ante una variación en la variable de entrada m_{jent} y m_{vent} se ilustran en la Figura 8 para nivel y la concentración de jugo de salida.

La simulación se realizó considerando una variación del 10% en el flujo de jugo de entrada a los 1,7 s manteniendo constante el flujo de vapor de entrada, igualmente a los 7,0 s se realizó una variación del flujo de vapor de entrada en un 10%, dejando constante el flujo de jugo de entrada. Se observa la interacción de las variables a controlar con respecto a las variables manipuladas: cuando aumenta el flujo de jugo de entrada, la concentración de jugo de salida disminuye de 21,1%°Brix a 20,7%°Brix y el nivel aumenta de 3,54 m a 4,0 m; y viceversa: para cuando el flujo de vapor aumenta, la concentración de jugo de salida en el evaporador aumenta de 20,7%°Brix a 21,3%°Brix y el nivel disminuye de 4,0 m a 3,81 m.

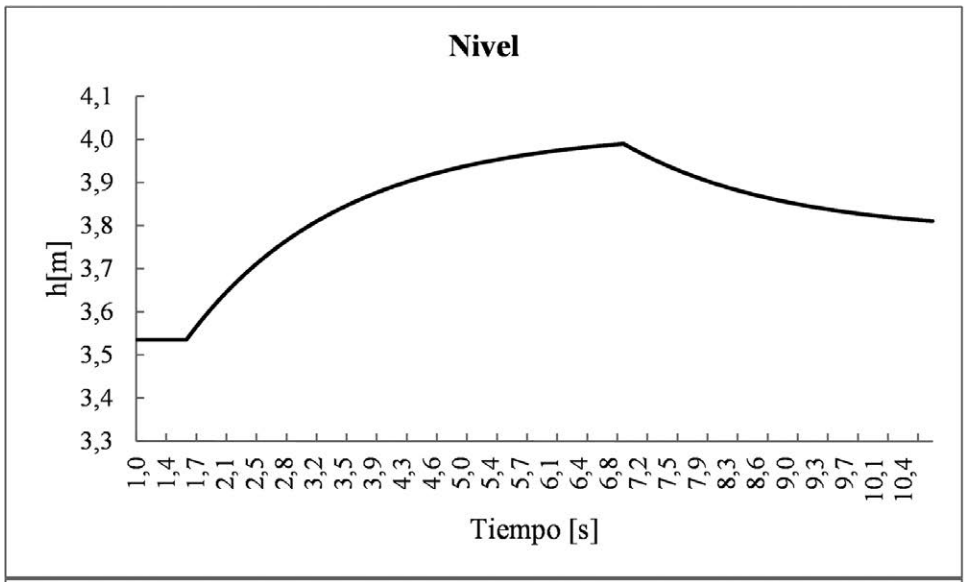

Figura 8. Respuesta del sistema ante una variación en el flujo másico de jugo de entrada y el flujo másico de vapor de entrada.

Fuente: Elaboración propia.

CONCLUSIONES

En este trabajo se presentó una aplicación desarrollada a partir de software libre que permite a los estudiantes e investigadores realizar simulaciones de modelos de sistemas dinámicos para ejecutarse en una plataforma en tiempo real a través de un navegador web y la máquina virtual de Java. El enfoque establecido para la herramienta es la interacción de los estudiantes con diferentes modelos o problemas de control, buscando acercar al estudiante a un contexto más cercano a la práctica.

Esta metodología basada en la creación de nuevos bloques permite fomentar habilidades en programación y manejo de herramientas de simulación de sistemas de control, igualmente permite al usuario reutilizar esos parámetros y código implementados para construir un bloque sobre la herramienta Scicos en el entorno Linux/RTAI/ScicosLab; además promueve el uso de software libre como una alternativa a las herramientas comerciales.

La forma de automatizar la generación del ejecutable en tiempo real en este trabajo evita modificar el macro generador de Código C y ejecutar ScicosLab para compilar el proyecto.

REFERENCIAS

Basso, M., Romagnoli, M., & Innocenti, G. (2008). A Distributed Remote Control Lab. *Proceedings of the 17th World Congress The International Federation of Automatic Control*. Seoul, Korea.

Bucher, R., & Balemi, S. (2008). CAN-bus based rapid control prototyping system for education laboratories. *Proccedings of the 17th World Congress The International Federation of Automatic Control Seoul*. Korea.

Bucher, R. (2004). Interfacing Linux RTAI with Scilab/Scicos. *Real Time Linux Workshop*. Singapore.

Bucher, R., & Dozio, L. (2003). CACSD under RTAI Linux with RTAI-LAB. *In Fifth Real-Time Linux Workshop*. Valencia, España.

Dormido, S. (2004). Control learning: Present and future. *Annu Rev Control*, 28, 115–136.

Fernández, L., Ramírez, J., & Orozco, M. (2012). Project-based learning approach for control system courses. *Sba: Controle & Automação Sociedade Brasileira de Automatica*, *23*(1), 94-107.

Fernández, L., Ramírez, J., & Orozco, M. (2010). Emulation and remote experimentation as support resources in a PBL approach for control systems. *Revista Facultad de Ingeniería*, 55, 194-202.

Franco, E., Jaramillo, A., Maya, A., & Ramos, C. (2000). Control Virtual Fuzzy en Tiempo Real con Scicos bajo Linux. *Congreso Latinoamericano de Control Automático y VI Congreso de la Asociación Colombiana de Automática.* Colombia.

Franco, E., Maya, A., & Ramos, C. (2004). Aplicaciones de Linux para Simulación y Control de Procesos Software GNU: Scilab y Scicos. *Energía y Computación*, *11*(2), 54-59.

Guiggiani, A., Marta, V., Basso, M., Vassalli, M., & Difato, F. (2011). Realtime Suite: a step-by-step introduction to the world of real-time signal acquisition and conditioning. *13th Real Time Linux Workshop.*

Ipanaqué, A., De Keyser, R., Dutta, A., Oliden, J., & Manrique, J. (2012). Control no lineal iterativo predictivo de evaporador en obtención de bio-etanol. *15th Latinamerican Control Conference.*

Janik, Z., & Zakova, K. (2012). Real-Time Experiments in Remote Laboratories Based on RTAI. *Interactive Collaborative Learning (ICL), 2012 15th International Conference on,* 26-28 Sept, 1-5, Villach. Doi: 10.1109/ICL.2012.6402075

jRtaiLab. *RTAI-XML*. Recuperado de http://www.rtaixml.net (Consultado el 25 de septiembre de 2014).

Macias, M., Guridi, E., & Ortiz, A. (2007). Extending the laboratory concept with computer emulations in automation. *Frontiers In Education Conference–Global Engineering: Knowledge Without Borders, Opportunities Without Passports,* pp. S3G-18, S3G-22.

Polo, M., Galiano, J., & Chica, J. (2004). Modelado, análisis y control de un evaporador de doble efecto. *XXV Jornadas de Automática.* Ciudad Real.

PERI. Recuperado de http://eieela.univalle.edu.co (Consultado el 16 de octubre de 2014).

RTAI. Recuperado de https://www.rtai.org. (Consultado el 16 de octubre de 2013).

RTAI-XML. Recuperado de http://www.rtaixml.net (Consultado el 16 de octubre de 2013).

Santos, D. (2009). *LABEXP-Laboratório de experimentação remota em tempo real.* (Dissertação Mestrado, Programa de Pós-Graduação em Engenharia Elétrica,–Universidade Federal do Pará, Instituto de Tecnologia). Belém.

ScicosLab. Obtenido de http://www.scicoslab.org (Consultado el 25 de septiembre de 2014).