DESIGN OF A MEASURING NODE TO REGULATE THE SPEED OF A DC MOTOR IN A NETWORKED CONTROL SYSTEM, BASED ON HARDWARE - SOFTWARE CODESIGN TECHNIQUES

DISEÑO DE UN NODO MEDIDOR PARA UN SISTEMA DE CONTROL EN RED QUE REGULA LA VELOCIDAD DE UN MOTOR DC, BASADO EN TÉCNICAS DE CODISEÑO HARDWARE – SOFTWARE

MSc. Diego Martínez C., Ing. Fernando Iván Arévalo, Ing. Carlos Fernando G.
PhD. Apolinar González Potes (*)

Grupo de Investigación de Sistemas de Telemando y Control Distribuido - GITCoD
Departamento de Automática y Electrónica, Universidad Autónoma de Occidente,
Santiago de Cali, Colombia
(*) **Facultad de Ingeniería Mecánica y Eléctrica**, Universidad de Colima, México
{dmartinez, fiarevalo, cfgiraldo}@uao.edu.co, apogon@ucol.mx

Abstract: This work shows the design of a measuring node that is part of a control system that regulates the speed of a motor DC in a scheme of control in network; the experimental results and of simulation allow to validate the made design. The characteristics of the levels that conform this system are considered in their totality during the phases of designing; it means, in a same representation are analyzed the retardations, the flow data and the functionality of the components of the same one, allowing to diminish the difference between the experimental results and the results of simulation. The procedure of used design is the result of integrating several techniques and methodologies to specify and to design realtime systems, which allow to use the advantages displayed in these techniques, such as the codesign hardware - software.

Resumen: En este trabajo se presenta el diseño de un nodo medidor que hace parte de un sistema de control que regula la velocidad de un motor DC en un esquema de control en red; los resultados experimentales y de simulación permiten validar el diseño realizado. Las características de los niveles que conforman este sistema son consideradas en su totalidad durante las fases de diseño; es decir, en una misma representación se analizan los retardos, el flujo de datos y la funcionalidad de los componentes del mismo, permitiendo disminuir la diferencia entre los resultados experimentales y los resultados de simulación. El procedimiento de diseño utilizado es el resultado de integrar varias técnicas y metodologías para especificar y diseñar sistemas de tiempo real, lo cual permite utilizar las ventajas presentadas en estas técnicas, tales como el codiseño hardware – software.

Keywords: Distributed Control Systems, Real Time, Embedded Systems, Hardware-Software Codesign, Colored Petri Nets.

1. INTRODUCCIÓN

Debido al aumento en la complejidad de los sistemas de control gran parte de las actividades se deben distribuir entre diferentes nodos, los cuales se comunican por medio de redes de comunicación. Adicionalmente, la implementación de sistemas distribuidos permite disminuir el impacto producido por las fallas en un componente del sistema y facilita las actividades de diagnóstico y mantenimiento del mismo.

De otro lado, al diseñar sistemas de control distribuido con dinámicas muy exigentes no siempre se obtiene una buena correspondencia entre los resultados experimentales y los de simulación, esto es debido al uso de modelos imprecisos para analizar y diseñar estos sistemas, métodos de validación poco elaborados y plataformas que no soportan los modelos empleados.

El análisis y diseño de sistemas de control distribuido es una disciplina relativamente nueva, por lo que en la literatura existen muy pocas publicaciones que hacen referencia a técnicas de modelado y simulación (Martínez and Velasco, 2004). Actualmente esta disciplina presenta una serie de desafíos, entre ellos se encuentran el desarrollo de herramientas, métodos y modelos para: soportar las etapas arquitecturales del diseño; facilitar la comunicación entre los diseñadores de diferentes disciplinas; y realizar validaciones adecuadas.

Sin embargo, durante los últimos años los investigadores han mostrado un gran interés por desarrollar nuevos métodos de especificación, algunos de las cuales se presentan en (Burns, et al., 1995; Martínez and Rodríguez, 2004; Powel, 2000).

En este artículo se presenta el diseño de un nodo medidor que hace parte de un sistema de control que regula la velocidad de un motor DC en un esquema de control en red, para lo cual se ha utilizado un procedimiento de diseño de sistemas de control distribuido de tiempo real (Martínez and Velasco, 2004), el cual permite resolver algunas de las dificultades presentes en el diseño de estos sistemas. En este caso, se emplean componentes para las diferentes fases de diseño, con el fin de modelar, analizar el comportamiento y validar el diseño de estos sistemas, de tal forma que los resultados obtenidos cumplan con las especificaciones propuestas en los sistemas de control.

El artículo está organizado de la siguiente forma. En la sección 2 se presenta el procedimiento de diseño utilizado. En la sección 3 se presenta el diseño de un nodo medidor implementado en un sistema de control en red que regula la velocidad de un motor DC. En la sección 4 se presentan las conclusiones y el trabajo futuro.

2. PROCEDIMIENTO DE DISEÑO DE SISTEMAS DE CONTROL DISTRIBUIDO

El procedimiento de diseño utilizado en este trabajo se presenta en (Martínez and Velasco, 2004). La arquitectura propuesta para implementar el sistema de control se presenta en la Figura 1, en ella se pueden observar tres niveles:

1. *Interfase de usuario*. Las aplicaciones en este nivel se implementan en una computadora personal (PC).
2. *Red de comunicaciones*. Para interconectar los nodos del sistema se seleccionó el protocolo CAN.
3. *Módulos de bajo nivel.* En este nivel se implementan las funciones de medición, y cálculo de acción de control y actuación. Para su diseño se utilizan Microprocesadores y FPGAs, y se integran conceptos de la metodología de codiseño hardware – software.

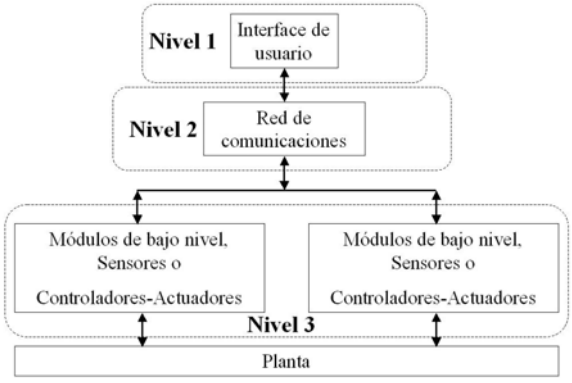

Fig. 1. Arquitectura del sistema de control distribuido

2.1. Procedimiento para diseñar sistemas de control distribuido de tiempo real

El procedimiento de diseño utilizado en este trabajo se presenta en la Figura 2 y se detalla en (Martínez, 2004).

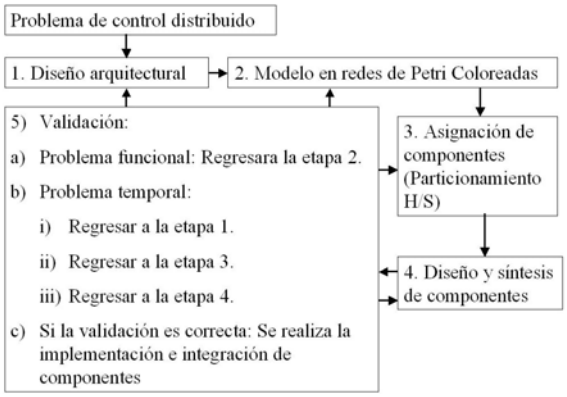

Fig. 2. Etapas del procedimiento de diseño

Las etapas del procedimiento son:

1. *Diseño arquitectural.* En esta fase se desarrolla un modelo arquitectural del sistema, en el cual no se definen las plataformas sobre las cuales serán implementados cada uno de los componentes.
2. *Diseño del modelo en redes de Petri Coloreadas (CPN).* Durante esta etapa se traslada el modelo arquitectural del sistema a un modelo en CPN, lo cual permite una validación funcional del sistema.
3. *Asignación de componentes*. En esta fase se definen las plataformas sobre las cuales serán implementados cada uno de los componentes.
4. *Diseño y síntesis de componentes*.
5. *Validación del sistema.* Durante esta fase se realiza la medición de los tiempos de cómputo de cada una de las actividades y se caracteriza el modelo obtenido en CPN, con el cual se realiza la validación dinámica del sistema.
   En función de los resultados obtenidos de la validación se realiza una de las siguientes acciones:
   a. Si se detecta un problema funcional se regresa a la etapa 2.
   b. Si se detecta un problema temporal, dependiendo del desfase que se presente se ha de regresar a: etapa 1, etapa 3, o etapa 4.
   c. Si los resultados son satisfactorios, se realiza la implementación e integración del sistema.

Los criterios de selección de los componentes para el diseño y análisis, la representación de estos en CPN, y la selección de los lenguajes utilizados en cada fase, se detalla en (Martínez, 2004).

3. DISEÑO DEL NODO MEDIDOR PARA REGULAR LA VELOCIDAD DE UN MOTOR DC EN UN SISTEMA DE CONTROL EN RED

El diseño de un sistema de control en red posee dos etapas, en la primera se realiza una selección del algoritmo de control, se calculan los coeficientes del mismo, y se seleccionan el periodo de muestro y los retardos máximos admitidos en el lazo de realimentación. Durante la segunda etapa se realiza el diseño e implementación de la aplicación.

En (Martínez and Arévalo, 2004) se presentan los análisis realizados para encontrar el algoritmo de control que regula la velocidad del motor DC, así como el periodo de muestreo y los retardos máximos que puede admitir este sistema antes de presentar una degradación considerable de su respuesta con respecto al desempeño esperado.

En 3.1 se presentan las especificaciones con las cuales se diseño el nodo medidor y en 3.2 se presentan los resultados de las etapas del diseño de este nodo.

3.1. Especificaciones

Las especificaciones funcionales y temporales del nodo medidor son:

- Especificaciones Funcionales
- Funciones del módulo medidor. Este módulo realiza el sensado de la velocidad y el acondicionamiento de esta señal. Sus funciones son:

- Medir y filtrar la velocidad del motor, y transmitir este dato a través de la red de comunicaciones CAN.

❑ Especificaciones Temporales

De acuerdo con los resultados presentados en (Martínez and Arévalo, 2004) se realizó la siguiente selección para el periodo de muestro y retardo máximo admitido para toda la tarea de control, la cual involucra: medición, filtrado, transmisión y recepción de información, cálculo de acción de control y acción sobre el sistema.

- El período de muestreo de la velocidad es de 8 mseg.
- El retardo máximo para la tarea de control es de un periodo de muestreo.

3.2. Etapas del diseño del nodo medidor

Una vez se finaliza la etapa de captura de especificaciones, se continúa con el diseño e implementación de la aplicación.

En esta sección se presentan los resultados de las etapas que se realizaron para diseñar e implementar el nodo medidor.

Diseño arquitectural del nodo medidor. El nivel inicial del diseño arquitectural del nodo medidor hace parte del diseño completo del sistema que regula la velocidad del motor DC (Martínez and Arévalo, 2004), y se presenta en la Figura 3. En esta figura se puede apreciar que este nodo está compuesto de tres componentes: Lectura, Filtrado y Driver de red (Controlador CAN).

Modelo del sistema en redes de Petri Coloreadas. La Figura 4 muestra el nivel de mayor jerarquía del modelo en redes de Petri Coloreadas del nodo medidor. Este modelo permite describir completamente la solución que se propuso para cumplir con las especificaciones propuestas, incluyendo todos los aspectos relacionados con su implementación.

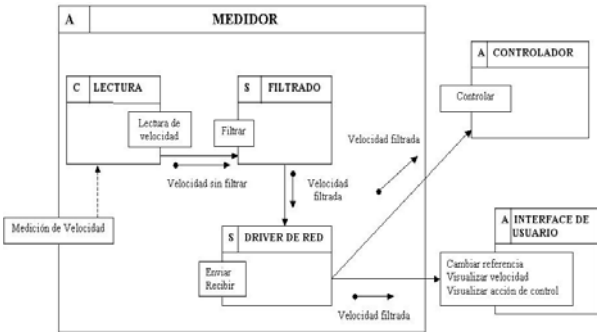

Fig. 3. Nivel inicial del diseño arquitectural del nodo medidor.

La validación dinámica de esta representación, permite confirmar el cumplimiento de las especificaciones del sistema, considerando el comportamiento de todos los componentes utilizados en la implementación. Adicionalmente, esta representación permite el intercambio de información entre grupos de diseño que trabajan a diferentes niveles para la solución.

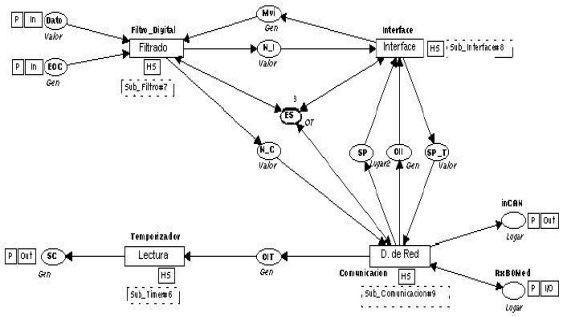

Fig. 4. Modelo en CPN del nodo medidor.

Implementación del nodo medidor. En la selección de las plataformas de implementación intervienen factores como: recursos, costo, disponibilidad, conocimiento, etc. En (Martínez and Velasco, 2004) se propone una implementación total en FPGA utilizando módulos IP de procesadores para la implementación del software, lo cual ofrece una gran flexibilidad. En este trabajo la implementación de los componentes software se realizó sobre el microcontrolador 89C52 y los componentes hardware se implementaron sobre la FPGA Flex 10K70. En el diseño del módulo medidor se aplicó la técnica de Codiseño Hardware – Software, con lo cual se logró obtener una adecuada implementación que cumple con las especificaciones propuestas.

La Figura 5 muestra los resultados de la simulación del componente del nodo medidor que maneja la comunicación a través de la red CAN.

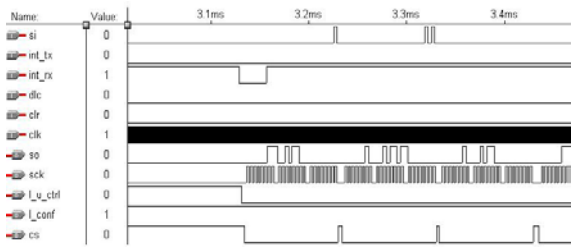

Fig 5. Resultados de la simulación del componente que maneja la comunicación a través de la red CAN.

Codiseño del módulo medidor. Los parámetros considerados para evaluar el costo de realizar una asignación de componentes en hardware y software fueron los siguientes:

- *Ap,* cantidad de área requerida para la implementación de componentes en hardware.
- *Tp,* latencia de cómputo.
- *Mp,* cantidad de memoria requerida para la implementación de componentes en software.

En (López and López, 2003) se presenta una función de costo general y eficiente que permite evaluar la calidad de la solución obtenida, la cual tiene la forma:

$$F(P)=\sum_{0}^{i} K_i \frac{C_i(P)}{C_i}+\sum_{0}^{i} K_{Ci} F_C\left(C_i, C_i(P)\right) \quad (1)$$

Donde *Ci*, es la restricción de diseño aplicado al iesimo parámetro, *Ci(P)* es la solución obtenida de una nueva partición *P* que es usada como parámetro de normalización, *Kci* es el peso del factor para los términos de corrección y *Fc* es la función de corrección.

La función de corrección utilizada en este trabajo es una función de penalidad, la cual no contribuye a la función de costo cuando la solución esta dentro del espacio permitido de búsqueda. La expresión de la función de penalidad es:

$$F_C\left(C_i, C_i(P)\right)=r^2\left[C_i, C_i(P)\right] \quad (2)$$

Donde, la función $r\left[C_i, C_i(P)\right]$ corresponde a:

$$r\left(C_i, C_i(P)\right)=máx\left\{0, \frac{\left[C_i(P)-C_i\right]}{C_i}\right\} \quad (3)$$

Los parámetros utilizados para realizar el análisis de costo fueron:

- Restricciones: A = 913 Celdas Lógicas, T = 2894 µs y M = 798 Bytes
- Coeficientes de las restricciones y de la función de corrección: K_A=0.3; K_T =0.4; K_M =0.3; $K_{CA}=K_{CT}=K_{CM}$ =150

En este trabajo, para encontrar la mejor partición se utilizó el algoritmo de *Fiduccia-Mattheyses*.
Se utilizó la siguiente abreviación para cada tarea:

- Lectura (L)
- Filtrado (F)
- Controlador CAN (D_C)

Después de tres iteraciones se obtuvieron los resultados que se presentan en la Tabla 1.

Tabla 1. Resultados de la tercera iteración del algoritmo *Fiduccia-Mattheyses* aplicado al diseño del nodo medidor.

Tarea Movida	Partición Resultante	Costo de la Partición Resultante	Se mejora el costo?
L	[ϕ,(L,F,D_C)]	0.4018	SI

De los resultados obtenidos en la Tabla 1 se concluyó que la mejor asignación corresponde a la implementación en hardware de todos los componentes que constituyen este nodo.

3.4. Resultados experimentales

La validación total del sistema se realizó utilizando la herramienta *DesignCPN*, con la cual se logró realizar un análisis dinámico del sistema, lo cual permitió observar el efecto producido por la red de comunicaciones y la ejecución de los algoritmos en hardware y software.

La Figura 6 muestra los resultados obtenidos experimentalmente, en ella se puede apreciar la correspondencia entre los resultados experimentales y los de simulación.

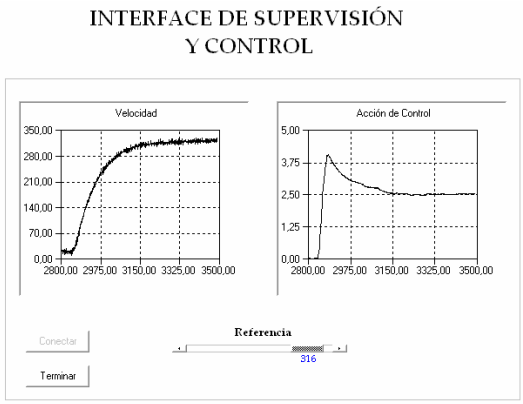

Fig 6. Resultados experimentales del sistema de control de velocidad.

4. CONCLUSIONES

1. En este trabajo se presenta el diseño de un nodo medidor para un sistema de control en red que regula la velocidad de un motor DC, para lo cual se utilizó un procedimiento de diseño de sistemas de control distribuido de tiempo real, el cual integra varias técnicas y metodologías para especificar y diseñar estos sistemas, permitiendo utilizar las ventajas presentes en las mismas.

2. En este diseño fue posible integrar aspectos relacionados con la técnica de codiseño hardware – software.

5. TRABAJO FUTURO

El trabajo futuro será orientado a:

- Modelar nuevos componentes para el diseño arquitectural. Así como también, mejorar la notación de los mismos con el fin de considerar algunos aspectos relacionados con sistemas de diagnóstico, detección y corrección de fallos.
- Analizar y desarrollar nuevas plataformas de implementación para mejorar el desempeño de estos sistemas, y verificar su incidencia sobre el desempeño de los algoritmos de control.
- Modelar las interfaces mecánicas del sistema e incluir estos modelos dentro de los análisis del mismo.

REFERENCIAS

Burns, A. and Wellings, A. J. (1995). HRT-HOOD a Structured Design Method for Hard Real-Time Systems.

Albertos, P. and Crespo, A. (2000). Sistemas de Control en tiempo real: Avances en el diseño y realización. Corporación Universitaria Autónoma de Occidente.

El-Khoury, J. and Törngren, M. (2001). Towards a Toolset for Architectural Design of Distributed Real-Time Control Systems, In: *Proc. of Real-Time Systems Symposium (RTSS), 3-6 December, London*, pp. 267-276.

López, M. and López, J. C. (2003). On the Hardware-Software Partitioning Problem: System Modeling and Partitioning Techniques. In: *ACM Transactions on Design Automation of Electronic System*, **Vol 8**, No 3, pp 269-297.

Martínez, D. Arévalo, F. I. and Giraldo, C. (2004). Diseño de un Sistema de Control en Red para Regular la Velocidad de un Motor DC. In: *XI Taller IBERCHIP.*

Martínez, D. Rodríguez, M. González, A. (2004). Diseño de un Sistema de Monitoreo y Control de Tiempo Real Basado en Técnicas de Codiseño Hardware/Software. In: *XVII Congreso Nacional y III Congreso Internacional de Informática y Computación de la ANIEI, Tepic, Nayarit, México, 20 al 22 de octubre de 2004.*

Martínez, D. Velasco, J. Pinedo, C. (2004). Procedimiento de diseño de sistemas de control distribuido de tiempo real. In: *Simpósium Iberoamericano de Educación, Cibernética e Informática: SIECI 2004, Orlando, Florida ~ EE.UU, 21 al 25 de julio del 2004.*

Powel, B. (2000). *Real-Time UML. Second Edition*, Addison-Wesley, Massachusetts, EE.U.U.

Martínez, D. (2004). Procedimiento de diseño de sistemas de control distribuido de tiempo real. In: *Tesis de maestría en automática. Universidad del Valle. Santiago de Cali, Colombia.*

Walsh, G. C. Beldiman, O. and Bushnell, L. (1999) Asymptotic Behavior of Networked Control Systems. In: *Submitted to Control Applications Conference.*

Zhang, W. (2001). Stability analysis of networked control systems.