# Construcción de Observadores de Secuencias para Sistemas de Eventos Discretos

***D. Ulises Carranza Sahagún,*C. Acosta Lúa, **R. Campos Rodríguez, **M. Alcaraz Mejía**

*Departamento de Ciencias Básicas, Centro Universitario de la Ciénega, Universidad de Guadalajara, Av. Universidad No. 1115, Col. Lindavista, Ocotlán, Jalisco, México, diegouli@cuci.udg.mx.
**Departamento de Electrónica, Sistemas e Informática, ITESO-Universidad Jesuita de Guadalajara, Periférico Sur No. 8585, Col. ITESO, Tlaquepaque, Jalisco, México, rcampos@iteso.mx, mildreth@iteso.mx.

**Resumen**
Este trabajo presenta el diseño e implementación de algoritmos para la construcción de observadores de secuencias para sistemas de eventos discretos. El modelo del sistema se captura como una Red de Petri, mientras que la implementación del es quema del observador se realiza en Simulink. Los algoritmos permiten verificar la propiedad de observabilidad, a la vez que construyen la matriz de detección de secuencias sobre la que se basa elfuncionamientodel observador.

**Palabras Clave:** Observador de Secuencias, Sistemas de Eventos Discretos, Simulación, Matlab/Simulink.

## I. Introducción

Los sistemas de eventos discretos (SED) se encuentran prácticamente en todos los aspectos de la vida cotidiana. Por ejemplo en los electrodomésticos, aparatos de comunicaciones, sistemas operativos de computadora, control de semáforos en las avenidas, sistemas de manufactura avanzada, entre otros. Los SED se han vuelto cada vez más complejos y sofisticados, de tal manera que para hacer más eficiente su uso, así como para implementar esquemas de tolerancia a fallas y recuperación de errores, se han propuesto varios esquemas de control[1]. Muchos de estos esquemas utilizan información de retroalimentación para calcular una adecuada ley de control. Por ejemplo, para el caso del Problema de Regulación de la Salida[2], el regulador requiere conocerel estado del SED cuya salida está controlando. En otros esquemas como el Control Supervisor, el controlador requiere retroalimentar estados o cadenas para una adecuada supervisión [3]. Algunas ocasiones es posible retroalimentar al controlador con la información completa del estado del sistema. Sin embargo, existen situaciones donde es complicado obtenerlo por completo, debido a limitaciones tecnológicas o a la gran cantidad de estados en el SED, lo que vuelve económicamente inviable medirlo directamente. Esto dificulta la aplicación de los esquemas de control a los SED, e incluso imposibilita su implementación en algunos casos.

Para enfrentar este problema se utiliza el concepto de un Observador, el cual es un algoritmo matemático responsable dereconstruir el estado completo del sistema que se requiere controlar. Su salida, que es una aproximación del estado del sistema, se utilizacomo retroalimentación para el controlador.La noción de observabilidad está relacionada con la posibilidad de inferir el estado actual e inicial de un SED en un número finito de eventos [4]. Dependiente del esquema de control utilizado, los observadores se pueden modelar como Autómatas, Redes de Petri o Sistemas Aditivos Vectoriales.

En particular, este trabajo se centra en el diseño de algoritmos para la verificación de la observabilidad y la construcción observadores para SED modelados como Redes de Petri. Los observadores construidos se pueden simular en Matlab/Simulink para verificar propiedades de desempeño, como la constante de convergencia

del observador, previo a su implementación física.

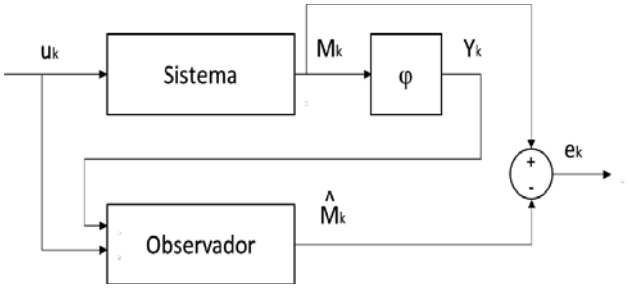


**Fig. 1. Esquema del observador.**

## II. Desarrollo

Las Redes de Petri (RP) son una metodologíaformal de modelado de SED. Sus principalescaracterísticas incluyen una representación gráfica muy conveniente y sus fuertes bases matemáticasque permiten el análisis formal de propiedades. Esto facilita por una parte, modelar sistemas de manera muy intuitiva, y por otra, obtenerinformación sobre el comportamiento dinámicodel sistema modelado mediante análisis formal y simulaciones por computadora.

De manera sucinta, una Red de Petri se definecomo un par $(G, M_0)$, donde $G$ representa laestructura de la red con una 4-tupla formada por: $P$ un conjunto de lugares; $T$ un conjunto detransiciones; $I: P \times T \rightarrow Z^+$ una función quedefine la incidencia de entrada a las transiciones;y $O: P \times T \rightarrow Z^+$ una función que define laincidencia de salida de las transiciones. Dada lanaturaleza de estas funciones $I, O$ se representanmediante matrices del tipo $I_{|P|\times|T|}$ y $O_{|P|\times|T|}$.

Típicamente, los lugares se representan concírculos, las transiciones con rectángulos, lafunción de entrada con arcos que van delos lugares a lastransiciones, mientras que la función de salida con arcos que van de lastransiciones alos lugares. Las marcas serepresentan con pequeños círculos negros dentrode los lugares.El denominado "juego de marcas" determina la dinámica de la red. La función de marcado $M(k) \rightarrow \mathbb{N}^m$representa el estado o marcado de la red, y determina el número de marcas en cada uno de sus lugaresal tiempo $k$, donde $m$ representa el número de lugares de la red. El vector $M_0$, conocido como marcado inicial, es un vector especial porque representa el número demarcas que reside en cada lugar inicialmente. Para sistemas puramente discretos, el contador $k$ se incrementa cada que ocurre un evento, el cual es asíncrono en el tiempo, y se considera como una especie de "base de tiempo" aunque no en el sentido de la teoría de control clásico.Una subclase de RP ampliamente estudiada esconocida como Máquinas de Estado (ME). En este tipo de redes las transiciones tienen sólo un lugar de entrada y un lugar de salida. La simplicidad y características de las máquinas de estado permiten hacer un análisis estructural de la propiedades como la observabilidad, controlabilidad, vivacidad, entre otras.El estudio de las principales propiedades de las RP se encuentra en [5].Existen varios enfoques que tratan el problema de la observabilidad en el marco de las RP[6], Informalmente, una RP $(Q, M_0)$ es *observable* si existe un número entero finito $k < \infty$ tal que para cada evolución de la red, mayor o igual a $k$, la información de las señales de entrada y salida de la red, y la estructura de $(Q, M_0)$ son suficientes para determinar de forma exclusiva el marcado inicial $M_0$ y el marcado actual $M_k$. El valor numérico de $k$ es conocido como la *constante convergencia de observabilidad* de$(Q, M_0)$.

En la Fig. 1 se muestra el esquema del observador retroalimentado mediante la salida $y_k$ del sistema. El observador se implementa como una copia del sistema al que además se le proporciona su entrada $u_k$, como se ilustra. El observador produce estimaciones $\widehat{M}_k$ las cuales se restan al estado $M_k$ del sistema para determinar el sistema del error $e_k = M_k - \widehat{M}_k$. El objetivo es que $e_k$ llegue a cero conforme $k$ tiende a infinito, es decir, $\lim_{k\rightarrow\infty} e_k \rightarrow 0$. El esquema de la Fig. 1 se puede implementar en Matlab/Simulink u Octave/SicLab para realizar simulaciones previo a la implementación de los algoritmos del observador.

ISBN: 978-607-28-0126-4

## III. Implementación

Se han elaborado una serie de algoritmos basados en [4], [6], [7], que permiten verificar la propiedad de observabilidad de una RP, principalmente para Maquinas de Estados.

*Una máquina de Estados (ME) es una RP*$(\boldsymbol{Q}, \boldsymbol{M_0})$*, con la propiedad de que para cada* $\boldsymbol{t_i} \in \boldsymbol{T}$ *se cumple que* $|\blacksquare \boldsymbol{t_i}| = \boldsymbol{1} = |\boldsymbol{t_i} \blacksquare|$. *Una ME es conectada si para cada* $\boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{P} \cup \boldsymbol{T}$ *existe una trayectoria desde* $u$ *hasta* $v$ *y de* $v$ *hacia* $u$. *Una ME es segura si es conectada y además* $|M_0| = 1$

Una ME segura exhibe una estructura que permite la caracterización dela observabilidad mediante la Secuencia-Detectabilidad. La Secuencia-Detectabilidad permite verificar elVector-Disparo-Detectabilidad y elMarcado-Detectabilidad, suficientes para verificar la propiedad de la observabilidad en ME seguras y conectadas[7].El primer algoritmo implementado permite la construcción de la *Matriz de Detección de Eventos.*

*Algoritmo 1: Matriz de Detección de Eventos.*Sea$(\boldsymbol{Q}, \boldsymbol{M_0})$una ME segura y conectada. La *Matriz de Detección de Eventos* $\boldsymbol{E(Q) = E[n-1, n-1]}$ etiquetada con columnas como $\mathbf{t_1}, \ldots, \mathbf{t_{n-1}}$ y filas como $\mathbf{t_2}, \ldots, \mathbf{t_n}$. Donde las entradas de la matriz están dadas de acuerdo al diagrama de flujo de la Fig. 2. El algoritmo recibe como entrada la matriz de incidencia $C$ de la RP y la matriz $\varphi$ que representa la función de salida e indica en qué lugares están los sensores en la red. El algoritmo itera por todos los pares de transiciones y marca con el símbolo de vacío aquellos pares que distinguen.Por el contrario, marca con el producto cruz de sus transiciones de salida a aquellos pares que se confunde.

La salida de este algoritmo es la Matriz de Detección de Eventos (Matriz de Evento-Detectabilidad), denotada como $E(Q)$.

El segundo algoritmo implementado toma como entrada la Matriz de Detección de Eventos y produce la Matriz de Detección de Secuencias, que para el caso de redes tipo ME seguras y conectadas corresponde a la Matriz de Observabilidad.

*Algoritmo 2: Matriz de Observabilidad (Matriz de Detección de Secuencias).*Sea $E(Q)$ la matriz de Detección de Eventos del *Algoritmo 1*. Este algoritmo itera por todas las entradas vacías en $E(Q)$ de manera recursiva. Primero, elimina toda ocurrencia en $E(Q)$ del par formado por el renglón y la columna correspondientes a una entrada vacía en particular. El resultado de dicho proceso de eliminación puede ser nuevas entradas vacías en $E(Q)$, por lo que el algoritmo se aplica de manera recursiva. El diagrama de flujo de la Fig. 3 muestra los pasos recursivos del algoritmo. El resultado es la Matriz de Observabilidad $E^s(Q)$, que para el caso de ME seguras y conectadas representa su matriz de Detección de Secuencias.

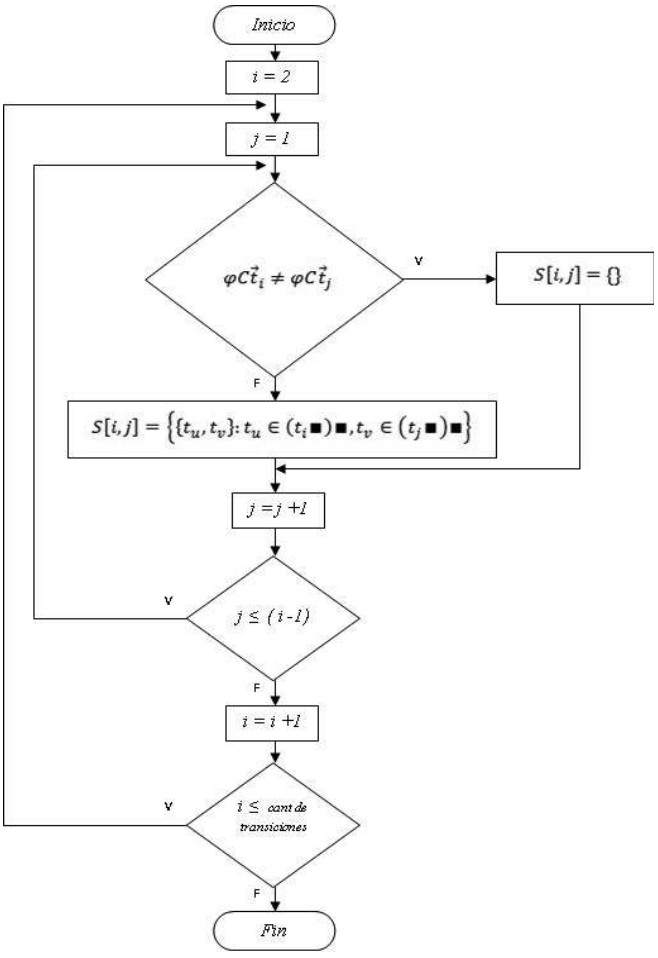


**Fig. 2. Diagrama de flujo matriz detección de eventos.**

ISBN: 978-607-28-0126-4

La Matriz de Detección de Secuencias tiene varias propiedades interesantes. Primero, en caso de ser vacía, implica directamente la propiedad de Secuencia-Detectabilidad de una ME. Segundo, se utiliza como punto de entrada para el diseño de un Observador de Secuencias. El observador de secuencias se puede implementar en el diagrama de la Fig. 1 para realizar simulaciones en entornos como Matlab/Simulink lo que permite verificar cuestiones de desempeño, como la constante de convergencia del observador. El siguiente ejemplo ilustra la utilización de los algoritmos anteriores.

## IV. Ejemplo Ilustrativo

*Ejemplo:* Considérese la red mostrada en la Fig. 4. Esta ME es segura y conectada. Cuenta con 8 lugares y 8 transiciones. Sin embargo tiene únicamente 3 sensores, denotados por las letras mayúsculas A, B, y C. La distribución de los sensores en los lugares de la red se detalla en la misma figura. La distribución de sensores tiene toda la intención de mostrar el concepto de observabilidad, ya que provoca que el observador tenga un gran número de confusiones al inicio y durante toda la evolución del sistema. Por ejemplo, si el marcado inicial pone una marca en el lugar $p_1$, entonces el observador no puede determinar si la marca se encuentra en $p_1, p_3, p_5$ o $p_7$.

La Tabla 1 muestra el resultado de la ejecución del Algoritmo 1. Se nota una gran cantidad de entradas no vacías, que representan pares de transiciones que por sí mismas se confunden. Además, dichas entradas contienen pares de transiciones a las salidas de los pares originales

Por ejemplo, en la entrada $S[t_3, t_1]$ aparece el par $\{t_4, t_2\}$ debido a que el disparo de $t_3$ hace el cambio del sensor A al B de la misma manera que el disparo de $t_1$ lo hace. En este caso, el Algoritmo 1 coloca en la matriz las transiciones que le siguen a $t_3$ y a $t_1$, que son $t_4$ y $t_2$, respectivamente. Lo mismo sucede con otras entradas mostradas en la Tabla 1.

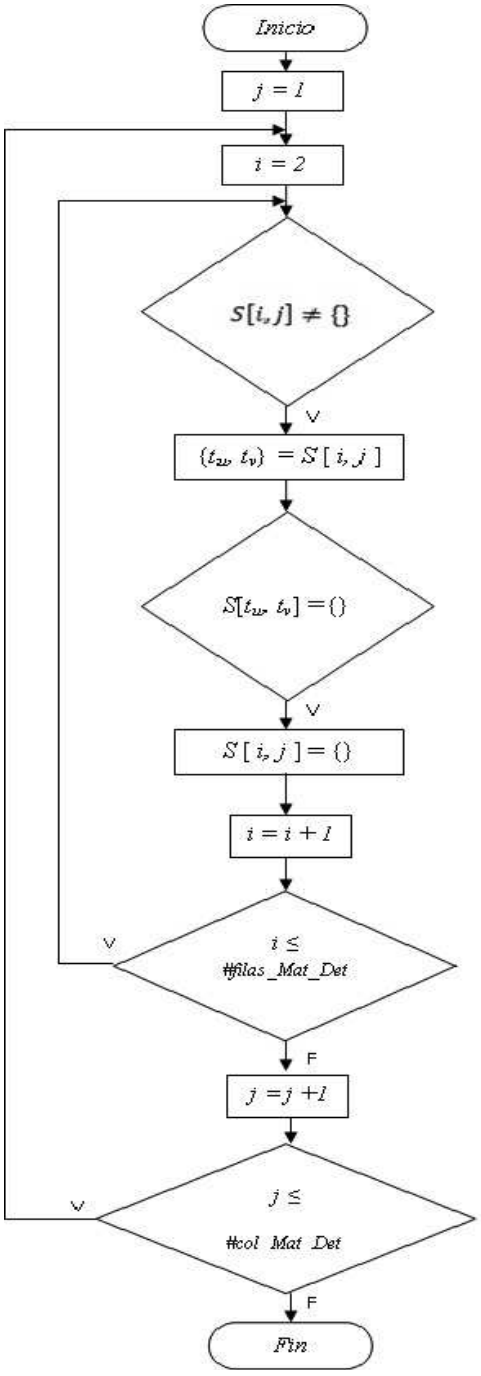


**Fig. 3. Diagrama de flujo matriz de detección de secuencias.**

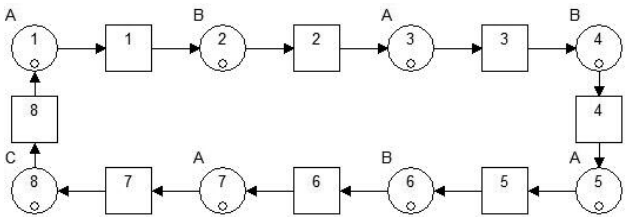


**Fig. 4. Máquina de estados segura y conectada**

| | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ |
|---|---|---|---|---|---|---|---|
| $t_2$ | { } | | | | | | |
| $t_3$ | $\{t_4, t_2\}$ | { } | | | | | |
| $t_4$ | { } | $\{t_5, t_3\}$ | { } | | | | |
| $t_5$ | $\{t_6, t_2\}$ | { } | $\{t_6, t_4\}$ | { } | | | |
| $t_6$ | { } | $\{t_7, t_3\}$ | { } | $\{t_7, t_5\}$ | { } | | |
| $t_7$ | { } | { } | { } | { } | { } | { } | |
| $t_8$ | { } | { } | { } | { } | { } | { } | { } |

**Tabla 1. Matriz *E(Q)* del algoritmo 1.**

Posteriormente se ejecuta el Algoritmo 2 sobre la matriz de Detección de Eventos $E(Q)$ mostrada en la Tabla 1. El resultado se muestra en la Tabla 2. Este segundo algoritmo busca las entradas vacías en $E(Q)$y elimina el correspondiente par de transiciones de cualquier otra entrada donde aparece.

Por ejemplo, la entrada $S[t_7, t_5]$ es vacía, lo que permite eliminar el par $\{t_7, t_5\}$ de la entrada $S[t_6, t_4]$. Esto a su vez provoca que la entrada $S[t_6, t_4]$ se vuelva vacía, lo que permite eliminar el par $\{t_6, t_4\}$ de la entrada de $S[t_5, t_3]$. El Algoritmo 2 se aplica de manera recursiva, con cada nueva entrada vacía, lo que permite propagar las nuevas entradas vacías. Utilizando las nuevas entradas vacías que se generan, es posible eliminar los pares $\{t_5, t_3\}$ y $\{t_7, t_3\}$ de la entrada $S[t_4, t_2]$ y $S[t_6, t_2]$ respectivamente. Finalmente, las entradas $S[t_3, t_1]$ y $S[t_5, t_1]$ también llegan a ser vacías, con lo que la tabla completa llega a ser vacía, con lo que se verifica la observabilidad de la red. La construcción de $E^s(Q)$ permite la obtención de un Observador de Secuencias. Las cadenas de secuencias del ejemplo de la Fig. 4 se muestran en la Tabla 3. Estas secuencias permiten a un observador dar seguimiento a la evolución del sistema y realizar estimaciones acerca de su estado. La longitud más larga de las secuencias en la tabla determina la velocidad de convergencia del observador. El ejemplo se eligió de tal manera que tuviera una constante de convergencia grande.

| | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ |
|---|---|---|---|---|---|---|---|
| $t_2$ | { } | | | | | | |
| $t_3$ | { } | { } | | | | | |
| $t_4$ | { } | { } | { } | | | | |
| $t_5$ | { } | { } | { } | { } | | | |
| $t_6$ | { } | { } | { } | { } | { } | | |
| $t_7$ | { } | { } | { } | { } | { } | { } | |
| $t_8$ | { } | { } | { } | { } | { } | { } | { } |

**Tabla 2. Matriz $E^S(Q)$ del algoritmo 2**.

| | | | | | | |
|---|---|---|---|---|---|---|
| $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ | | |
| $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ |
| $t_4$ | $t_5$ | $t_6$ | $t_7$ | | | |
| $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ | |
| $t_5$ | $t_6$ | $t_7$ | | | | |
| $t_6$ | $t_7$ | | | | | |
| $t_7$ | | | | | | |
| $t_8$ | | | | | | |

**Tabla 3. Secuencias del observador**

## V. Conclusiones

En este trabajo se presenta la implementación de algoritmos para la construcción de observadores para sistemas de eventos discretos, modelados mediante Redes de Petri. El artículo se centra específicamente en Máquinas de Estado, debido a que su estructura esta bien definida y han sido ampliamente estudiadas en la literatura. El primer algoritmo obtiene como resultado la Matriz de Detección de Eventos, la cual sirve como entrada a un segundo algoritmo que da como resultado la Matriz de Detección de Secuencias. La primera matriz determina las transiciones que se detectan inmediatamente unas de otras, mientras que la segunda matriz determina las secuencias de transiciones que se distinguen de aquellas que no lo hace. Los algoritmos presentados permiten la construcción de observadores de secuencias, además de obtener información, como las cadenas de secuencias, que se pueden utilizar en el análisis de sistemas de eventos discretos.

ISBN: 978-607-28-0126-4

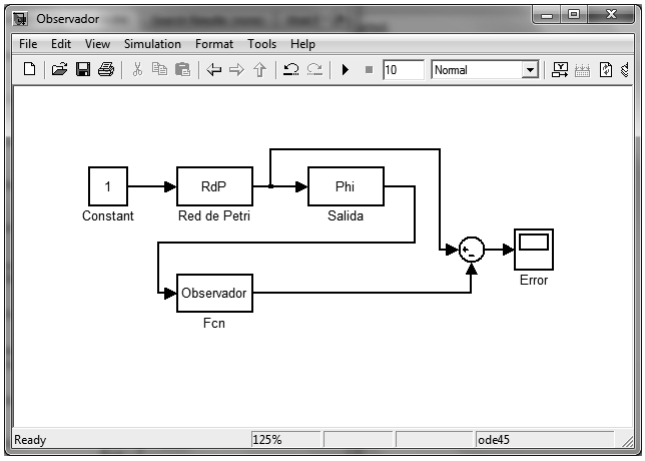


**Fig. 5. Diagrama de flujo matriz detección secuencias.**

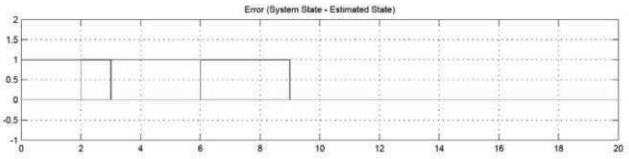


**Fig. 6. Diagrama del error del observador.**

Los resultados en simulación muestran que los algoritmos se pueden utilizar para simular el comportamiento del observador en Matlab/Simulink a fin de observar su desempeño previo a su implementación.

## VI. Referencias

[1] Lafortune Stephane, Cassandras Christos, Introduction to Discrete Event Systems, vol. 11, Int. Series on Discrete Event Dynamic Systems. Springer, 1999.

[2] J.F. Sánchez-Blanco, A. Ramirez-Trevino, and A. Santoyo, "Multiple specification regulation control in interpreted petri nets," IEEE Proc. Int. Conf. SMC, 2004, vol. 5, pp. 4989—4994.

[3] Y. Li, W. M. Wonham, "Control of Vector Discrete-Event Systems. I. The base model", IEEE Trans. Automatic Control, vol. 38(8):1214—1227, 1993. 0018-9286.

[4] R. Campos-Rodríguez, CINVESTAV-IPN, Tesis de Doctorado: "Algoritmos para el Control de Sistemas de Eventos Discretos bajo Observación Parcial del Estado", Guadalajara, Jalisco, México, Junio de 2007.

[5] J. Desel, J. Esparza, Free Choice PetriNets, Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, Nueva Edición, Septiembre 8, 2005, ISBN-10: 0521019451.

[6] R. Campos-Rodríguez, M. Alcaraz-Mejía. "A matlab/simulink framework for the design of controllers and observers for discrete-event systems", Electronics and Electrical Engineering, No. 3(99), 2010, ISSN 1392-1215.

[7] R. Campos-Rodriguez, A.Ramirez-Trevino, E. Lopez-Mellado, "Observability Analysis of Free-choice Petri net Models," IEEE Proc. System of System Eng., 2006, pp. 77-82.

[8] M. Silva, Las redes de Petri: En la automática y la informática, Editorial AC, 1985, ISBN-10: 8472880451.

[9] R. Campos Rodríguez, M. Alcaraz Mejía, J. Mireles Garcia, "Supervisory control of discrete event systems using observers", Control & Automation, 2007.MED '07. Mediterranean Conference on Publication Date: 27-29, June 2007, Págs: 1-7, ISBN: 978-1-4244-1282-2, Digital Object Identifier: 10.1109/MED.2007.4433816.

[10] A. Ramirez Treviño, I. Rivera Rangel, E. López Mellado, "Observability of discrete event systems modeled by interpreted Petri Nets", IEEE Transactions on Automatic Control, Vol. 19, No. 4, August 2003.

## VII. Autores

**M. en C. Diego Ulises Carranza Sahagún** es Ingeniero en Computación por la Universidad de Guadalajara (1999) Obtuvo el grado de Maestría en Computación Aplicada, (mención Programación) en la Universidad Marta Abreu de las Villas, Cuba. (2002) Actualmente se desempeña en el área de programación de sistemas y es estudiante del Doctorado en Ciencias en el CUCiénega de la Universidad de Guadalajara

**Dr. Raúl Campos Rodríguez** es Ingeniero en Computación por la Universidad de Guadalajara, (2000). Estudió la Maestría (2002) y Doctorado (2007) en Ciencias en Ingeniería Eléctrica en el CINVESTAV-IPN Actualmente es profesor-investigador en el Departamento de Electrónica, Sistemas e Informática del ITESO A.C., donde además coordina la Especialidad en Sistemas Embebidos.

**Dra. Mildreth Alcaraz Mejía** es Ingeniero en Sistemas Computacionales por el Tecnológico de Colima (2000). Estudió la Maestría (2002) y Doctorado (2007) en Ciencias en Ingeniería Eléctrica en el CINVESTAV-IPN. Actualmente es profesor-investigador en el Departamento de Electrónica, Sistemas e Informática del ITESO A.C., donde además coordina la Maestría en Sistemas Computacionales.

**Dr. Cuauhtémoc Acosta Lúa** es Ingeniero en Electrónica por el Tecnológico de Morelia (2001). Obtuvo los títulos de Maestro (2003) y Doctor (2007) en Ciencias con especialidad en Ingeniería Eléctrica en el CINVESTAV-IPN. Actualmente es profesor-investigador en el CUCiénega, Universidad de Guadalajara.

ISBN: 978-607-28-0126-4