UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Tesis para obtener el grado de PhD

Identificación y Diagnóstico de Fallos en Sistemas de Eventos Discretos Estocásticos.

Doyra Mariela Muñoz Añasco

Julio 2015

Director: Dr. Antonio Correcher Salvador
Instituto de Automática e Informática Industrial
Supervisión y Diagnóstico de Fallos.

A Laura y Eduardo por su sincero y oportuno acompañamiento en todo este proceso y por hacer de esto, también su sueño.

Índice general

Índice de figuras

Índice de tablas

Resumen

Este trabajo presenta un método de diagnóstico de fallos para sistemas de eventos discretos estocásticos, sin modelo previo. Para lograr el objetivo, el método identifica el comportamiento normal a partir de las señales de entrada / salida (E/S) del sistema obtenidas on-line. Cada señal es discretizada mediante un generador de eventos, así el sistema es modelado bajo la teoría de lenguajes. Además de la generación de eventos, el método también identifica el tiempo entre eventos, de esta manera el lenguaje del comportamiento normal puede ser modelado como una red de Petri, interpretada, temporizada, estocástica; la cual solo representa el lenguaje observado y evita el no-determinismo. Una vez se ha identificado el comportamiento normal, el método de diagnóstico propuesto compara el lenguaje identificado con el lenguaje observado on-line; si hay desviación entre los lenguajes, se ha detectado un fallo. Este trabajo presenta un diagnosticador que es capaz de usar esa información para detectar el fallo y aprender el comportamiento fallido. El sistema es modular y esto incluye herramientas para localizar el fallo. La información recolectada es una buena base para que un experto diagnostique totalmente el fallo.

Summary

This work presents a fault diagnosis method for stochastic discrete event systems without previous model. To achieve this goal, the

method identifies the normal behavior from to online input/output system signals. Each signal is discretized through an event generator, so the system is modelled from the language theory. Besides the pure event generation, the method also identifies the time between events, so the normal behaviour language can be modeled with a stochastic, timed, interpreted Petri net which represents only the observed language and avoids the non-determinism. Once the normal behavior has been identified, the diagnostic method compares the identified language with on-line observed language. If there is any deviation, then a fault has been detected. This work presents a diagnoser that is able to use that information to detect the fault and to learn the faulty behavior. The system is modular and it includes tools to locate the fault. The collected information is a good base for an expert to fully diagnose the fault.

Resum

Aquest treball presenta un mètode de diagnosi de fallades per a sistemes de esdeveniments discrets estocàstics sense model anterior. Per aconseguir aquest objectiu, el mètode identifica el comportament normal observant les senyals de eixida/entrada en línias. Cada senyal és discretiza amb un generador d'esdeveniment, així doncs, el sistema es modela amb la teoria de llenguatges. A més de la generació d'esdeveniment pura, el mètode també identifica el temps entre esdeveniments, així que el comportament normal es pot modelar amb una Xarxa de Petri estocàstica, temporitzada e intrepretada que representa només el llenguatge observat i evita el no-determinisme. Una vegada que el comportament normal ha estat identificat, el mètode de diagnòstic compara el llenguatge identificat amb el observat en línia. Si hi ha qualsevol desviació llavors una fallada ha estat detectada. El diagnosticador és capaç

d'utilitzar aquesta informació per detectar la fallada i per aprendre el seu comportament. El sistema és modular i inclou eines per localitzar la fallada. La informació recollida és una bona base per a que un expert puga plenament diagnosticar la fallada.

Agradecimientos

El presente trabajo de investigación se ha realizado gracias a la colaboración del grupo de investigación Supervisión y Diagnóstico de Fallos de la Universitat Politècnica de València, UPV, y a la Universidad del Cauca por la concesión de la comisión académica para realizar mi estudio doctoral.

Especialmente agradezco a mi director, Dr. Antonio Correcher, por el asesoramiento, el apoyo y la dedicación durante el desarrollo de este trabajo, pues su cocimiento y orientación han sido el eje fundamental para llevar a cabo la investigación y lograr los resultados.

A mi familia agradezco su apoyo incondicional y por creer que podía llevarlo a cabo, a mis compañeros de la Universidad del Cauca su oportuna colaboración y a todos mis amigos en Valencia que hicieron que esta estancia fuese muy agradable.

Gracias

Mariela

1 Introducción

La competitividad hace que las empresas implementen estrategias para el mejoramiento de la productividad y el aumento de la seguridad de los operarios. Una estrategia importante en este proceso, es la definición de políticas apropiadas de mantenimiento preventivo y correctivo de los procesos. El diagnóstico de fallos cobra importancia en este sentido, puesto que una eficaz detección de fallos es una herramienta clave para tomar medidas apropiadas para generar políticas de mantenimiento, que permitan disminuir costos y aumentar la fiabilidad de los sistemas; por lo tanto, el diagnóstico de fallos se ha convertido en un importante tema de investigación tanto en la parte industrial como en la academia.

Un fallo es una desviación del sistema de su comportamiento normal o requerido, que puede ser causado por la ruptura o mal funcionamiento de un dispositivo o de un componente del sistema. El diagnóstico de fallos es el proceso de detectar, aislar e identificar tales desviaciones del sistema, determinando las causas que lo producen usando la información disponible de las variables del sistema. La detección es una funcionalidad que decide si el sistema trabaja en condiciones normales o si ha ocurrido un fallo; si ha ocurrido un fallo, el objetivo del aislamiento es localizar el componente(s) del sistema que causa el fallo y la identificación se relaciona con la definición de la naturaleza específica del fallo, [4].

El diagnóstico de fallos en sistemas de eventos discretos (SED) ha sido dirigido por muchos investigadores con el objeto de desarrollar

nuevos modelos, nuevas propiedades, nuevos algoritmos y eficientes soluciones para el diagnóstico de SED. Estos métodos se dividen en tres grupos (de acuerdo a [5]): Los enfoque basados en reglas, (if then else), los enfoques conducidos por datos (minería de datos o reconocimiento de patrones) y los enfoques basados en modelos. A su vez, los basados en modelos, pueden ser divididos en dos grupos: El primer grupo considera modelos que tienen comportamiento libre de fallo así como modelos con comportamiento de fallo y el segundo grupo usa solo modelos libres de fallo.

La principal desventaja que tienen los métodos con modelo de fallo, es que es necesario tener un conocimiento exhaustivo respecto a los posibles fallos del sistema; por lo tanto, sólo los fallos considerados explícitamente en el modelo del sistema pueden ser detectados y localizados; esto conlleva a que los algoritmos propuestos identifiquen y diagnostiquen fallos en sistemas de baja complejidad. Los programas de mantenimiento suelen basarse en fallos predefinidos; sin embargo, la presencia de errores a nivel industrial no se limita solo a estos fallos predefinidos; así, algunos fallos no pueden ser diagnosticados. Este problema aumenta cuando se trata de nuevos procesos, modificaciones sobre las líneas de producción o sobre máquinas.

Los métodos de diagnóstico, sin modelo de fallo, evitan este inconveniente. Éstos consisten básicamente en detectar comportamientos no deseados comparando la información observada en los sistemas de supervisión, control y adquisición de datos (SCADA) de los procesos automatizados, con lecturas esperadas; si hay diferencia se ha detectado un fallo. Sin embargo, el aislamiento y la identificación no pueden ser posibles puesto que el modelo no incluye el comportamiento en fallo; por lo tanto, la diagnosticabilidad de un fallo dado no se puede garantizar.

La literatura respecto a métodos de diagnóstico sin modelo de fallo es muy escasa, en [6] los autores utilizan plantillas de condición

para determinar si el sistema genera eventos en el orden correcto o dentro de los plazos de tiempo determinados, se detecta un fallo cuando hay ausencia o reacciones equivocadas en el proceso. En estos casos, los eventos relacionados con la plantilla ayudan a aislar el fallo. En [7] un fallo es detectado por la ocurrencia de un evento inesperado o por la no ocurrencia de un evento esperado dentro de intervalos de tiempo predefinidos, la ocurrencia de un evento inesperado se detecta cuando la condición de habilitación de este evento no está satisfecha y la no ocurrencia de un evento esperado se detecta mediante una plantilla. Roth en [8] inicia con la identificación del modelo libre de fallo del sistema y luego compara el comportamiento observado con el nominal y detecta fallos a partir de la teoría de residuos. Muestra cómo un modelo identificado como un autómata no determinístico, puede ser utilizado para realizar diagnóstico sin tener en consideración la temporización de los eventos.

1.1. Motivación y Objetivos

Teniendo en cuenta el incremento de la complejidad de los sistemas industriales y dadas las restricciones de los métodos de diagnóstico basados en modelos tales como: conocimiento exhaustivo del sistema en comportamiento de fallo, modelos complejos, modelos no-determinísticos y algunos con costo computacional alto; en este trabajo se pretende generar un método que diagnostique fallos sin conocimiento previo del modelo del sistema, ni en funcionamiento normal ni en comportamiento de fallo y que a su vez presente ventajas frente a propuestas con el mismo concepto (sin modelo de fallo), como es la aplicabilidad a sistemas reales de gran escala con comportamiento estocástico y que bajo la teoría de lenguajes permita realizar pruebas de detectabilidad.

Para solucionar el problema, se deben tener en cuenta varios aspectos: Por un lado, para realizar el proceso de detección de un fallo es necesario obtener un comportamiento normal del sistema; el principal problema es encontrar este modelo, sobre todo cuando se tratan de grandes sistemas industriales. La idea es obtener on-line, el modelo del sistema a partir de las señales de entrada - salida (E/S) de un SED en lazo cerrado y que sea representado en un generador determinístico de lenguaje, con el objeto que luego permita comparar el lenguaje observado con el lenguaje normal del generador. Bajo el supuesto que el modelo identificado es preciso y completo, una desviación entre el lenguaje observado y el lenguaje normal implica necesariamente un fallo en el sistema.

Por otro lado, en un método de diagnóstico sin modelo previo, se puede realizar la detección y en algunas ocasiones la localización, pero no es posible realizar la identificación; por lo tanto no se puede cumplir con las tres tareas para completar el diagnóstico; entonces es necesario encontrar un mecanismo que permita extraer la información necesaria y suficiente de los resultados de obtenidos para localizar el fallo para que un experto pueda establecer el tamaño, criticidad e importancia y de esta forma completar con el diagnóstico.

Teniendo en cuenta lo anterior, el objetivo principal de la tesis doctoral es:

- Generar un método de identificación y diagnóstico de fallos sin modelo previo, aplicable a sistemas de eventos discretos industriales, temporizados, ya sea de naturaleza determinística o de naturaleza estocástica que trabaje de manera independiente al tamaño del sistema.

Ese objetivo puede cumplirse a partir de los siguientes objetivos específicos:

- Representar las señales de entrada - salida de un sistema de

eventos discretos temporizado en eventos, con el objeto de definir un lenguaje regular temporizado.

- Proponer una estructura que genere un lenguaje regular temporizado para modelar el comportamiento del sistema de eventos discretos, que solo represente el lenguaje del sistema, posea propiedades de determinismo y observabilidad y que pueda ser aplicado a problemas de diagnóstico.
- Definir y validar un proceso de identificación de sistemas de eventos discretos de naturaleza determinística y estocástica, aplicable a sistemas de gran escala.
- Definir y validar un proceso de diagnóstico de fallos a partir de la comparación del lenguaje normal y del lenguaje observado y una estructura apropiada de diagnosticador que represente el lenguaje normal y el lenguaje de fallo.

1.2. Contribuciones de la tesis

Los resultados obtenidos parten de un adecuado manejo de las señales internas y externas, temporizadas de un SED en lazo cerrado, para ser convertidas en eventos y bajo la teoría de lenguajes, se ha logrado establecer un generador de lenguaje que modela sistemas estocásticos de gran tamaño, adecuado para realizar procesos de diagnóstico; además, evita la explosión de estados y el no-determinismo y se basa en una estructura temporizada, lo cual presenta ventajas frente a otras propuestas de diagnóstico.

Se presenta un algoritmo de identificación de SED, capaz de manejar gran cantidad de variables y obtener un modelo de un SED estocástico, como una red de Petri interpretada, temporizada, estocástica (st-IPN). Las transiciones de la st-IPN, generan información sobre la función de densidad de probabilidad del tiempo entre

eventos para cada modo de funcionamiento del sistema.

Con base en la estructura propuesta de evento compuesto temporizado, se define la observabilidad y detectabilidad de eventos inobservables; así como el test de identificabilidad del lenguaje generado por la st-IPN.

Con una pequeña adecuación, el modelo identificado representado como una red de Petri con características particulares, se convierte en un diagnosticador con estructura variable que puede ser usado como observador de un proceso, capaz de representar lenguajes con comportamiento normal y con comportamiento de fallo a partir de la comparación entre ellos. Dada su estructura variable, aprende los lenguajes de fallo detectados y actualiza su estructura.

La división en subsistemas permite localizar los fallos y la información generada por el algoritmo de diagnóstico caracteriza los fallos detectados; ya que, define la señal de entrada o de salida que presenta un comportamiento no deseado en cuanto a valor y/o tiempo, el instante en el que ocurre, así como el estado en el cual se encontraba el sistema. Esta información la puede analizar un experto en el sistema con el objeto de identificar la naturaleza del fallo y se puedan tomar medidas para corregirlos o prevenirlos. De esta manera se completa la tarea de diagnóstico.

Se propone un algoritmo de filtrado de fallos, el cual detecta y elimina fallos de propagación, disminuyendo de esta manera las falsas alarmas.

1.3. Organización del Documento

El presente documento costa de ocho capítulos; en el capítulo 2, se presenta una descripción de conceptos teóricos que son la base para abordar el tema de investigación; el capítulo 3, relaciona el

estado del arte respecto a las contribuciones realizadas en torno al modelado, identificación y diagnóstico de fallos en sistemas de eventos discretos, en el ámbito académico; en el capítulo 4 se formaliza la propuesta de modelado bajo una red de Petri interpretada, temporizada, estocástica (st-IPN) y define sus características particulares; en el capítulo 5 se presenta el método de identificación de sistemas de eventos discretos estocásticos, propuesto; en el capítulo 6 se expone el método de diagnóstico propuesto, se definen y comprueban sus propiedades y se muestran las ventajas frente a propuestas existentes; en el capítulo 7, se comprueba la aplicabilidad del método de diagnóstico en diferentes procesos reales y por último, se presentan las conclusiones y se relacionan posibles trabajos futuros.

1.4. Publicaciones

- Identification of Stochastic timed Discrete Event Systems with st-IPN. Mathematical Problems in Engineering, 2014. (Publicado)
- Fault detection and isolation in a photovoltaic system. International Conference on Renewable Energies and Power Quality (ICREPQ'15), La Coruña (Spain), Renewable Energy and Power Quality Journal (RE&PQJ) ISSN 2172-038 X, No.13, April 2015. (Aceptado).
- Stochastic DES fault diagnosis with Coloured Interpreted Petri Nets. Mathematical Problems in Engineering, 2015. (Publicado).
- Generación Determinística de Lenguajes Legales para SED. Revista Iberoamericana de Automática e Informática Industrial, 2015. (En revisión de correcciones).

2 Conceptualización Teórica

En esta sección se presentan los conceptos teóricos que se utilizarán a largo de esta investigación relacionados con la definición de sistemas de eventos discretos (SED), redes de Petri (RdP o PN por sus siglas en ingles, Petri Nets), lenguajes y técnicas de análisis del comportamiento estocástico de datos.

2.1. Sistema y tipos de sistemas

Guash et al. [9], definen un sistema como una colección de objetos o entidades que interactúan entre sí para alcanzar un cierto objetivo y el estado de un sistema como un conjunto mínimo de variables necesarias para caracterizar o describir todos aquellos aspectos de interés del sistema en un cierto instante de tiempo; a estas variables se les denomina variables de estado.

Según Guash et al., los sistemas se pueden clasificar según su comportamiento a lo largo del tiempo en:

- Sistemas Continuos: Aquellos en los cuales las variables de estado evolucionan de un modo continuo a lo largo del tiempo.
- Sistemas Discretos: Son aquellos en los que las propiedades de interés del sistema cambian en un cierto instante o secuencia de instantes, que normalmente obedecen a un patrón periódico.

- Sistemas Orientados a Eventos: Al igual que los sistemas discretos, se caracterizan porque las propiedades de interés del sistema cambian únicamente en una secuencia de instantes de tiempo, permaneciendo constantes el resto del tiempo. La secuencia de instantes en los cuales el estado del sistema puede presentar un cambio obedece a un patrón aleatorio.
- Sistemas Combinados: Son aquellos que combinan subsistemas continuos o discretos respectivamente.

Los sistemas en tiempo continuo se representan mediante ecuaciones diferenciales, en tiempo discreto se suelen representar mediante ecuaciones en diferencia y si las variables pueden cambiar en cualquier momento se denominan de eventos discretos.

Los sistemas de interés en esta investigación son los SED cuyo comportamiento se considera: dinámico porque las variables de interés evolucionan respecto al tiempo; estocástico, pues requieren de una o más variables aleatorias para formalizar las dinámicas a investigar y discreto porque las variables de estado cambian de valor en instantes no periódicos de tiempo. Estos instantes de tiempo se corresponden con la ocurrencia de un evento. Un evento es una acción instantánea que puede cambiar el estado del modelo.

El comportamiento de un SED se caracteriza por una secuencia finita o infinita de estados delimitados por eventos que ocurren de manera asíncrona o síncrona [10] y se puede describir a través de la ocurrencia de eventos desde un estado inicial. Un SED es un sistema dinámico que evoluciona con la aparición, a intervalos irregulares, de acontecimientos físicos, [11]. Surgen en muchos ámbitos, en la fabricación, robótica, tráfico de vehículos, logística y en las redes informáticas.

Una manera de estudiar el comportamiento de los SED es emplear la teoría de lenguajes. Un evento es la representación de un cambio instantáneo en alguna parte del sistema, puede caracterizarse por

un valor y un instante en el que ocurre, [12]. Un evento constituye una letra y el conjunto de eventos es el alfabeto, una secuencia de eventos es una palabra, [10]. Al conjunto de todas las combinaciones válidas de eventos que pueden ocurrir en el sistema se le denomina lenguaje, $\mathcal{L}$. El lenguaje que modela el comportamiento de un sistema se puede representar por su expresión regular, [13], que es una síntesis de todas las combinaciones válidas de eventos.

En [11] un SED es modelado como un autómata y su comportamiento es representado por su lenguaje; pero este tipo de modelos pueden presentar inconvenientes en sistemas con gran cantidad de dispositivos debido a la explosión combinatoria de estados, suele ser más conveniente emplear formalismos con mayor capacidad de condensación de estados, como las PNs o las máquinas de estados finitos.

2.2. Eventos y Lenguaje del sistema

Sea Ω un conjunto de eventos; Ω^* es un conjunto finito de palabras sobre Ω, un lenguaje $\mathcal{L}$ es un subconjunto de Ω^*. El conjunto de eventos, Ω, también incluye el evento nulo, ε, que modela la situación en la que no ocurre ningún evento, [14].

Dadas dos palabras s y $s\prime$, $ss\prime$ es la concatenación de s y $s\prime$. $|s|$ es el tamaño de la palabra s.

Definición 1 (Operación de Proyección)**.** Dados dos conjuntos de eventos Ω_1, Ω tales que $\Omega_1 \subseteq \Omega$, se define la operación de proyección de palabras, $P_{\Omega_1} : \Omega^* \to \Omega_1^*$, así $P_{\Omega_1}(\varepsilon) = \varepsilon$ siendo ε un evento nulo, y para $a \in \Omega$, $s \in \Omega^*$, $P_{\Omega_1}(as) = aP_{\Omega_1}(s)$ si $a \in \Omega_1$ y $P_{\Omega_1}(as) = P_{\Omega_1}(s)$ si $a \notin \Omega_1$.[15]

Esta operación permite eliminar palabras en Ω^* que no pertenecen

a Ω_1^*. Dentro del contexto del presente trabajo es importante definir un evento compuesto y la proyección sobre lenguajes compuestos.

Definición 2 (Evento Compuesto)**.** Dados dos conjunto de eventos Ω_p, Ω_q y dados dos eventos e_p y e_q, tal que, $e_p \in \Omega_p$ y $e_q \in \Omega_q$, un evento compuesto ω es la concatenación de e_p y e_q, $\omega = e_p e_q$. $\Omega = \Omega_p\Omega_q$ es un conjunto de eventos compuestos (Ω sobre $\Omega_p\Omega_q$) y un lenguaje definido sobre Ω será $\mathcal{L} \subset (\Omega_p\Omega_q)^*$.

Ejemplo 1. Dados dos conjunto de eventos $\Omega_p = \{p_1, p_2\}$, $\Omega_q = \{q_1, q_2\}$ un conjunto de eventos compuestos será $\Omega = \{(p_1q_1), (p_1q_2), (p_2q_1), (p_2q_2)\}$.

Definición 3 (Operación de Proyección sobre conjuntos de eventos compuestos, Pc)**.** Dado un conjunto de eventos compuestos Ω sobre $\Omega_p\Omega_q$. La operación Pc se define como: $Pc : \Omega \to \Omega_q$, es $Pc_{\Omega_q}(\varepsilon) = \varepsilon$; $Pc_{\Omega_q}(e_p e_q) = P_{\Omega_q}(e_p) P_{\Omega_q}(e_q)$.

Para el ejemplo anterior $Pc : \Omega \to \Omega_p$; $Pc_{\Omega_p}(p_2q_2) = p_2$. Dada una secuencia de eventos $s \mid s \in \Omega^*$ donde $s = (p_1q_1)(p_2q_2)(p_1q_1)$, Pc de s sobre Ω_p es $Pc_{\Omega_p}(s) = p_1p_2p_1$.

Esta operación elimina palabras en Ω^* que no pertenecen a $\left(\Omega_p^*\Omega_q^*\right)$ y eventos Ω_p (Ω_q) que no pertenecen a Ω_p^* (Ω_q^*).

Definición 4 (Evento temporizado)**.** Un evento temporizado es una composición de un evento y el tiempo transcurrido entre los dos eventos consecutivos. Entonces $e^i = ev.t^{ev}$ al instante τ_i donde $t^{ev} = |\tau_{current\_ev}| - |\tau_{previous\_ev}|$.

El lenguaje es un conjunto finito de secuencias de símbolos o eventos $\sigma = e_1e_2\cdots e_n$ conocidos como palabras; la adición de tiempo a una palabra puede hacerse dando a cada símbolo un valor de tiempo $t \in \mathbb{N}$, resultando una secuencia de pares de símbolos

símbolo-tiempo $\sigma = (e_1, t_1)(e_2, t_2)\cdots(e_n, t_n)$. Cada valor de tiempo de la palabra temporizada representa el tiempo que ha transcurrido desde la ocurrencia del evento anterior. Un conjunto formado por palabras temporizadas se llama un lenguaje temporizado,[16].

De acuerdo a [10], el lenguaje generado por un SED puede ser considerado con tres niveles de abstracción: lenguaje temporizado, lenguaje temporizado estocástico y lenguaje sin tiempo; es decir, las palabras generadas tienen o no información del tiempo. El alcance de este trabajo es el lenguaje estocástico temporizado.

Definición 5 (Lenguaje temporizado y Lenguaje temporizado estocástico)**.** Un lenguaje, $\mathcal{L}$, definido sobre un conjunto de eventos temporizados Ω, es un conjunto de cadenas de tamaño finito formadas a partir de eventos temporizados en Ω, es decir, $\mathcal{L} \subset \Omega^*$; $\mathcal{L} = \{s \,|s \subset \Omega^*\}$ donde s es una secuencia de eventos temporizados a instantes $\tau_0 \leq \cdots \leq \tau_k$. En sistemas reales no es posible que un mismo evento siempre tenga la misma duración de tiempo. En tal caso, el tiempo se ajusta a una función de densidad de probabilidad, $t^{ev} \sim f(t^{ev})$, entonces los eventos se llaman eventos temporizados estocásticos.

La operación de proyección de eventos temporizados sobre lenguajes temporizados se define como:

Definición 6 (Operación de Proyección de secuencias de eventos temporizados: Pt)**.** Dados dos conjuntos de eventos temporizados Ω_1, Ω, tal que $\Omega_1 \subseteq \Omega$ y sea $s = e^0 \cdots e^k$ una secuencia de eventos temporizados, con $e^i = e.t^{ev}$; $Pt : \Omega^* \to \Omega_1$, Pt se define como: $Pt_{\Omega_1}(s) = Pt_{\Omega_1}(e^0) \cdots Pt_{\Omega_1}\left(e^k\right)$, $\forall e^i \in s$. La cual es calculada de la siguiente manera: una variable es inicializada a cero $t_{proy} = 0$; entonces, para $i = 0 \cdots k$; $Pt_{\Omega_1}(e^i) = e.t^{ev}$ si $e^i \in \Omega_1$ entonces $t_{proy} = t^{ev}$ y la variable regresa a cero, $t_{proy} = 0$; de lo contrario $Pt_{\Omega_1}(e^i) = \varepsilon$ y $t_{proy} = t_{proy} + t^{ev}$.

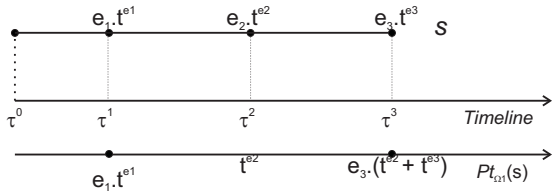


Figura 2.2.1: Ejemplo de Pt.

Ejemplo 2. Dado un conjunto de eventos temporizados $\Omega_1 = \{e_1.t^{e_1}, e_3.t^{e_3}\}$ y dada una secuencia $s = e_1.t^{e_1}e_2.t^{e_2}e_3.t^{e_3}$, basado en la Figura 2.2.1, la $Pt : \Omega^* \rightarrow \Omega_1$; $Pt_{\Omega_1}(s) = Pt_{\Omega_1}(e_1.t^{e_1})$ $Pt_{\Omega_1}(e_2.t^{e_2})$ $Pt_{\Omega_1}(e_3.t^{e_3})$; entonces:

- para $i = 0$,$e_1.t^{e1} \in \Omega_1$, entonces $t_{proy} = t^{e1}$,$Pt_{\Omega_1}(e_1.t^{e1}) = e_1.t_{proy}$ y $t_{proy} = 0$;
- para $i = 1$, $e_2.t^{e_2} \notin \Omega_1$, entonces $t_{proy} = t_{proy} + t^{e_2} = t^{e_2}$ y $Pt_{\Omega_1}(e_2.t^{e_2}) = \varepsilon$;
- para $i = 2$, $e_3.t^{e_3} \in \Omega_1$, entonces $t_{proy} = t_{proy} + t^{e_2} = t^{e_3} + t^{e_2}$ y $Pt_{\Omega_1}(e_3.t^{e_3}) = (e_3.t_{proy})$.

Por lo tanto: $Pt_{\Omega_1}(s) = e_1.t^{e1}e_3.(t^{e_3} + t^{e_2})$.

Dado un conjunto de eventos temporizados estocásticos $\Omega_1 = \{e_1.$ $f(t^{e1}), e_2.f(t^{e2})\}$ y dada una secuencia de eventos temporizados $s = e_1.(t^{e_1*})\, e_3.(t^{e_3})$ la Pt de s sobre Ω_1 es $Pt_{\Omega_1}(s) = Pt_{\Omega_1}(e_1.t^{e_1*})$ $Pt_{\Omega_1}(e_3.t^{e_3})$; $Pt_{\Omega_1}(e_1.t^{e_1*}) = e_1.f(t^{e1})$ si $e_1 \in \Omega_1 \wedge (a \leq t^{e_1*} \leq b)$; donde $\int_a^b f(t^{e_1}) \geq (1-\alpha)$ y $[a, b]$ es el intervalo de confianza de $f(t^{e_1})$.

2.3. Bases sobre Redes de Petri

Las redes de Petri (PN) son modelos capaces de describir el flujo de información total de un sistema con procesos concurrentes y

distribuidos.

Las PN son ampliamente usadas para modelar SED, [17, 18]. Proveen modelos compactos y capturan características importantes de los SED, como concurrencia, sincronismo, relaciones causales, recursos compartidos, etc. La presente propuesta se basa en PN, por ese motivo se presentan a continuación algunas definiciones.

Definición 7 (PN)**.** Una PN N, es un grafo bipartito representado por la tupla $N = (P, TR, Pre, Post, M_0)$ [19] donde: P es el conjunto de lugares con cardinalidad np y TR es el conjunto de transiciones con cardinalidad ntr, $Pre : P \times TR \to \mathbb{N}$, $Post : TR \times P \to \mathbb{N}$ son la matrices que contienen los arcos que conectan lugares y transiciones. La matriz de incidencia $I = Post - Pre$ es una matriz de np x ntr. La función de marcado $M : P \to \mathbb{N}$ representa el número de marcas en cada lugar, M_0 es el marcado inicial, [19, 20].

Los conjuntos de lugares previos y posteriores a una transición, se denotarán como: $\bullet tr = \{p \in P : Pre\,(p, tr) > 0\}$ $tr\bullet = \{p \in P : Post\,(p, tr) > 0\}$, [20].

Definición 8 (Conjunto de alcanzabilidad de una PN)**.** Sea N una PN, la transición tr_r está habilitada si $\forall p_q \in \bullet tr_r$, $m\,(p_q) \geq I\,(p_q, tr_r)$. La transición habilitada tr_r puede ser disparada alcanzando un nuevo marcado M_{k+1} que se calcula por $M_{k+1} = M_k + I.\overrightarrow{tr_r}$, donde $\overrightarrow{tr_r}$ es un vector de ceros de tamaño $|TR|$, excepto en la posición r, que es igual a 1. La evolución del marcado de una PN, dada una secuencia de disparo $\sigma = tr_1 tr_2 \cdots tr_k$ se denota como $M\,[\sigma > M_k$. El conjunto de alcanzabilidad de una PN es el conjunto de todos los posibles marcados alcanzables desde M_0, disparando solo transiciones habilitadas; este conjunto es denotado por $R(N, M_0)$. [21]

Una extensión de las PN son las redes de Petri Interpretadas (IPN, por sus siglas en inglés, interpreted Petri Nets) que asocian señales de entrada y salida al modelo, [22].

Definición 9 (Red de Petri Interpretada (IPN))**.** Una IPN es una tupla $Q = (N, \Sigma, \Phi, \lambda, \varphi)$, donde, N es una PN como la definida en Definición 7, Σ es el conjunto de símbolos de entrada. Φ es el conjunto de símbolos de salida. $\lambda : TR \to \Sigma \cup \varepsilon$ es una función de etiquetado que asigna un símbolo de entrada a cada transición. $\varphi : R(N) \to \Phi$ es una función de salida que asigna un símbolo de salida a cada marcado alcanzado.

Una $tr_r \in TR$ de una IPN está habilitada si $\forall p_q \in \bullet tr_r$, $m(p_q) \geq I(p_q, tr_r)$. Si $\lambda(tr_r) \neq \varepsilon$ está presente y si tr_r está habilitada entonces tr_r se dispara y se alcanza un nuevo marcado M_{k+1}, el cual se calcula mediante la ecuación de estado:

$$\begin{array}{rcl} M_{k+1} & = & M_k + I.\overrightarrow{tr_r} \\ y_k & = & \varphi(M_k) \end{array} \tag{2.3.1}$$

donde $I.\overrightarrow{tr_r}$ se definen como en la Definición 8.

Definición 10 (Determinismo)**.** Una IPN es determinística, si dada una secuencia de transiciones $\sigma_i = tr_1 \cdots tr_k \cdots$ tal que $M_0 \lceil \sigma_i > M_k$, $M_0 \lceil \sigma_i > M_k'$ implica que $M_k = M_k'$. es decir, el marcado alcanzado después del disparo de una secuencia de transiciones es única, [23].

El lenguaje de disparo de una IPN, se define como:

Definición 11 (Lenguaje de Disparo de una IPN)**.** [24]. Sea Q una IPN y $M_0 \lceil \sigma_i > M_k$. El conjunto de todas secuencias de disparo se llama lenguaje de disparo. $\mathcal{L}_F(Q) = \Big\{ \sigma | \sigma = tr_1 \cdots tr_k \wedge M_0 \stackrel{tr_1}{\to} \cdots M_w \stackrel{tr_k}{\to} M_k \mid M_0, \cdots, M_w \in R(N) \Big\}$.

Por lo tanto, σ, es una secuencia ordenada en una línea de tiempo; ya que cada transición sucede en un instante τ_i.

Es de notar que la evolución de una IPN podría ser no-determinista si φ es no isomórfica. Por ejemplo, una evolución no-determinista ocurriría si existe al menos un lugar con al menos dos transiciones con la misma etiqueta que conducen a dos lugares diferentes. Sin embargo, el lenguaje generado por muchos sistemas (ver [2]) es representado por un autómata no-determinístico. Pero para algunas aplicaciones, como el diagnóstico, el no-determinismo es una característica no deseable.

2.3.1. Red de Petri Coloreada (CPN)

Otra extensión de las PN son las CPN, este tipo de redes son muy importantes porque permiten construir modelos más compactos y paramétricos, lo cual facilita el proceso de mantenimiento y actualización de los modelos. Además presentan otro tipo de ventajas, ya que existe un conjunto de aspectos asociados a las entidades que fluyen entre los elementos del sistema a modelar, identificar o diagnosticar. Las CPN permiten formalizar tanto los atributos o características de los objetos que fluyen en el sistema, como las propiedades que deben tener para que cierto evento pueda suceder.

Antes de presentar la definición de una CPN, se presentan de manera formal la definición de Multiconjunto:

Definición 12 (Multiconjunto "*bg*")**.** Un *bg* sobre un conjunto no vacío A, es una función, tal que: $bg : A \to \mathbb{N}$, denotado como $\sum_{a \in A} bg\,(a) \times a$. $Bag\,(A)$, es el conjunto de todos los *bg* sobre A.

Ejemplo 3. Sea $A = \{a, b, c, d\}$, un *bg* sobre A es $bg_1 = \{a, a, b, b, b, d\}$ o escrito como $bg_1 = 2 \times a + 3 \times b + d$.

2.3.1.1. Representación gráfica de una CPN.

Una CPN se representa gráficamente como en la Figura 2.3.1, para un proceso de maquinado. El proceso consta de dos robots R1 y R2. R1 recoge y transporta la pieza A, hacia el área de maquinado de la cual se genera el Producto1; de igual forma para el Producto 2, pero con el robot 2 y la pieza B.

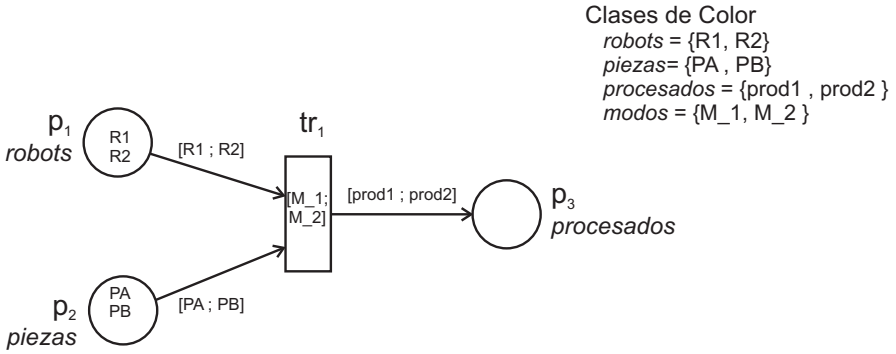


Figura 2.3.1: Ejemplo de representación de una CPN

Como se puede observar en la Figura 2.3.1, existen cuatro conjuntos de clases de color; la asignación de los colores a cada lugar es: $cd(p_1) = \{R1, R2\}$, $cd(p_2) = \{PA, Pb\}$ y $cd(p_3) = \{prod1, prod2\}$ y a la transición es: $cd(tr_1) = \{M\_1, M\_2\}$, donde M_1 y M_2 son modos de funcionamiento para la transición. tr_1 está habilitada en M_1 ($tr_1(M\_1)$); si $\exists\,(m(p_1) = \langle R1\rangle \wedge m(p_2) = \langle PA\rangle)$, si $tr_1(M\_1)$ es disparada entonces $m(p_3) = \langle prod1\rangle$ es alcanzada.

Se puede concluir que las transiciones de una CPN, se habilitan y/o disparan en una ocurrencia específica que se denotará como β_i, el conjunto de ocurrencias es una clase de color, contemplada dentro del conjunto de clases de color C; que en el ejemplo de la Figura 2.3.1 es la clase *modos*.

A cada lugar se le asigna una clase de color o un conjunto de clases de color. Las clases de color asignadas a cada lugar se representan en letra *cursiva* cerca al símbolo del lugar. Los arcos son vectores, con las marcas de color como pesos.

Definición 13 (Red de Petri Coloreada (CPN))**.** [25]. Una CPN es una estructura denotada como: $CN = (P, TR, C, cd, Pre, Post, M_0)$; donde: P, TR y M_0 se definen como en la Definición 7; $C = \{clase_1, \cdots, clase_{ncc}\}$ es el conjunto de clases de color de cardinalidad ncc; $cd : P \cup TR \to C$ es la asignación del dominio color; $Pre, Post \in \beta^{|P|\times|TR|}$ son las matrices de Pre y $Post$ de incidencia; $Pre\,[p, tr] : cd\,(tr) \to Bag\,(cd\,(p))$[1] y $Post\,[p, tr] : cd\,(tr) \to Bag\,(cd\,(p))$ son asignaciones para cada par $(p, t) \in P \times TR$. La representación de las entradas de las matrices Pre y $Post$ será por vectores. El marcado de un lugar es un vector, tal que $m\,[p] \to Bag\,(cd\,(p))$ para cada $p \in P$; m_0 es el marcado inicial.

$Pre\,[p, tr] : cd\,(tr) \to Bag\,(cd\,(p))$ define para cada ocurrencia $\beta_i \in cd\,(tr)$ el multiconjunto $Bag\,(cd\,(p))$ que identifica las marcas a ser removidas de p, cuando se dispara tr bajo la ocurrencia β_i; es decir, $Pre\,[p, tr]\,(\beta_i) \in Bag\,(cd\,(p))$. De manera similar $Post\,[p, tr] : cd\,(tr) \to Bag\,(cd\,(p))$ define para cada $\beta_i \in cd\,(tr)$ el multiconjunto $Bag\,(cd\,(p))$que identifica las marcas a ser adicionadas en p cuando dispara tr bajo la ocurrencia β_i; es decir, $Post\,[p, tr]\,(\beta_i) \in Bag\,(cd\,(p))$.

Una transición $tr \in TR$, está habilitada con respecto a β_i, a la marca M, sii $M \geq Pre\,[\bullet tr]\,(\beta_i)$. La transición habilitada tr puede ser disparada alcanzando una nueva marca $M^{'} = M + Post\,[\bullet tr]\,(\beta_i) - Pre\,[\bullet tr]\,(\beta_i)$, $M^{'} = M + C\,[\bullet tr]\,(\beta_i)$.

[1] $cd\,(tr)$ define la asignación de colores $\beta_i \in C$ a cada transición. $cd\,(p)$ define la asignación de colores a cada lugar.

2.4. Procesos estocásticos

En un proceso industrial interactúan materiales, máquinas, mano de obra, mediciones, medio ambiente y métodos de producción que no permanecen constantes de un ciclo de producción a otro. Esto hace los procesos variables.

La variabilidad interfiere con el proceso de identificación, puesto que los estados pueden variar de un ciclo a otro; por lo tanto, es necesario tener un criterio para determinar cuándo el proceso de identificación se ha estabilizado y se pueda concluir que el modelo resultante sea el que represente de manera eficiente el sistema.

Para lograr este objetivo es necesario realizar muchas observaciones al funcionamiento del sistema, con el objeto de tener información suficiente para aplicar conceptos estadísticos, [26], [27], [28].

Entre estos conceptos se destacan: La estimación de parámetros; test de bondad de ajuste con el objeto de establecer la distribución de los datos; tamaño de muestra del estudio, entre otros. Estos conceptos se explican a continuación:

2.4.1. Estimación Estadística

Para estimar parámetros de una población cualquiera, por ejemplo su media poblacional, se toma una muestra aleatoria de dicha población y se estiman los parámetros muestrales, lo cual conlleva a un error de aproximación.

Existen dos tipos de estimaciones, la estimación puntual y la estimación por intervalo.

2.4.1.1. Estimación Puntual

Una estimación puntual del valor de un parámetro poblacional desconocido (como puede ser la media μ, o la desviación estándar σ), es un número que se utiliza para aproximar el verdadero valor de dicho parámetro poblacional. A fin de realizar tal estimación, se toma una muestra de la población y se calcula el parámetro muestral asociado ($\overline{X}$ para la media, S para la desviación estándar, etc.). El valor de este parámetro muestral será la estimación puntual del parámetro poblacional.

Propiedades debe cumplir todo buen estimador:

- Insesgado: Un estimador es insesgado cuando la media de su distribución muestral asociada coincide con la media de la población. Esto ocurre, por ejemplo, con el estimador $\overline{X}$, ya que $\mu_{\overline{X}} = \mu$
- De varianza mínima: La variabilidad de un estimador viene determinada por el cuadrado de su desviación estándar. En el caso del estimador $\overline{X}$, su desviación estándar es $\sigma_{\overline{X}} = \frac{\sigma}{\sqrt{n}}$, también llamada error estándar de μ.

2.4.1.2. Estimación por intervalo

Dada una población X, que sigue una distribución cualquiera con media μ y desviación estándar σ, se sabe por el teorema central del limite, TCL, que, para valores grandes de n, la media muestral $\overline{X}$ sigue una distribución aproximadamente normal con media $\mu_{\overline{X}} = \mu$ y desviación estándar $\sigma_{\overline{X}} = \frac{\sigma}{\sqrt{n}}$.

Por otra parte, el Teorema de Chebyshev dice que en una distribución normal, aproximadamente un 95 % de los datos están situados a una distancia inferior a 1.96 desviaciones estándar de la media. De lo anterior se deduce que: $P\left(\mu - 1{,}96\sigma_{\overline{x}} \leq \overline{x} \leq \mu + 1{,}96\sigma_{\overline{x}}\right) = 0{,}95$.

Por tanto, ésta última fórmula genera un intervalo de valores, tal que la probabilidad de que la media de la población μ esté contenida en él, es de 0,95. Este tipo de intervalos se llaman intervalos de confianza de un parámetro poblacional.

El nivel de confianza $(1-\alpha)$, del intervalo, es la probabilidad de que éste contenga al parámetro poblacional.

2.4.2. Intervalos de confianza.

Intervalo de confianza para μ con σ conocida. Dada una población X, de la cual no se conoce su distribución ni su media; pero si la desviación estándar poblacional, $X \sim (\mu, \sigma)$, la media muestral $\overline{X}$, por el TCL, se aproxima a la distribución normal: $\overline{X} \sim \left(\mu_{\overline{X}} = \mu, \sigma_{\overline{X}} = \frac{\sigma}{\sqrt{n}}\right)$. Por lo tanto, el intervalo de confianza del 95 % para la media de la población μ es $\left(\overline{X} \pm z_{\alpha/2} * \frac{\sigma}{\sqrt{n}}\right)$.

El error máximo de estimación es la mitad de la longitud del intervalo, $E = z_{\alpha/2} * \frac{\sigma}{\sqrt{n}}$.

Si se extraen varias muestras del mismo tamaño y se calcula un intervalo de confianza para cada muestra, el 95 % de todos los intervalos van a incluir la media poblacional.

Intervalo de confianza para μ con σ desconocida. Dada una población X, con su media y su desviación estándar desconocidas, $X \sim (\mu, \sigma)$. Aunque n sea $n < 30$, la media muestral $\overline{X}$, sigue una distribución normal $\overline{X} \sim \left(\mu_{\overline{X}} = \mu, S_{\overline{X}} = \frac{S}{\sqrt{n}}\right)$.

Para estimar la desviación estándar poblacional σ se utiliza la desviación estándar muestral S.

Por lo tanto, el intervalo de confianza del 95 % para la media μ, será: $\left(\overline{X} \pm t\,(n-1, \alpha/2) * \frac{S}{\sqrt{n}}\right)$. Se utiliza la distribución t-Student porque la desviación estándar poblacional σ, es desconocida.

2.4.3. Procedimiento para estimación de parámetros.

Para estimar los parámetros se propone el método de máxima verosimilitud. Este método es bastante intuitivo pues se basa en suponer que siempre ocurre lo más probable, y por ello estima el parámetro mediante aquel valor que hace más probable la muestra.

Definición 14 (Función de Máxima Verosimilitud)**.** Se define la función de verosimilitud, para variables continuas, de un muestreo aleatorio simple, (m.a.s.) de tamaño n, como $L(\theta, X_1 \cdots X_n) = f_\theta(X_1 \cdots X_n) = \prod_{i=1}^{n} f_\theta(x_i)$, donde θ es un parámetro desconocido de la variable aleatoria.

Fijados $X_1 \cdots X_n$, valores de la muestra conocidos, la función de verosimilitud sólo depende de θ.

Definición 15 (Estimador de máxima verosimilitud, EMV)**.** Sea $L(\theta, X_1 \cdots X_n)$ la función de verosimilitud para un m.a.s. $X_1 \cdots X_n$, se define EMV de θ, $\tilde{\theta}_{MV}$, al valor de θ que maximiza $L(\theta, X_1 \cdots X_n)$.

El EMV es aquel valor que hace máxima la probabilidad de que se dé la muestra obtenida. En la práctica si la función L es derivable respecto al parámetro, se puede calcular la primera derivada respecto a ese parámetro, se iguala a 0 y se despeja el valor; comprobando posteriormente que la solución obtenida corresponde a un máximo mediante la segunda derivada. Suele ser útil aplicar previamente el logaritmo neperiano; ya que al ser el logaritmo neperiano una función creciente, la función de verosimilitud alcanzará el máximo para el mismo punto que su logaritmo. Aplicando el método de máxima verosimilitud se pueden encontrar los parámetros para las posibles distribuciones que modelen los datos obtenidos en el proceso de identificación.

Definición 16 (Distribución Uniforme)**.** Se dice que una variable aleatoria continua X sigue una distribución uniforme en el intervalo (a,b), si su función de densidad está definida como $f(x) = \begin{cases} \frac{1}{b-a} & sí \quad a < x < b \\ 0 & c.c \end{cases}$. Se denota por $X \sim U(a, b)$.

Sus parámetros se estiman como: $U(0, \tilde{\theta})$; $\tilde{\theta} = 2\bar{X}$.

Definición 17 (Distribución Exponencial)**.** Se dice que la variable aleatoria continua X, sigue una distribución exponencial de parámetro λ , con $\lambda > 0$ si su función de densidad es $f(x) = \begin{cases} \lambda e^{-\lambda x} & sí \quad x > 0 \\ 0 & c.c \end{cases}$. Se denota por $X \backsim Exp(\lambda)$.

El parámetro se estiman como: $\tilde{\lambda} = \frac{1}{\bar{X}}$.

Definición 18 (Distribución Normal)**.** Se dice que una variable aleatoria X, sigue una distribución normal de parámetros $\mu \in \mathbb{R}$ y $\sigma^2 > 0$, si su función de densidad es:

$f(x) = \frac{1}{\sqrt{2\pi}\sigma} exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$, $x \in \mathbb{R}$. Se denota por $X \backsim N(\mu, \sigma^2)$.

Sus parámetros se estiman como: $\tilde{\mu} = \bar{X}$; $\tilde{\sigma}^2 = S_x^2 = \frac{1}{n-1} \sum \left(X_i - \bar{X}\right)^2$ (Cuasivarianza muestral).

2.4.4. Comportamiento de los datos.

A partir del comportamiento de los datos es posible inferir qué tipo de distribución siguen los datos de la población con base en las pruebas de bondad de ajuste. Estas pruebas permiten verificar que la población de la cual proviene una muestra, tiene una distribución especificada o supuesta.

La pruebas de bondad de ajuste se utilizan para comparar el ajuste de distribuciones, selección de modelos y determinar qué tan bien se ajusta a los datos. El objetivo de la prueba es medir la "distancia" entre los datos y la distribución que se está probando y comparar esa distancia a un cierto “valor crítico”. Si la distancia, llamada “estadístico de la prueba” es menor que el valor crítico, el ajuste se considera bueno.

La lógica de la aplicación de diversas pruebas de bondad de ajuste es el mismo; sin embargo, difieren en la forma de calcular los estadísticos de prueba y los valores críticos. Los estadísticos se definen generalmente como una función de los datos de la muestra y de la función de distribución acumulada teórica (ajustada).

Los valores críticos dependen del tamaño de la muestra y el nivel de significación elegido. El nivel de significación es la probabilidad de rechazar una distribución ajustada (como si fuera un mal ajuste) cuando en realidad es una buena opción. El nivel de significación se indica con la letra griega α y los niveles más utilizados son 0,05 y 0,01, [28].

Como contrastes de bondad de ajuste se destacan: Test de Kolmogorov-Smirnov, χ^2 de bondad de ajuste, prueba de Anderson Darling y contrastes específicos para la distribución normal como gráficos de normalidad y test de Shapiro- Wilks.

2.4.4.1. Test de Kolmogorov-Smirnov de bondad de ajuste

Si se tiene una muestra aleatoria simple (m.a.s.) de una variable aleatoria (v.a.) X; $X_1, \cdots, X_n$, donde n es el número de muestras, se define la función de distribución empírica de la muestra, $F_n(x)$, como la proporción de observaciones en la muestra menores o iguales que x, $\forall x \in \mathbb{R}$. $F_n(x)$ es una función escalonada y no decreciente, construida a partir de la muestra, de forma que, en cada

observación muestral, da un salto de magnitud igual a la fracción de datos iguales a ese valor (cuando no hay repeticiones se trata de saltos de amplitud $1/n$).

Para calcular $F_n(x)$, se ordena la muestra de menor a mayor y se define como:

$$F_n(x) = \begin{cases} 0 & sí \quad x < x_1 \\ \frac{card(x_j < x)}{n} & sí \quad x_i < x < x_{i+1}, i = 1 \cdots n \\ 1 & sí \quad x_n < x \end{cases}$$

donde $card(x_j < x)$ es el número de observaciones muestrales menores o iguales que x.

El test de Kolmogorov-Smirnov de bondad de ajuste se basa en comparar la función de distribución empírica de la muestra con la función de distribución que se propone para describir los datos, $F_0(x)$. Es válido únicamente para variables aleatorias continuas.

Entonces, sea X una variable aleatoria continua con función de distribución $F_0(x)$.

Se plantea el contraste:

- $H_0 : F(x) = F_0(x)$, para todo x.
- $H_1 : F(x) \neq F_0(x)$, para algún x.

Es decir, se propone un modelo de distribución para los datos, $F_0(x)$, y como alternativa que los datos no se distribuyen según este modelo.

Se calcula un estadístico D, el cual mide la discrepancia máxima entre la función de distribución empírica y la propuesta en H_0.

$D(X_1, \cdots, X_n) = sup_x |F_n(x) - F_0(x)|$. Con $F_n(x)$ como la función de distribución empírica de la muestra.

La región crítica es $D(X_1, \cdots, X_n) \geq d(n, (1-\alpha))$; $d(n, (1-\alpha))$ es el $p - valor$ del contraste con una muestra de tamaño n y un

nivel de significancia α: $p - valor_\alpha$. Se acepta H_0 sí el $p - valor$ (D) obtenido es menor que el del nivel de significación elegido para realizar el test; es decir se acepta si: $p - valor \leq p - valor_\alpha$.

En la mayoría de los casos al utilizar el estadístico de Kolmogorov-Smirnov es necesario estimar los parámetros desconocidos que caracterizan a la distribución teórica. Cuándo no se tiene esta información, éstos se estiman por el método de máxima verosimilitud.

2.4.4.2. El test de Kolmogoroff-Smirnoff-Lilliefors para normalidad (contraste KSL)

Si la distribución que se desea ajustar es una normal, hay que estimar la media y la desviación típica. En este caso, los parámetros se estiman por máxima verosimilitud y la distribución del estadístico cambia, de donde $\tilde{\mu} = \bar{X}$; $\tilde{\sigma}^2 = S_x^2 = \frac{1}{n-1} \sum \left(X_i - \bar{X}\right)^2$ (Cuasivarianza muestral).

Entonces $D\left(X_1, \cdots, X_n\right) = sup_x \left|F_n\left(x\right) - P\left(\frac{X-\bar{X}}{S_x}\right)\right|$, donde $P(z)$ es la función de distribución de una normal estándar.

El estadístico D representa la máxima discrepancia, en vertical, entre la función de distribución empírica y la función de distribución de la normal ajustada (esto es, de la normal con media y varianza estimadas). La distribución de este estadístico fue tabulada por Lilliefors (contraste KSL) y, por tanto, es con respecto a esta tabulación (y no con respecto a la tabla de Kolmogoroff-Smirnoff) como se debe juzgar la significación del valor obtenido para este estadístico.

2.4.4.3. Test de Shapiro-Wilks

En general, la prueba de Shapiro-Wilks es más adecuada para contrastar la normalidad de las observaciones que el test de Kolmogorov-

Smirnov.

Sea X una variable aleatoria con función de distribución F. Se plantea el contraste:

- $H_0 : X \sim Normal$;
- $H_1 : X \nsim Normal$.

El estadístico de Shapiro-Wilks, que se denota por W, mide el ajuste de la muestra, representada en el papel probabilístico normal (P-P plot), a una recta. Se rechaza la normalidad cuando el ajuste lineal es malo, lo que se reflejaría en valores pequeños del estadístico. Así la región crítica de este contraste es R.C.: $W \leq w\,(n, \alpha)$.

$W = \frac{\left(\sum_{i=1}^{n} a_i x_{(i)}\right)^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$, donde:

- $x_{(i)}$es el número que ocupa la *i-ésima* posición de la muestra.
- $\bar{x}$, es la media de x.
- Las constantes a_i se calculan a partir de:

  $(a_1 \cdots a_n) = \frac{mV^{-1}}{(m^T V^{-1} V^{-1} m)^{1/2}}$, donde $m = (m_1 \cdots m_n)^T$, siendo $m_1 \cdots m_n$, los valores medios del estadístico ordenado de variables aleatorias independientes e idénticamente distribuidas, muestreadas de distribuciones normales. V es la matriz de covarianzas de ese estadístico de orden.

Se rechaza H_0 sí el $p - valor$obtenido es menor que el del nivel de significación elegido para realizar el test.

2.4.4.4. Prueba χ^2

Esta prueba es aplicable para variables aleatorias discretas o continuas. Sea una muestra aleatoria de tamaño n tomada de una población con una distribución especificada $f_0(x)$ que es de interés verificar. Los datos se deben presentar en k clases.

El estadístico de contraste será: $\chi^2 = \sum_{l=1}^{k} \frac{(O_l - E_l)^2}{E_l}$; donde O_l es la frecuencia observada en la clase l, E_l es la frecuencia esperada (teórica) en la clase l.

Cuanto menor sean el valor del estadístico χ^2, más coherentes serán las observaciones obtenidas con los valores esperados. Por el contrario, valores grandes de este estadístico indicarán falta de concordancia entre las observaciones y lo esperado. En este tipo de contraste se suele rechazar la hipótesis nula cuando el estadístico es mayor que un determinado valor crítico.

2.4.5. Tamaño de muestra

El tamaño de muestra está en función de: $tamaño\ muestra(n) = f(\alpha, D, S^2)$. Específicamente n puede ser calculado, a partir de la ecuación propuesta por [29], como:

$$n = \left(st_{\alpha/2}\right)^2 \times S^2/d^2 \tag{2.4.1}$$

en el cual el nivel de significancia α se define como: $Prob\left(\left|\bar{X} - \mu\right| \geq d \times \mu\right) = \alpha$, donde d es la máxima diferencia aceptada entre la media poblacional y la media muestral, es decir el error relativo; S^2 es la cuasi-varianza muestral. α es un riesgo muy pequeño al que se está dispuesto a aceptar, generalmente α se establece en 0.05, [28]. El nivel de significancia puede ser lateral o bilateral, sí es bilateral la probabilidad se halla a partir de $\alpha/2$.

Cómo hallar el estadístico para $\alpha/2$ $\left(st_{\alpha/2}\right)$. Se desea encontrar $f(\alpha)$ con el objeto de establecer un comportamiento de media

muestral respecto a la media poblacional; por lo tanto, se puede aplicar el teorema del límite central. Entonces, si X es la media de una muestra aleatoria de tamaño n extraída de una población que tiene media μ y varianza finita σ^2, la variable aleatoria $Z = \frac{\bar{X}-\mu}{\sigma/\sqrt{n}}$ se distribuye normalmente con media 0 y varianza 1, cuando n tiende al infinito. Es decir la media muestral de una variable aleatoria se distribuye como una normal con parámetros μ y σ^2/n, si n es lo suficientemente grande, ($n \geq 30$).

Cuando no se conoce σ^2, ésta es estimada a partir de la cuasivarianza muestral S^2; por lo tanto se sustituye σ^2 por S^2; pero al realizar la sustitución se define una nueva variable $t = \frac{\bar{X}-\mu}{S/\sqrt{n}}$.

La función de probabilidad t es similar a la normal pero es más dispersa ya que el uso de S introduce más incertidumbre. El grado de dispersión depende del valor de n y se expresa como el parámetro de la distribución t, sus grados de libertad.

Los valores de $t_{\alpha/2}$, se calculan de manera similar que la distribución normal (área bajo la curva).

Aunque la distribución t, exige que la población de la cual proviene la media sea normal, la estimación de las probabilidades se considera un buen estimador en casos en los cuales la muestra es grande; independiente del tipo de distribución de la variable aleatoria.

Suponiendo un valor de $n = 30$; sí $\alpha = 0{,}05$, entonces $t_{\alpha/2} = 2{,}042$; sí $1 - \beta = 0{,}85$; entonces $t_{\beta} = 1{,}055$.

3 Estado del Arte

Como se mencionó en la introducción, el alcance general de la presente propuesta de investigación es la identificación y diagnóstico de fallos en Sistemas de Eventos Discretos (SED).

Al rededor de esta temática han surgido numerosas líneas de investigación, diferenciadas por las características del sistema que identifican o diagnostican, por la herramienta de modelado, por el algoritmo aplicado o por el tipo de compilación de la información, entre otras.

Con base en la motivación del trabajo presentada en la introducción, para alcanzar los objetivos de la tesis se hace necesario abordan diferentes temáticas como modelado e identificación, entre otras; que a su vez son por si solas líneas de investigación. Entonces, teniendo en cuenta este alcance y en aras de tener una mejor organización, a continuación se presenta un análisis de las principales contribuciones en cada temática.

3.1. Respecto a Modelado

El problema de modelado de un sistema dinámico se puede ver, de acuerdo a [30] como: "Dado un sistema dinámico S, con un conjunto de preguntas B sobre su comportamiento, encontrar una representación M que ayude a responder a las preguntas dadas; entonces, M se llama el modelo de S". Desde este enfoque se precisa

que dado que el modelo se utiliza para resolver un problema, éste no necesariamente es único.

El modelado de SED, ha sido estudiado desde hace muchos años, tanto por la academia como la industria; puesto que la evolución exponencial de las tecnologías industriales de computación, comunicación y de sensores han traído nuevos sistemas dinámicos, complejos y flexibles; que se caracterizan por acontecimientos asincrónicos de eventos discretos, algunos controlados y otros no, algunos observados por sensores y otros no, algunos aparecen de forma automática a partir de los procesos físicos remotos y algunos se generan manualmente por los usuarios. Los SED tienen un comportamiento que se caracteriza por una secuencia finita o infinita de estados delimitados por eventos que ocurren de manera asíncrona o síncrona, [10].

Varias propuestas de modelado se han dado, usando diferentes enfoques: autómatas, máquinas se estado finito, Redes de Petri (PN) y cadenas de Markov. Los problemas que han solucionado este tipo de modelos están relacionados con: *i)* Supervisión y diagnóstico de procesos, para decidir si el desempeño del sistema se encuentra dentro de límites prescritos o si han ocurrido fallos; *ii)* Control supervisor, que implica la evaluación cualitativa de las condiciones de funcionamiento actuales de un sistema con el fin de evitar puntos de operación críticos y de cumplir con los objetivos de control y *iii)* Simulación, para realizar procesos de identificación de SED. Las PN han sido reconocidas como un modelo apropiado para describir SED, [31, 32, 17], particularmente cuando se trata de sistemas asíncronos, [33]. Las PN incorporan la noción de estado distribuido y de reglas que permiten pasar de un estado a otro, lo cual captura tanto el comportamiento estático como el comportamiento dinámico de los sistemas reales.

Teniendo en cuenta que un sistema en lazo cerrado es una interrelación de señales, se han desarrollado propuestas diferentes propues-

tas de modelado bajo PN, una de ellas se ha denominado modelos de sistemas de condiciones, estos modelos se representan mediante PN con entradas y salidas explícitas llamadas condiciones, [34]. Esas representaciones permiten la interacción entre subsistemas así como la interacción de la planta con el controlador, [35].

Sreenivas, et al. [36] definen una clase de sistemas dinámicos de eventos discretos (DEDS), en tiempo continuo, llamados sistemas condición/evento (C/E) y también definen modelos para estos sistemas basados en una extensión de las PN y los llaman C/E PNs (por sus siglas en inglés), con dos tipos de valores de señales discretas de entrada - salida (E/S): condiciones de señales y señales de eventos; las señales de condición E/S llevan información del estado y las señales de evento E/S llevan información de transición de estado. La idea general es modelar un sistema como un conjunto de módulos con un comportamiento dinámico particular y su interconexión a partir de sus señales, [37]. Las condiciones y los eventos de entrada pueden ser conectados con algunas transiciones dentro del módulo por condiciones y arcos de eventos. Los módulos de lugares pueden ser conectados a las condiciones de salida por condiciones de arco, y las transiciones a los eventos de salida por eventos de arco. Estos conceptos proveen una base para un enfoque de composición con el objeto de construir modelos de sistemas grandes a partir de componentes pequeños, [37]. Este tipo de propuestas de modelado se han desarrollado básicamente para ser utilizadas en verificación de propiedades como: alcanzabilidad, acotabilidad y vivacidad; aunque [38], presenta una propuesta basada en sistemas de condición para realizar diagnóstico de fallos.

Otra forma que ha sido ampliamente estudiada en el modelado de SED utilizando PN, son las redes de Petri temporizadas (t-PN), que fueron introducidas por [39, 40]. Las t-PN son PN extendidas con tiempo en las transiciones, poseen la expresividad necesaria para modelar sistemas en lazo cerrado y presentan beneficios para

realizar análisis de verificación de propiedades, [41, 42]. Para modelar sistemas complejos se han realizado propuestas de división en subsistemas aunque la composición de PN no ha sido trivial. En [43], los autores presentan una propuesta de composición siempre que las transiciones sean sincronizadas. En [44] se presenta una solución que facilita la especificación de los sistemas dependientes del tiempo, en un planteamiento de composición de componentes modelados como t-PN. En la mayor parte de las t-RdP las transiciones tienen un tiempo asociado, por lo que la evolución queda determinada por el disparo de las transiciones. Una transición se puede disparar, si el tiempo está dentro de su intervalo de disparo estático, [45]. En [46] se propone una PN conducida por eventos en lugar del paso del tiempo, denominada EDPNs, (por sus siglas en inglés). Las EDPNs emplean arcos inhibidores, arcos de ensayo y lugares con capacidad finita; como el tiempo es considerado un estado (marcado) del sistema (PN), su paso se corresponde con el juego de marcas que genera un árbol de alcanzabilidad. Al introducir arcos inhibidores existen muchas dificultades para utilizar las herramientas de análisis cualitativo de las PN.

Por otro lado, como la transición de estados en un SED es un evento [11], éste se puede modelar a partir de su lenguaje. El lenguaje que modela el comportamiento de un sistema se puede representar por su expresión regular, [13], que es una síntesis de todas las combinaciones válidas de eventos. En [11] un SED es modelado como un autómata y su comportamiento es representado por su lenguaje; pero este tipo de modelos pueden presentar inconvenientes en sistemas con gran cantidad de dispositivos debido a la explosión combinatoria de estados, suele ser más conveniente emplear formalismos con mayor capacidad de condensación de estados, como las PNs o las máquinas de estados finitos.

Una PN tienen un poder más descriptivo que las máquinas de estados finitos en el sentido de que el conjunto de lenguaje de la

PN es un superconjunto de los lenguajes regulares, y permite un modelado más conciso, [47, 48]. Algunas propuestas para modelar SED bajo el lenguaje PN se pueden ver en: [49, 50, 51, 23] o para encontrar ciclos en sistemas de estados repetitivos en [52].

Respecto a la utilización de PN para identificación y diagnóstico varias propuestas de modelado se han generado utilizando PN extendidas como PN interpretadas (IPN), [53, 54, 55], PN temporizadas [56], algunas de estas propuestas se presentan a continuación.

3.2. Respecto a identificación

El problema de identificación en SED consiste en obtener un modelo aproximado de un sistema desconocido, expresado en un formalismo adecuado, a partir del comportamiento observado representado como secuencias de datos finitos de entrada - salida (E/S).

En el contexto de los SED, los datos de entrada se dan generalmente en términos de descripciones de comportamiento (por ejemplo, la transición de estados o un lenguaje). El conjunto de secuencias de comportamiento pueden ser fijas o se pueden incrementar en el curso del proceso de identificación mediante la realización de nuevos experimentos.

El problema de identificación pretende abordar dos aspectos principales. En primer lugar, decidir si para la especificación de comportamiento dada existe un SED, por ejemplo, un tipo de PN, que genere el comportamiento especificado. En segundo lugar, proporcionar un procedimiento constructivo para determinar tal modelo del SED.

Los métodos de identificación pueden ser comparados, por su complejidad en términos de las características del sistema, el proceso

de identificación y el modelo o algoritmo aplicado, [57]; así como también por los objetivos del modelo de identificación.

Los métodos más simples están basados a partir de datos de E/S, donde los estados concurrentes no son identificados y la ejecución se realiza off-line, [57]. Si el proceso de identificación es incremental, el algoritmo debe ser ejecutado on-line; pero la complejidad puede llegar a ser exponencial o polinomial, dependiendo del enfoque usado para el modelado.

Existe una amplia literatura relacionada con la identificación de SED. Las primeras contribuciones aparecieron en los años sesenta y se han formulado en términos de identificación de lenguajes, cuyas soluciones se designan comúnmente como técnicas de aprendizaje, [57], apareciendo después otras contribuciones en el contexto de Autómatas Máquinas de Estado Finito (FSM), Moore o Mealy y PN.

Actualmente, las técnicas más utilizada son las PN y los Autómatas, [33]. Sin embargo, el formalismo de las PN es muy reconocido como un modelo adecuado para describir la estructura y el dinamismo en sistemas complejos por su particularidad de evitar la explosión de estados, [58].

3.2.1. Enfoque bajo Autómatas

Se consideran modelos identificados como autómatas, autónomos, finitos, no - determinísticos, (NDAA), el objetivo de este enfoque es construir un modelo que se aproxime al lenguaje del sistema original. El lenguaje observado del SED en lazo cerrado es transferido a una máquina de estado finito por un algoritmo de identificación apropiado. En [2] y en [59] un (NDAA), sin señales de salida, es elegido como modelo apropiado para reproducir el lenguaje observado. Este enfoque identifica un sistema en lazo cerrado basado en

la interacción del controlador con comportamiento determinista y el proceso físico con comportamiento no-determinístico; combinando eventos externos e internos; además se presenta una propuesta de división del sistema en subsistemas más pequeños, [60]. Una evolución de este método es presentado en [61], el cual trata con la identificación de sistemas con eventos temporizados, el autor propone un método para identificar con eventos no solo ordenados en el tiempo sino que generan una correcta temporización de eventos de entrada y de salida para la completa descripción del sistema en el contexto de los autómatas temporizados. Esta tendencia presenta algunas desventajas por la no concurrencia de estados y por la explosión combinatoria de estados en sistemas complejos.

En [62] se presentan algunas aspectos relacionados con la aplicación del algoritmo de [2] a un sistema real.

3.2.2. Enfoque bajo PN

Bajo el formalismo de la PN, se han realizado diferentes formulaciones de problemas desde principios de los años noventa y varias soluciones se han dado en muy distintos entornos. En [57] se ha realizado una interesante clasificación de los métodos de identificación. Respecto a PN se definen algunas tendencias:

Identificación Progresiva Este enfoque propone identificar modelos de SED a partir de observaciones de señales de E/S. Un algoritmo que trabaja on-line, recibe una secuencia de señales de salida y calcula un modelo de IPN que describe el comportamiento desconocido del SED, [63]; cada vez que un ciclo es detectado el modelo previo calculado es actualizado. Este enfoque considera que cada entrada al sistema es reflejada en la salida, esto significa que si incluso un sistema no está completamente instrumentado, la

información provista por lo sensores es suficiente para detectar un cambio de estado.

En [64] los autores presentan una evolución al anterior método, este trabaja con sistemas concurrentes, detectando transiciones concurrentes. El modelo identificado es usado principalmente para diagnóstico estructural.

En [65] los autores dirigen el problema de encontrar las secuencias de disparo de transiciones de menor costo para una IPN, basándose en la observación de secuencias etiquetadas. Las PN poseen transiciones observables, las cuales se asocian posiblemente a una única etiqueta y transiciones inobservables cuyo disparo no genera alguna etiqueta observable, ellos asumen que cada transición en una IPN está asociada con un costo no negativo que captura la probabilidad (por ejemplo, en términos de la carga de trabajo o la cantidad de energía necesaria para ejecutar una determinada transición); dada la observación de una secuencia de etiquetas, la tarea es encontrar la secuencia de disparo de las transiciones que sea consistente con la secuencia de etiquetas y la estructura de la PN y además sea la de menor costo.

Identificación de Lenguaje con Programación Lineal Entera
En [66] se presenta un enfoque de identificación de una IPN libre[1], a partir del conocimiento de una cadena finita de secuencias de transiciones que representa su lenguaje. La solución se basa en la definición de un conjunto de restricciones lineales enteras, las cuales se solucionan como problemas de programación lineal entera y fuerzan a la IPN a aceptar el lenguaje observado. Muchas extensiones a este trabajo se han realizado en [67, 68].

[1]Redes en las cuales las transiciones pueden ser marcadas con la misma etiqueta y pueden ser habilitadas simultáneamente a partir de un marcado alcanzado.

Otra contribución a este enfoque es dado en [56], que trata con el problema de identificación para sistemas determinísticos como PN temporizadas $\lambda - freee$, a partir de secuencias observadas temporizadas. Este método genera un lenguaje de red mientras que la información temporizada es usada para identificar cadenas de lenguaje que no son aceptadas por la red; por lo tanto, el algoritmo no necesita conocer el lenguaje de la red total. La solución está basada en un problema de programación lineal entera y en la estructura de la red temporizada. Esto permite, junto con la observación de eventos, determinar si una transición temporizada ha expirado. El algoritmo es llevado a cabo a través de dos subrutinas, una para observar el lenguaje on-line y otra para resolver el problema de programación lineal off-line.

Identificación con Programación Lineal Entera Este enfoque es una extensión del trabajo de [66]; pero además de una secuencia de eventos, se utiliza la secuencia de respuesta de salida disponible del SED para hacer la inferencia de un modelo de PN,[69, 70, 20].

El problema de identificación consiste en determinar un conjunto de lugares de cardinalidad m, un conjunto de transiciones de cardinalidad n, así como la función de etiquetado definido como un problema de programación lineal entera.

Identificación de IPN Paramétrica Este enfoque fue presentado en [71], el objetivo es descubrir cómo construir un modelo compacto que pueda mostrar explícitamente un comportamiento descubierto a partir de observaciones del comportamiento del sistema, expresado como una sola secuencia de sus señales de E/S y de cómo los componentes del sistema están relacionados. Las señales de entrada son señales de los sensores y las de salida son las emitidas por un PLC al actuador. El método propuesto tiene un enfoque de

identificación pasiva; que permite construir paso a paso un modelo de IPN en tiempo polinomial, el cual describe en forma detallada el comportamiento reactivo del controlador; para prevenir el crecimiento del modelo en sistemas complejos propone un método de identificación estadística que descubre mediante el análisis de la frecuencia relativa la relación de los cambios de salidas con respecto a los cambios de entrada; tales relaciones se expresan en términos de probabilidad condicional. Por esta razón el enfoque se denomina estadístico.

Analizando estos enfoques se puede decir que:

El enfoque basado NDFA se aplica también a sistemas temporizados, trabaja mejor en sistemas simples, genera modelos nodeterminísticos, a un costo computacional normal pero su principal desventaja es la explosión combinatoria de estados. El inconveniente con el trabajo de Klein [2], es que no permite identificar concurrencia. La segunda desventaja es que la metodología considera que se proporcionan secuencias cíclicas. Para la recolección de tales secuencias, el sistema identificado debe reiniciarse en cada ciclo de producción. Esta suposición no es posible cumplir en muchos casos en los que el sistema no vuelve al estado inicial para el ciclismo, [71]. En la propuesta de Roth [60], los modelos construidos no son adecuados para representar información estructural como el paralelismo dentro de los subsistemas y el intercambio de recursos.

La identificación de lenguajes para ser representados como IPN, no siempre se puede encontrar una solución al problema debido a la cantidad de contra-ejemplos.

En la identificación con programación lineal entera no es fácil establecer los límites del número de lugares ni del número de transiciones, sobretodo cuando se identifican sistemas sin ningún tipo de información previa, además la solución del problema de programación lineal entera no es muy fácil de encontrar.

En la identificación de IPN Paramétrica, la aplicación a sistemas complejos hace que los modelos crezcan cuando existe paralelismo y la aplicación del método de identificación solo se aplica a sistemas en los cuáles existan ciclos repetitivos que se puedan representar estadísticamente.

3.3. Respecto a Diagnóstico de Fallos

El diagnóstico de fallos juega un rol importante en los sistemas industriales ya que permite detectar comportamientos de fallo tan pronto como es posible, con el objeto de evitar daños serios sobre el sistema o lesiones a un operador y además permite disminuir los costos de producción y por consiguiente la conservación del entorno necesario para permanecer en los mercados competitivos, [72].

Bajo el diagnóstico de fallos hay dos tipos de problemas a resolver: Por un lado está el diagnóstico de un fallo y por otro el de diagnosticabilidad. El primer problema implica la definición de un método de diagnóstico el cual, dada una secuencia de eventos generados en un sistema, determine si corresponde a un comportamiento normal o a un comportamiento de fallo y el segundo problema trata con determinar si un sistema es diagnosticable, es decir, establecer si una vez que ha ocurrido un fallo el sistema puede detectar su ocurrencia en un número finito de pasos.

Un fallo es una desviación del sistema de su comportamiento normal o requerido. El diagnóstico de fallos es el proceso de detectar e identificar tales desviaciones del sistema, determinando las causas que lo producen usando la información disponible de las variables del sistema, [73].

El diagnóstico de fallos en SED es un problema que ha sido estudiado desde diferentes enfoques. De acuerdo a [4], el diagnóstico de

fallos tiene como objetivo lograr tres tareas complementarias: La detección del fallo, el aislamiento y la identificación del fallo. La detección es una funcionalidad que decide si el sistema trabaja en condiciones normales o si ha ocurrido un fallo; si ha ocurrido un fallo, el objetivo del aislamiento es localizar el componente(s) del sistema que causa el fallo; y la identificación se relaciona con la definición de la naturaleza específica del fallo (su tamaño, criticidad, importancia, etc).

En el proceso de diagnóstico de fallo se utiliza un diagnosticador que asocia a cada secuencia de eventos observados un estado de diagnóstico; por lo tanto, un diagnosticador ideal es aquel que tiene un exhaustivo y correcto modelo en comportamiento de fallo, lo cual no es posible en sistemas reales complejos.

Este problema ha sido dirigido por muchos investigadores con el objeto de desarrollar nuevos modelos, nuevas propiedades, nuevos algoritmos y eficientes soluciones para el diagnóstico de SED.

Los métodos de diagnóstico de acuerdo a [5], se dividen en tres grupos: Los enfoque basados en reglas, (if then else), los enfoques conducidos por datos (minería de datos o reconocimiento de patrones) y los enfoques basados en modelos. A su vez los basados en modelos, pueden ser divididos en dos grupos: El primer grupo considera modelos que tienen comportamiento libre de fallo así como modelos con comportamiento de fallo y el segundo grupo usa solo modelos libres de fallo.

Respecto a los métodos con comportamiento de fallo se puede argumentar lo siguiente:

El trabajo de Sampath [74, 13], provee una fundamentación formal respecto al diagnóstico de fallos y respecto al análisis de diagnosticabilidad de SED, la cual ha sido la base de muchos enfoques de diagnóstico. Su propuesta está basada en el uso del modelo clásico de autómata para el sistema y el lenguaje generado por ese au-

tómata representa el conjunto de todas las posibles ejecuciones o secuencias de eventos del sistema en funcionamiento normal o de fallo.

También [75] dirige el problema de identificación basado en Autómatas temporizados; el algoritmo identifica y detecta fallos en actuadores en sistemas de baja complejidad.

Como se describió en el estado del arte respecto a modelado, las PN tienen ventajas sobre los autómatas para modelar SED y transfieren esas ventajas al modelar sistemas con comportamiento de fallo.

En [76], se presenta una clasificación de los métodos de diagnóstico basados en PN, con base en la técnica adoptada de diagnóstico o de diagnosticabilidad. Esta clasificación se basa en técnicas algebraicas, enfoque de T-invariante, representación de alcanzabilidad o de marcado de red bajo observación parcial, verificador de red, enfoque de red desarrollada, enfoque de programación lineal entera, entre otras. Algunas de las aportaciones se describen a continuación.

Las PN han sido utilizadas para problemas de diagnóstico a partir de [77], [78] y [79], quienes han presentado propuestas de diagnóstico a partir de la estimación de estados de fallo. En [80] se presenta un enfoque de red desarrollada para diagnóstico asíncrono on-line, su propuesta evita el problema de explosión de estados que típicamente resulta de tener diferentes componentes que interactúan asíncronamente en un sistema distribuido; pero el costo computacional para realizar el diagnóstico on-line es alto. En [81], los autores adaptan y extienden la propuesta de Sampath [74] a diagnóstico on-line de sistemas modelados como PN, resultando un algoritmo de diagnóstico centralizado basado en la noción de "Diagnosticador de PN" donde algunas de las transiciones son etiquetadas por diferentes tipos de eventos inobservables de fallo. Una versión distribuida de este algoritmo centralizado también es

presentado; luego estos autores presentan en [82] dos nuevos algoritmos, uno se extiende al caso de múltiples módulos y el otro con etiquetas de mensajes de tamaño fijo, que mejora significativamente los requisitos de comunicación en tiempo real. Giua y Seatzu ([83]) asumen que los fallos son modelados por transiciones inobservables; pero pueden existir otras transiciones que representan comportamientos legales pero que son inobservables, ellos prueban que todas las posibles secuencias de disparo correspondientes a una observación dada, pueden ser caracterizadas basándose en la noción de marcado básico y en las justificaciones. Para el cálculo del conjunto de marcado básico se usa un árbol de alcanzabilidad básico; Cabasino ([84]) cambia el concepto de marcado básico y solo enumera un subconjunto del espacio de alcanzabilidad, esto genera una caracterización diferente en términos de nociones nuevas y originales de justificaciones y explicaciones mínimas.

El trabajo de [85] considera el sistema modelado como una IPN permitiendo describir el estado del sistema a partir de eventos observados parcialmente; el modelo incluye los fallos que pueden ocurrir.

Dotoli, et al. [86] con el objeto de evitar el rediseño y la redefinición de la estructura del diagnosticador cuando la estructura del sistema cambia, propone una técnica de detección de fallos que trabaja on-line; el diagnosticador espera por un evento inobservable y un algoritmo decide si el comportamiento del sistema es normal o puede exhibir algún posible fallo, para cumplir con este objetivo define problemas de programación lineal entera (ILPP) y provee una secuencia de transiciones inobservables que contienen los fallos que podrían haber ocurrido; el sistema es modelado por una IPN donde los eventos a ser diagnosticados referidos como fallo son modelados como transiciones inobservables y una etiqueta diferente es asociada con cada transición que modela el comportamiento regular. [1] parte de los resultados de [86] y considera un siste-

ma más general en un doble sentido, primero asume una fuente de no-determinismo originado del hecho de que diferentes transiciones observables pueden tener la misma etiqueta y segundo, aplica su técnica de detección de fallos a un sistema distribuido y por último [87] construye un diagnosticador on-line usando IPN para la definición y resolución.

Las PN coloreadas (CPN) también han sido orientadas para el diagnóstico de fallos. Algunas propuestas evitan la explosión combinatoria de estados, así que pueden ser usadas en sistemas complejos. En [88], los autores presentan un método para modelar un sistema de manufactura flexible el cual incluye modelo de fallos (basado en árboles de fallo). En [89], los autores presentan un método para modelar y diagnosticar en un BPEL Services; este enfoque es restrictivo a modelos se servicios web. En [90] los autores presentan un método para incluir el diagnóstico de fallos en un controlador embebido, sin embargo, este enfoque tiene el mismo punto de vista de los otros enfoques basados en modelo.

En [91] se presenta un método de diagnóstico basado en una CPN, llamado método de anidamiento latente. La idea es partir de un modelo del sistema basado en una PN, que represente el comportamiento dinámico correcto aplicando técnicas de modelado de PN generalizadas. El conjunto de fallos a ser diagnosticados son definidos y asignados a un subconjunto de marcas de color. Un evento de fallo será definido estableciendo condiciones dinámicas en cada marcado y subsecuentemente en cada estado alcanzado por el sistema y por el conjunto de lecturas no esperadas de los sensores. Luego, las marcas coloreadas de fallo son apropiadamente localizadas en los lugares llamados lugares de anidamiento latente de fallo. Estas marcas son susceptibles de disparar desde un lugar por la activación de una secuencia de eventos, s, asociada a un lectura no normal de un sensor.

Las ventajas de los métodos con modelo de fallo, radican en la po-

sibilidad de generar garantías con respecto a la diagnosticabilidad de fallos; además, si ciertas condiciones se mantienen, los fallos considerados en el modelo se pueden localizar con precisión. Una desventaja inherente de los enfoques basados en modelos del fallo es que sólo los fallos considerados explícitamente en el modelo del sistema pueden ser detectados y localizados.

Los métodos de diagnóstico sin modelo de fallo evitan este inconveniente; además, construyen modelos simples ya que no es necesario ningún conocimiento especial del comportamiento defectuoso del sistema; un inconveniente de este enfoque es la localización de los fallos, ya que los modelos han sido construidos con menos conocimiento y otro inconveniente es que la diagnosticabilidad de fallos dados, normalmente no se puede garantizar. En [6] se utilizan plantillas de condición para determinar si el sistema genera eventos en el orden correcto o dentro de los plazos de tiempo determinados; se detecta un fallo cuando hay ausencia o reacciones equivocadas en el proceso. En estos casos, los eventos relacionados con la plantilla ayudan a aislar el fallo. En [7] un fallo es detectado por la ocurrencia de un evento inesperado o por la no ocurrencia de un evento esperado dentro de intervalos de tiempo predefinidos, la ocurrencia de un evento inesperado se detecta cuando la condición de habilitación de este evento no está satisfecho y la no ocurrencia de un evento esperado se detecta mediante una plantilla Otra clase de métodos se basan en la comparación de las salidas del sistema con el modelo de salidas nominales. En [8, 3] el método propuesto compara el comportamiento observado con el comportamiento esperado identificado previamente; un fallo puede ser detectado y se determina un conjunto de fallos candidatos; inspirado en el método de residuos para sistemas continuos, introduce diferentes conjuntos de operaciones para generar un conjunto de fallos candidatos. Después de la detección y localización de un primer fallo, se presenta un procedimiento para realizar una mejor localización del fallo a

partir del análisis del comportamiento observado del sistema.

Respecto a diagnosticabilidad Sampath, [74], provee una definición de diagnosticabilidad en el marco de los lenguajes formales y presenta una condición necesaria y suficiente para diagnosticabilidad de sistemas. En el marco de las IPN, [92] define y caracteriza la propiedad de diagnosticabilidad de entrada - salida en modelos de IPN que evita el análisis de alcanzabilidad y proveen condiciones estructurales suficientes para diagnosticabilidad de fallos permanentes basados en la condición que algún T-semiflujo debe contener todas las transiciones de riesgo, pero en [21] ellos mejoran su propuesta y presentan un algoritmo polinomial que calcula el área de influencia y determina si una IPN es diagnosticable. Cabasino en [93] provee una condición necesaria y suficiente para diagnosticabilidad de PN acotadas, es decir, PN cuyo marcado alcanzable es finito. Su propuesta está basada en la construcción de dos grafos orientados y etiquetados denotados como grafo de alcanzabilidad básico modificado (MBRG) y el grafo diagnosticador de alcanzabilidad básico (BRG), concepto introducido [53] para el diagnóstico de IPN acotadas. La efectividad del procedimiento propuesto ha sido ilustrado en [94], donde mostraron que, especialmente en la presencia de sistemas altamente concurrentes, el número de marcas básicas es siempre mucho menor con respecto al número de marcas alcanzables (que incrementa exponencialmente con el tamaño de la red).

3.4. Análisis de los métodos

Con el objeto de analizar el proceso de diagnóstico, a continuación se relaciona el funcionamiento de un método basado en modelo de fallo y uno sin modelo de fallo.

Fanti et al. [1], parten de los resultados de [86] y dirigen el problema de diagnóstico de fallos en un sistema que presentan una fuente de no-determinismo originado del hecho de que diferentes transiciones observables pueden tener la misma etiqueta. El modelo es representado como una IPN, $S = \langle PN, E, \lambda, M_o \rangle$, los fallos son modelados por transiciones inobservables (existen transiciones T_o y T_u). Este modelo considera una subred, $PN_u \angle_{T_u} PN$ como una subred acíclica para las T_u; $\Delta = \{f_1, \cdots, f_F\}$ como el conjunto de fallos que pueden ocurrir, cada $f_i \in \Delta$ es modelado como una T_u y un fallo ocurre cuando f_i se dispara. $\lambda(t) = \varepsilon$ si $t \in T_u$ y $\lambda(t) = e$ si $t \in T_o$, el conjunto de transiciones con la misma etiqueta es $T(e) = \{t \in T_o \mid \lambda(t) = e\}$. Se define el conjunto de interpretaciones como $\Sigma(M_0, \sigma_0, f_i) = \{\sigma \in \Sigma(M_0, \sigma_0) \mid f_i \in \sigma\}$; es decir, es el conjunto de secuencias observables y que contiene la transición de fallo; la función Φ como $\Phi(M_0, \sigma_0) = \{N\}$ si $\forall f_i \in \Delta \mid \Sigma(M_0, \sigma_0, f_i) = \emptyset$, es decir todas las interpretaciones que no contienen fallos o como $\Phi(M_0, \sigma_0) = \{f_i \in \Delta \mid \Sigma(M_0, \sigma_0, f_i) \neq \emptyset\}$ cuando los fallos son contenidos en al menos una interpretación y como $\Phi(M_0, \sigma_0) = \{f_i \in \Delta \mid \Sigma(M_0, \sigma_0, f_i) \neq \emptyset\} \cup \{N\}$ cuando hay dos interpretaciones simultáneas es decir: i) el fallo f_i está contenido en el menos una interpretación y ii) existe al menos una interpretación que no contiene el fallo f_i. El diagnosticador es una función $D : L \times \Delta = \{0, 1, 2\}$ que asocia a cada observación $\omega \in L$ ($\omega = \lambda(\sigma_0)$) y a cada fallo $f_i \in \Delta$ un diagnóstico de estado. 0: normal; 1: fallido y 2: ambiguo. El algoritmo propuesto define y soluciona $\Phi(M_0, \sigma_0)$ como un problema de programación lineal entera (ILPP). En la Figura 3.4.1, se muestra un sistema al cual se aplica la propuesta de [1] y la Tabla 3.4.1 muestra el conjunto de transiciones con las respectivas etiquetas.

En el diagnosticador (Figura 3.4.2), suponiendo que se observa el evento $e1$, las transiciones que tienen esa etiqueta son $\sigma_{0,1} = t_1$, $\sigma_{0,2} = t_3$, $\sigma_{0,3} = t_5$; aplicando el algoritmo $\Phi(M_0, t_1) = \{N\}$,

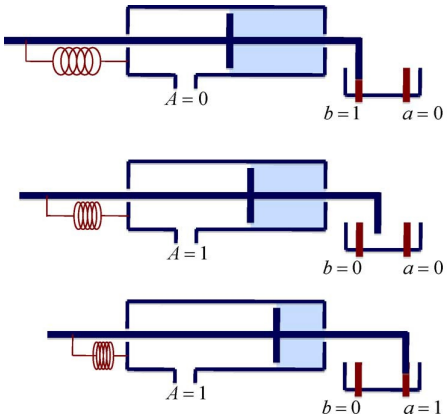


Figura 3.4.1: Cilindro

Tabla 3.4.1: Asociación de transiciones a eventos de la Figura 3.4.1

Etiquetas de Eventos Observables	Transiciones Asociadas
e_1 Incremento de presión.	$T(e_1) = \{t_1, t_3, t_5\}$.
e_2 Decremento de presión.	$T(e_2) = \{t_2, t_4, t_6\}$.
e_3 La señal b cambia de 0 a 1.	$T(e_3) = \{t_7\}$.
e_4 La señal b cambia de 1 a 0.	$T(e_4) = \{t_8\}$.

$\Phi(M_0, t_3) = \Phi(M_0, t_5) = \emptyset$; por lo tanto se establece que $\Lambda(e_1) = \{\sigma_{0,1}\}$ y el estado del diagnosticador $D(e_1, f_1) = 0$ y $D(e_1, f_2) = 0$ entonces el sistema tiene comportamiento normal.

Asumiendo un segundo evento e_2 y $\omega = e_1 e_2$, entonces $\sigma_{0,1} = t_1 t_2$, $\sigma_{0,2} = t_1 t_4$ y $\sigma_{0,3} = t_1 t_6$ y el algoritmo determina $\Phi(M_0, t_1, t_2) = \{N\}$, $\Phi(M_0, t_1 t_4) = \{f_1\}$ y $(M_0, t_1 t_6) = \{f_1\}$; por lo tanto se establece que $\Lambda(e_1 e_2) = \{t_1 t_2, t_1 t_4, t_1 t_6\}$ y el diagnosticador da $D(e_1 e_2, f_1) = 2$ y $D(e_1 e_2, f_2) = 0$ entonces el sistema tiene un comportamiento ambiguo.

Fanti et al. además proponen un método de diagnóstico distribuido

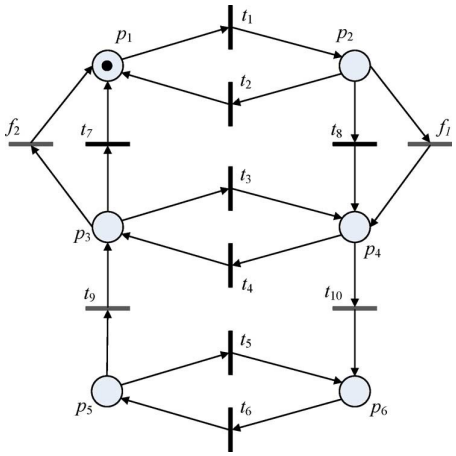


Figura 3.4.2: Modelo de diagnóstico de cilindros, basado en [1]

para sistemas complejos. Asumen que hay un protocolo de comunicación entre los módulos, de modo que un módulo a la vez chequea si ha ocurrido un evento e inicia el proceso de detección de fallo; el diagnosticador del módulo captura ω y halla Φ y Λ y evalúa la función D en cada módulo; de tal manera que al conocer los marcados alcanzados por los módulos se puedan analizar los lugares comunes alcanzados. El uso de los lugares comunes tiene una doble ventaja: *i)* el hecho que no están conectados con las transiciones silenciosas, las marcas correspondientes sólo dependen de las ocurrencias de transición observables y *ii)* la información de los eventos en los módulos vecinos se puede utilizar para actualizar las marcas de lugar común. Una vez que se determina la función, el módulo envía al lugar marcas comunes a los módulos cercanos.

Esta propuesta parte del conocimiento del conjunto de fallos y del modelo de fallo no-determinístico, en la cual los fallos son detectados on-line. El esfuerzo computacional crece con el número de eventos y no se tiene en cuenta el tiempo de los eventos.

3.4.1. Concepto de residuos para localización de fallos en SED.

Una de las principales desventajas de los métodos con modelo de fallo es que limita el proceso de diagnóstico a fallos definidos previamente, basados en la experiencia y conocimiento del proceso o asumiendo fallo en todos los dispositivos del sistema, [91].

Como el enfoque de este trabajo es el de diagnóstico de fallos usando un modelo del sistema libre de fallos, a continuación se analiza una propuesta que utiliza el enfoque de residuos para detectar un fallo.

En [2] se propone un método de identificación para propósitos de detección de fallos; considera un sistema en lazo cerrado compuesto por un controlador y el dispositivo bajo control, como el mostrado en la Figura 3.4.3, donde el controlador tiene un comportamiento determinista pero la planta no. Como no se conoce el modelo de la planta ésta se asume como una "*caja negra*" que puede ser modelada a partir de eventos. Una forma apropiada para representar el comportamiento observable del sistema es como una máquina de estados finitos que refleje las características de un generador de eventos de una "*caja negra*".

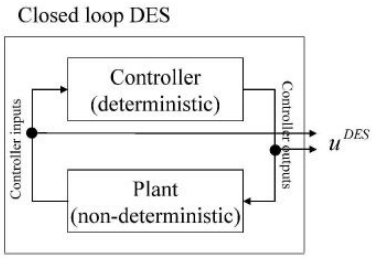


Figura 3.4.3: Sistema en lazo cerrado definido en [2]

Las señales observadas son las señales de entrada y salida del con-

trolador, representadas como un vector de E/S en un tiempo t, denotado como $u_i(t)$. Cada observación corresponde a un estado estable del sistema. El subíndice i representa la secuencia de la observación de un ciclo de producción dado, $\sigma_i = (u_i(1), \cdots, u_i(|\sigma_i|))$. Es de notar que el sistema a considerar es cíclico, por lo tanto $\forall(i,j), u_i(1) = u_j(1), u_i(|\sigma_i|) = u_j(|\sigma_j|)$. El conjunto de todas las observaciones, $\Sigma = \left\{\sigma_1, \cdots, \sigma_{|\Sigma|}\right\}$, contiene el comportamiento observado que se le ha denominado modelo de transición de estado. Los comportamientos asumidos en el sistema se muestran en la Figura 3.4.4.

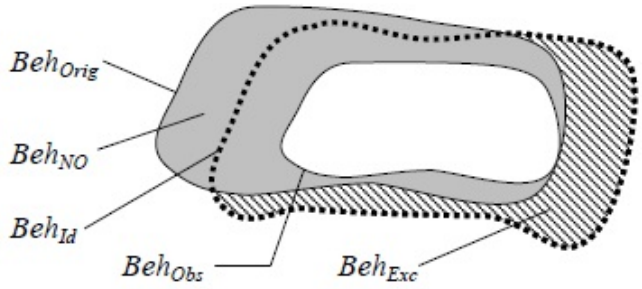


Figura 3.4.4: Comportamiento de un DES, según Klein ([2])

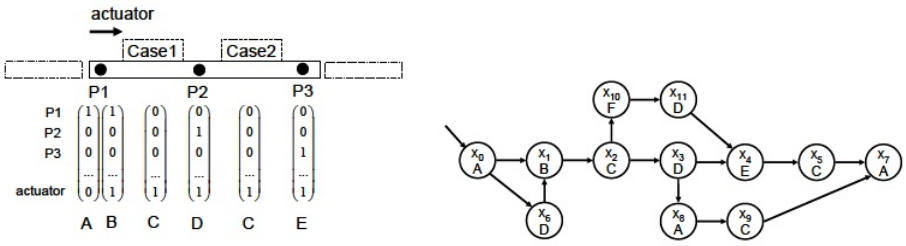
Cada uno de estos comportamientos es un conjunto finito de trayectorias entre un conjunto finito de estados.

El autor propone representar trayectorias en un espacio de estados finito como un autómata autónomo no-determinista con salida, (NDAAO). El lenguaje observado es un conjunto de palabras de tamaño k tal que: (una letra es un vector de E/S).

$L^k_{Obs} = \bigcup_{\sigma_i \in \Sigma} \left(\bigcup_{j=1}^{|\sigma_i|-k+1} (u_i(j), u_i(j+1), \cdots, u_i(j+k-1)) \right)$ y el comportamiento observado de tamaño n es $Beh^n_{Obs} = \bigcup_{k=1}^{n} L^k_{Obs}$.

El algoritmo de identificación de [2] consiste en la construcción del autómata autónomo no-determinista con salida, (NDAAO); que genera el lenguaje observado de tamaño k; de tal manera que $L_{Obs}^{k+1} = L_{Ident}^{k+1}$; y además permite establecer el parámetro k, para ajustar la precisión del autómata identificado.

Para visualizar el método de identificación, se consideran las secuencias de vectores de E/S observadas: $\sigma_1 = (A, B, C, D, E, C, A)$, $\sigma_2 = (A, D, B, C, D, A, C, A)$ y $\sigma_3 = (A, D, B, C, F, D, E, C, A)$, para el sistema de la Figura 3.4.5a y el NDAAO identificado se muestra en la Figura 3.4.5b.



(a) Sistema en lazo cerrado **(b)** NDAAO Identificado

Figura 3.4.5: Transportador con dos escenarios, presentado en [2]

En [59, 3] los autores proponen un método de detección y aislamiento de fallos (FDI), basado en el algoritmo de Klein para identificar el comportamiento libre de fallo. En la Figura 3.4.6 se observa el procedimiento de este método.

La detección del fallo es realizada en tiempo real comparando las salidas del modelo y las salidas del SED en el bloque de "Detección y aislamiento del fallo". El modelo para detección y diagnóstico deben ser iniciados de manera paralela. Si hay diferencia entre las secuencias observadas y las modeladas, un fallo es detectado. Además, comparando las secuencias observadas y las esperadas es posible localizar el fallo, con base en las lecturas de los sensores

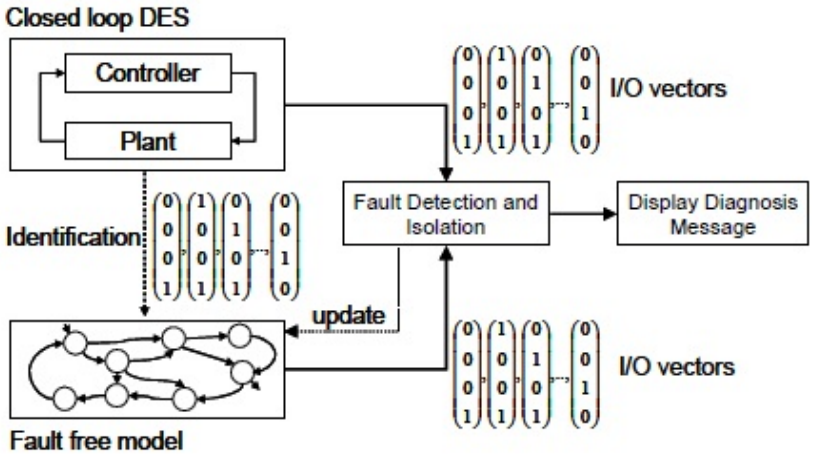


Figura 3.4.6: Método de FDI de Roth

que han fallado. La información de diagnóstico puede ser delegada a los operadores del sistema.

Durante el procedimiento de detección y aislamiento del fallo pueden suceder desviaciones que se asumen como fallos, pero pueden ser falsas alarmas. Esta decisión la toma los expertos del sistema o por algunas reglas eurísticas; en este caso el modelo se actualiza con las nuevas secuencias en comportamiento libre de fallo. Cuando un fallo ha sido detectado el modelo debe ser reiniciado.

Para la localización del fallo utilizan el cálculo de un residuo, el cual se calcula como: $Res = ES\left(\lambda\left(x\right), u_{actual}\right) \backslash \bigcup_{\forall x' \in r(x)} ES\left(\lambda\left(x\right), \lambda\left(x'\right)\right)$, donde $ES\left(\lambda\left(x\right), \lambda\left(x'\right)\right)$, (ver Definición 8 de [3]), es el conjunto de los "bordes" de las subidas y bajadas observadas entre dos funciones de salida consecutivas $\left(\lambda\left(x\right), \lambda\left(x'\right)\right)$; $r\left(x\right)$ es una relación de transición no-determinista $r : X \rightarrow 2^X$, donde 2^X es el powerset del espacio de estados X. Por lo tanto, existe un evento de fallo

cuando se genera un vector de E/S que no pertenece al modelo identificado y al menos una de sus señales binarias ha cambiado; si cambia de 0 a 1 el "borde" es 1; de 1 a 0 el "borde" es 0 y si no hay cambio el "borde" es ε; el conjunto de todos los "bordes" de dos vectores de E/S es el residuo Res.

De manera muy general a continuación se presenta un resumen del proceso de FDI, propuesto por estos autores, basado en [3] y en el esquema de la Figura 3.4.7.

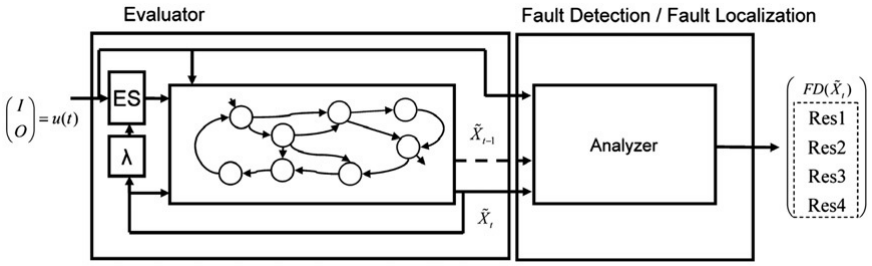


Figura 3.4.7: Proceso de FDI de [3]

El estado inicia es el estado x_0.

- Se observa un nuevo vector de E/S $u(t)$.
- Calcula $\lambda(x)$.
- Se estima $\widetilde{X}_{t-1}$ si $\widetilde{X}_{t-1} = \varepsilon$; entonces hallar un $x_t \mid \lambda(x) = u(t)$ y x_t es el estado actual.
- Si $\widetilde{X}_{t-1} \neq \varepsilon$; se calcula $ES(\lambda(x_t), \lambda(x_{t-1}))$ en los posibles estados a ser alcanzados $\widetilde{X}_t$; si $\left|\widetilde{X}_t\right| \neq 1$ un fallo se ha detectado y x_t es el estado actual.
- Si $\left|\widetilde{X}_t\right| > 1$;
  - Calcular el comportamiento inesperado como: $Res1(\widetilde{x}, u(t)) = ES(\lambda(\widetilde{x}), u(t)) \setminus \bigcup_{\forall x' \in r(\widetilde{x})} ES\left(\lambda(\widetilde{x}), \lambda\left(x'\right)\right)$.

- Calcular el comportamiento esperado como: $Res2\left(\widetilde{x},u\left(t\right)\right)=ES\left(\lambda\left(\widetilde{x}\right),u\left(t\right)\right)\setminus\bigcap_{\forall x^{'}\in r\left(\widetilde{x}\right)}ES\left(\lambda\left(\widetilde{x}\right),\lambda\left(x^{'}\right)\right)$.
- Calcular el comportamiento perdido (esperado pero no observado) como: $Res3\left(\widetilde{x},u\left(t\right)\right)=\bigcap_{\forall x^{'}\in r\left(\widetilde{x}\right)}ES\left(\lambda\left(\widetilde{x}\right),\lambda\left(x^{'}\right)\right)\setminus ES\left(\lambda\left(\widetilde{x}\right),u\left(t\right)\right)$.
- Calcular el comportamiento esperado como: $Res4\left(\widetilde{x},u\left(t\right)\right)=\bigcup_{\forall x^{'}\in r\left(\widetilde{x}\right)}ES\left(\lambda\left(\widetilde{x}\right),\lambda\left(x^{'}\right)\right)\setminus ES\left(\lambda\left(\widetilde{x}\right),u\left(t\right)\right)$.

- Reducción de los fallos detectados.

En la Figura 3.4.8 se muestra el NDAAO con fallos detectados para un sistema modelado como un NDAAO.

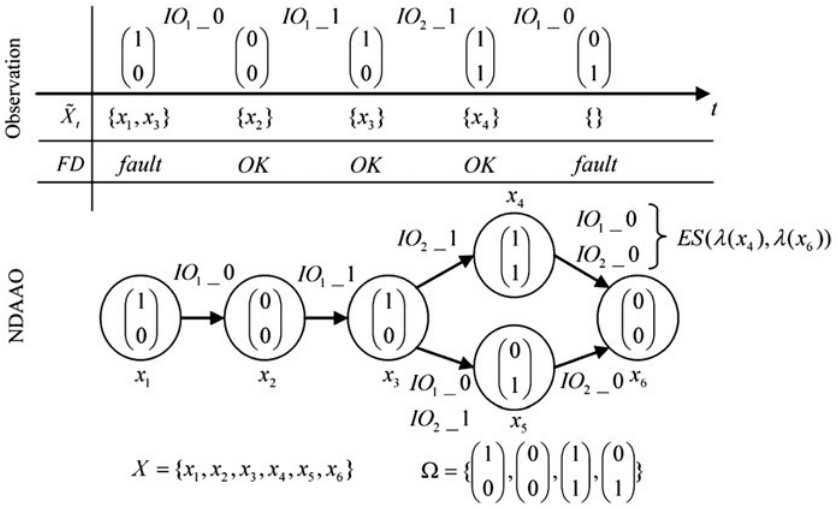


Figura 3.4.8: NDAAO para FDI de [3].

En sistemas de gran escala con alto grado de concurrencia, este método no observa todas las palabras libres de fallo en un tiempo

razonable; entonces [8] presenta un método de diagnóstico distribuido basándose en el NDAAO de Klein. Ellos proponen dividir el sistema en subsistemas concurrentes para que sean identificados on-line, incluso si no están totalmente observados. El sistema global es dividido a partir de la división de los vectores de E/S, con base en algún conocimiento previo del sistema. Cada subsistema es identificado como un NDAAO.

3.4.2. Discusión

Los métodos de diagnóstico basados en modelos de fallo son muy eficientes cuando suceden fallos previstos, debido a la diagnosticabilidad del modelo; pero cuando el número de fallos es alto y se utiliza la composición paralela para encontrar el diagnosticador los modelos no son manejables; el uso de las PN reduce el tamaño; otro inconveniente es que es necesario un estudio y conocimiento profundo del sistema, con el objeto de tener un conjunto de fallos previstos. Entonces la ocurrencia de un fallo no modelado implica un mal funcionamiento del diagnosticador e imprecisiones en el aislamiento del fallo.

En [1] se presenta un método de diagnóstico con modelo de fallo previo, con las características antes mencionadas. El diagnosticador es no-determinista y fácilmente se llega a un estado ambiguo en el proceso de diagnóstico. La localización del fallo es muy sencilla puesto que se parte de fallos previstos. Un inconveniente del método es que $\Phi\left(M_0, \sigma_0\right)$ se soluciona como un problema de programación lineal entera (ILPP) y esta solución tiene costo alto de cómputo en la medida que el sistema crezca.

Por otro lado, los métodos de diagnóstico sin modelo de fallo presentan ventajas frente a los anteriores; puesto que permiten detectar y aislar diferentes tipos de fallo y como parten sin conocimiento

previo del sistema pueden ser aplicados a cualquier tipo de sistemas independiente de su tamaño, solo con la medición de las señales que se intercambian entre el controlador y la planta.

El método de FDI presentado en [3] tienen grandes ventajas respecto a propuestas sin modelo de fallo, aunque el modelo sobre el cual basa la detección y aislamiento es un modelo no-determinista y esta es una propiedad no deseable en el diagnóstico de fallos, puesto que se presenta ambigüedad en la evolución de los estados, con tendencia a confundirse con fallos. Además, el tamaño de las palabras juega un papel muy importante en la precisión del modelo identificado y este tamaño no es fácil de encontrar en la medida que el sistema aumenta de tamaño. Por otro lado, algunos fallos debidos a fallos en los sensores no se detectan y no tiene en cuenta el tiempo como herramienta de identificación del modelo.

4 Modelado de Sistemas de Eventos Discretos

En este capítulo se presenta una propuesta de modelado de SED como un conjunto de Redes de Petri Interpretadas Temporizadas Estocásticas, a partir del lenguaje finito generado por el sistema, con el objeto que el modelo resultante se aplique en el diagnóstico de fallos de SED estocásticos; por lo tanto ésta debe permitir explorar los eventos observables para detectar la ocurrencia de eventos no-observables. Los eventos no-observables son eventos que ocurren por algún tipo de fallo o son eventos que generan cambios en el sistema sin que sean registrados por los sensores.

Como se describió en la introducción, de acuerdo a [30], un modelo debe generarse para resolver preguntas sobre el comportamiento del sistema al que representa: por lo tanto, las preguntas a solucionar teniendo en cuenta que el modelo se genera para realizar diagnóstico son:

- Dado el lenguaje legal de un SED, ¿cómo representarlo bajo una PN de tal manera que el lenguaje legal sea igual al lenguaje de la PN?.
- ¿Qué tipo de eventos no-observables pueden ser explicados y bajo qué condiciones, a partir de algunas secuencias de eventos observados?.

Para responder las anteriores preguntas, primero se hace necesario proponer una forma particular de representación de las señales de

Entrada - Salida (E/S), como símbolos o letras de un alfabeto para definir lenguajes. Esta propuesta se presenta en la siguiente definición.

Definición 19 (Representación de señales binarias)**.** Dado un conjunto de ns señales, cuyos valores se representan en cadenas de ceros o unos; la notación x_{dec} representa la combinación de una cadena tal que x es el nombre de la señal y dec es la representación decimal de la cadena. El conjunto X agrupa todas las combinaciones posibles de las señales; es decir $X = \{x_0, \cdots, x_{2^{ns}-1}\}$.

Por ejemplo: si el número de señales binarias es $ns = 3$, entonces $X = \{x_0, x_1, \cdots, x_7\}$; donde $x_0 = [000] \cdots x_7 = [111]$.

Por otro lado, es importante definir el sistema a considerar; el cual es un sistema de condiciones en lazo cerrado con señales de entrada, las señales externas que afectan al sistema y las órdenes de control que son emitidas por el controlador a la planta y con señales de salida, la respuesta de los sensores a las órdenes de control. La ocurrencia de eventos se considera temporizada y de naturaleza estocástica.

Con base en lo anterior, se propone un generador de lenguaje que representa el comportamiento de un SED con las características del sistema a considerar expuestas, que permita realizar diferentes pruebas de observabilidad. Esta propuesta se explica en las siguientes secciones.

4.1. Definición del Sistema

La mayoría de los sistemas dinámicos de gran escala, pueden considerarse como SED en un determinado nivel de abstracción, con mayor justificación los sistemas de control en el nivel de supervisión. En este sentido, se han propuesto modelos de eventos discretos

aplicables no solamente a los sistemas específicos clasificables como SED, sino también a los sistemas de naturaleza continua.

El sistema a considerar es un SED de gran escala. Este tipo de sistema está compuesto por una gran cantidad de dispositivos distribuidos independientes o no, que generan concurrencia de estados a partir de la ocurrencia asíncrona o síncrona de eventos discretos, algunos controlados y otros no; es decir sistemas cuyas variables evolucionan de manera discreta en una base de tiempo en general continua.

Además es un sistema que se ve afectado por una cantidad de factores que hacen que su comportamiento sea variable y por lo tanto es un sistema estocástico. Los factores que influyen en esta variabilidad, entre otros son: la fluctuación en el suministro de energía, la variabilidad en los materiales a procesar, deterioro de equipos y herramientas.

El sistema a considerar tiene una estructura basada en la lectura de señales de E/S, como el mostrado en la Figura 4.1.1.

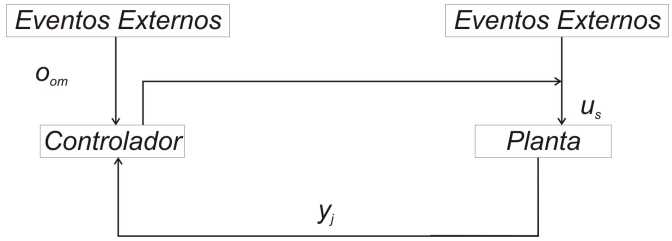


Figura 4.1.1: Sistema a modelar

El comportamiento del sistema se ve afectado por señales de entrada tales como: externas al controlador, externas a la planta e internas a la planta (órdenes de control): $Entradas = \{EnExC, EnExP, EnInP\}$; y por señales de salida,

representadas por las lecturas sensoriales. La planta y el controlador se interconectan vía señales internas en lazo cerrado.

$EnExC$ son las entradas externas al controlador $EnExC = \{ec_1, \cdots, ec_{n_{ec}}\}$, n_{ec} es el número de $EnExC$, tales como: comandos del SCADA, requerimientos del operario, entre otros, estas señales afectan el funcionamiento del sistema. Su lectura es binaria e indica si un evento externo está activo o no en un instante de tiempo determinado. Al conjunto de todas las combinaciones posibles de $EnExC$ se le ha denominado modo de operación (OM). Con base en la Definición 19, $OM = \{o_0, \cdots, o_{2^{n_{ec}}-1}\}$, entonces $o_{om} \in OM$ es:

$$o_{om} = [ec_1 \cdots ec_{n_{ec}}] \tag{4.1.1}$$

$EnExP$ son las señales de entrada externas que afectan directamente a la planta y pueden corresponder a comportamientos legales o comportamientos de fallo y pueden ser observables o no observables, entonces $EnExP = \{EnExP^o, EnExP^{uo}\}$, con $EnExP^o = \left\{eop_1, \cdots, eop_{n_{eop}}\right\}$ donde n_{eop} es el número de entradas externas observables a la planta, $EnExP^{uo} = \left\{euop_1, \cdots, euop_{n_{euop}}\right\}$, n_{euop} es el número de entradas externas no observables a la planta.

$EnInP$ son las entradas internas a la planta y están constituidas por el conjunto de acciones de control. $EnInP = \{Cc\} = \{cc_1, \cdots, cc_{n_{cc}}\}$, siendo n_{cc} el número de órdenes de control.

Las señales de salidas se representan por el conjunto de lecturas sensoriales: $Salidas = \{sr_1, ..., sr_{n_{sr}}\}$, siendo n_{sr} el número lecturas sensoriales.

Para modelar el comportamiento de la planta controlada se interrelacionan sus entradas y sus salidas. Las entradas a la planta son $U \subset Entradas$, donde $U = \{EnExP^o, EnExP^{uo}, Cc\}$, Cc,

$EnExP^o$ son entradas observables y $EnExP^{uo}$ son entradas no observables. Las salidas de la planta son $Y = \{Salidas\}$. Todas se consideran observables porque se basan en las lecturas observadas de los sensores.

El conjunto $U = 2^m$, es el conjunto de todas las combinaciones posibles de entradas a la planta, donde m es el número total de entradas, $m = m_{uo} + m_o$, donde m_o es el número de entradas observables, $m_o = n_{eop} + n_{cc}$ y el conjunto $Y = 2^n$, es el conjunto de todas las combinaciones posibles de salidas de la planta, donde n es el número total de salidas, $(n = n_{sr})$.

Un símbolo de entrada u_s es el conjunto de valores de entrada en un instante determinado; entonces:

$$u_s = \left[euop_1 \cdots eup_{n_{euop}} eop_1 \cdots ep_{n_{eop}} cc_1 \cdots cc_{n_{cc}}\right] \tag{4.1.2}$$

u_s es un símbolo de entrada cuya representación se basa en la Definición 19.

Un símbolo de salida y_j es el conjunto de valores de salida en un instante determinado; entonces:

$$y_j = [sr_1 ... sr_n] \tag{4.1.3}$$

y_j es un símbolo de salida cuya representación se basa en la Definición 19.

Por ejemplo, dado un conjunto de entradas a la planta $U = \{cc_1, cc_2\}$; si $cc_1 = 1$ y $cc_2 = 0$ entonces el símbolo de entrada está representado por u_2.

Un símbolo de E/S relaciona un símbolo de entrada y uno de salida: $(u_s y_j)$, donde $u_s \in U$ siendo $U = \left\{u_0, u_1, \cdots, u_{|2^m|-1}\right\}$ y $y_j \in Y$

siendo $Y = \left\{y_0, y_1, \cdots, y_{|2^n|-1}\right\}$; por lo tanto, un símbolo de E/S se considera un evento compuesto, como el definido en la Definición 2.

Definición 20 (Alfabeto del sistema)**.** El alfabeto del sistema Ω, es el conjunto de símbolos de E/S temporizados ω^i. Con $\omega^i = (u_s y_j).f\left(t_{om}^{ev}\right)$, donde $u_s y_j \in U.Y$, $|U.Y| = |2^m| \times |2^n|$ y $f\left(t_{om}^{ev}\right) \in \delta$ es la función de densidad de probabilidad del evento y δ es una función de asignación de las funciones de densidad. Es decir $\omega^i \in (U.Y)^*.\delta$.

Dado que el comportamiento de los sistemas industriales no es determinístico, se considera modelar bajo lenguajes estocásticos temporizados.

El lenguaje estocástico temporizado $\mathcal{L}$, para el sistema a modelar es un subconjunto de secuencias de eventos temporizados; tal que, $\mathcal{L} \subset \Omega^*$, donde $\Omega : \{\omega^i = (u_s y_j).f\left(t_{om}^{ev}\right)\}$ (ver Definición 5), que se generan cuando existe un evento en instantes τ_i. Es decir el lenguaje del sistema sobre el alfabeto es un subconjunto de Ω^*; tal que:

$$\mathcal{L} = \{s \,|s \subset \Omega^*\} \tag{4.1.4}$$

Por lo tanto, $\mathcal{L} = \left\{\omega^0, \cdots, \omega^k\right\}$ a instantes $\tau_0 \leq \cdots \leq \tau_k$.

4.1.1. Restricciones del Sistema

El sistema tiene las siguientes restricciones:

- El sistema es controlado en lazo cerrado. Además cada subsistema puede estar solo en un estado a la vez.

- El intercambio de señales entre el controlador y la planta son señales discretas con solo dos valores (código binario).
- Su comportamiento es definido por la generación de eventos internos y externos en una línea de tiempo.
- Un evento externo afecta al controlador; un evento interno es la concatenación de las órdenes de control y de las lecturas sensoriales en un instante de tiempo específico.
- Una secuencia de eventos externos es una secuencia temporizada de eventos que genera un "Modo de operación" del sistema.
- Un evento interno se genera solo uno a la vez. Si hay más de un evento en un mismo instante de tiempo, éstos se ordenan en diferentes instantes pero con una duración entre ellos de cero.

4.1.2. Funcionamiento del Sistema

Los eventos externos llegan al controlador e imponen un modo de operación particular. Una vez se ha introducido un modo de operación, el controlador ejecuta una estrategia de control generando una secuencia de órdenes de control, esta secuencia entra a la planta y activa los actuadores, la planta reacciona y genera una secuencia de lecturas sensoriales.

De ese modo, en cada instante, se aplica una orden de control $u_s \in U$, formada por una combinación particular de acciones de control individuales. Asimismo, en cada instante se dispone de una lectura sensorial $y_j \in Y$, formada por una combinación particular de lecturas sensoriales, la cual determina un estado particular del sistema.

Es importante remarcar que en cualquier instante de tiempo se está aplicando al sistema un conjunto de órdenes de control y se obtiene un conjunto de lecturas sensoriales, puesto que los sensores siempre están midiendo y la orden de control "*no activar nada*" también está incluida. Por lo tanto, el comportamiento del sistema puede ser descrito en términos de la secuencia de eventos temporizados.

El proceso de modelado de un sistema consiste en calcular el modelo matemático a partir de la medición de sus señales de entrada y de salida. Para ello se requiere de una estructura que asegure un flujo ordenado de los acontecimientos y que se adapte a la creciente complejidad de estos sistemas.

Con base en las definiciones de símbolo de entrada, símbolo de salida, alfabeto del sistema; a continuación se presenta un ejemplo de modelado con base en una IPN, (ver Definición 9).

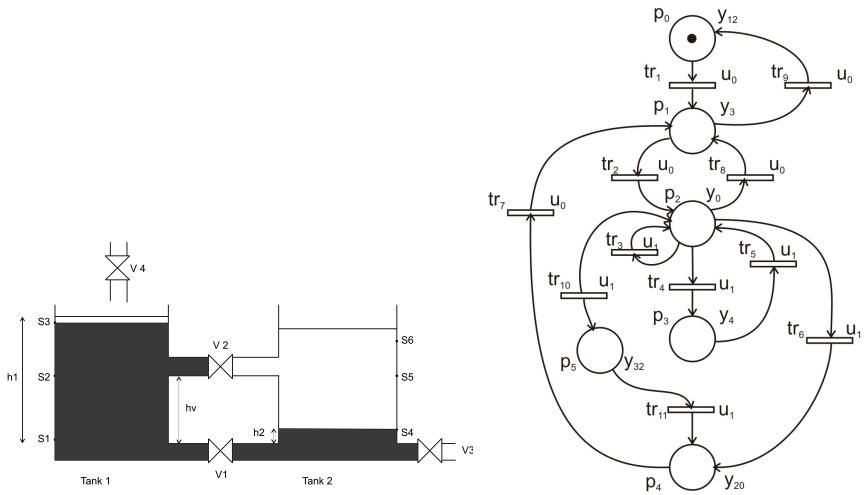
Ejemplo 4. El sistema a considerar son los tanques acoplados presentados en [95]. En este ejemplo se incluyen nuevos sensores con el objeto de extender el lenguaje del sistema. El sistema tiene dos tanques conectados, donde h_1 es el nivel de líquido en tanque 1 y h_2 es el nivel en tanque 2. El flujo entre tanques puede ser regulado con válvulas. El sistema tiene tres sensores en cada tanque y cuatro válvulas (ver Figura 4.1.2a). El sistema trabaja con una política de control que mantiene el nivel de los tanques sobre la altura h_v. Si $h_1, h_2 < h_v$ entonces cerrar la válvula $\overline{v}_3$, de lo contrario abrirla. Por lo tanto, la estrategia de control tiene solo una orden de control.

Entonces $Cc = \{cc_1\}$. El conjunto de símbolos de entrada es $U = \{u_0, u_1\}$, ya que $cc_0 = \{0, 1\}$. Las salidas del sistema son: $Salidas = \{sr_1, sr_2, sr_3, sr_4, sr_5, sr_6\}$ y el conjunto de símbolos de salida es $Y = \{y_0, \cdots, y_{63}\}$.

Asumiendo que las lecturas sensoriales inicialmente son: $sr_1 = 0$, $sr_2 = 0$, $sr_3 = 1$, $sr_4 = 1$, $sr_5 = 0$ y $sr_6 = 0$, el símbolo de salida

inicial es $y_{12} = [001100]$ lo cual significa que el tanque 1 está lleno y el tanque 2 vacío y $cc = 0$. Por lo tanto en τ_0 el símbolo de salida es y_{12} y el símbolo de entrada es u_0. El lenguaje observado es: $\mathcal{L}^o = \Big\{(u_0y_{12}), (u_0y_3), (u_0y_0), (u_1y_0), (u_1y_4), (u_1y_0), (u_1y_{20}), (u_0y_3), (u_0y_0), (u_0y_3), (u_0y_{11}), (u_0y_3), (u_0y_0), (u_1y_{32}), (u_1y_{20}), (u_0y_3), (u_0y_{12})\Big\}$, a instantes $\tau_0, \tau_1, \tau_2, \tau_3, \tau_4, \tau_5, \tau_6, \tau_7, \tau_8, \tau_9, \tau_{10}, \tau_{11}, \tau_{12}\tau_{13}, \tau_{14}, \tau_{15}, \tau_{16}$.

Asumiendo φ no isomórfica, la IPN (Q), que genera el lenguaje para este ejemplo se muestra en la Figura 4.1.2b.



(a) Proceso **(b)** IPN que genera el lenguaje observado

Figura 4.1.2: Sistema de tanques acoplados.

Es importante tener en cuenta que el sistema es determinístico pero la red no. Una consecuencia de que la red sea no determinista es que genera secuencia de eventos que no hacen parte del lenguaje del sistema.

El lenguaje de la red es, (Definición 11): $\mathcal{L}(Q) = \Big\{((u_0y_3)(u_0y_{12}))^* \Big((u_0y_3)(u_0y_0)(u_0y_3)(u_0y_{12})\Big)^* \Big((u_0y_3)(u_0y_0)(u_1y_0)(u_1y_4)(u_1y_0)(u_1y_{20})(u_0y_3)(u_0y_{12})\Big)^* (\cdots) \Big((u_0y_{12})(u_0y_3)(u_0y_0)(u_1y_0)(u_1y_4)(u_1y_0)(u_1y_{20})(u_0y_3)(u_0y_0)(u_0y_3)(u_0y_{11})(u_0y_3)(u_0y_0)(u_1y_{32})(u_1y_{20})(u_0y_3)(u_0y_{12})\Big)^*\Big\}$. Es de notar que $\mathcal{L}^o \subset \mathcal{L}(Q)$.

La generación de secuencia de eventos que no están incluidas en el lenguaje del sistema, debe ser evitado sobretodo sí el generador va a ser usado como modelo para diagnóstico. Para identificación es conveniente asumir la función φ isomórfica. La asignación isomórfica dará lugar a equivalencias entre los símbolos de salida y las marcas alcanzadas. Por lo tanto, será inmediato inferir el estado de la red a partir de una combinación particular de símbolos de salida $\varphi(m[P]) = y_k$.

4.2. Modelo del Sistema

Las PN ordinarias no permiten caracterizar eventos, ni aplicar conceptos de temporización propios de los sistemas industriales; por lo tanto, con ellas sólo es posible describir la estructura lógica del sistema impidiendo un modelado adecuado. Además las PN no tienen señales explícitas de E/S externas en su definición, [31].

Una propuesta interesante para solucionar este inconveniente es modelar el comportamiento interno del sistema como una PN y que este módulo interactúe con su entorno a través de las señales del sistema y no a partir de las marcas de la PN, con dos tipos de señales discretas: Condiciones de señales y señales de eventos;

a este tipo de modelado se le conoce como NCES (Net Condition Event Systems).

En el presente estudio se propone una forma de modelar SED inspirada en algunos conceptos del formalismo de modelado por NCES, específicamente en condiciones de E/S y los eventos con el fin de etiquetar las transiciones y lugares; pero se añaden propiedades para generar un enfoque de modelado de SED con el propósito de realizar diagnóstico.

La propuesta de modelado se ha llamado st-IPN que significa "Red de Petri Interpretada Temporizada Estocástica", la cual utiliza las señales de E/S para la definición de las transiciones y los lugares, evita el no-determinismo que se observó en la Figura 4.1.2 y en propuestas de modelos y define la temporización en las transiciones como funciones de densidad probabilidad.

Las características agregadas al concepto básico de PN son: en primer lugar la adición de etiquetas en las transiciones y en las funciones de salida de los lugares, con el objeto de relacionarlos con las señales de E/S y generar una interpretación en el comportamiento y cambio de estado del modelo y en segundo lugar se tiene en cuenta el tiempo en el disparo de las transiciones y el comportamiento de este tiempo se asume bajo una función de densidad de probabilidad no establecida, es decir puede ser normal, o exponencial, o etc; por este motivo se remarca que sea "temporizada - estocástica".

Entonces, a partir de las IPN definidas en la Definición 9, se asignan las señales de control como funciones de entrada asociadas con las transiciones y las señales de los sensores se representan por funciones de salida al marcado alcanzado; además, se adiciona un atributo de tiempo variable, constituyendo las st-IPN en redes con lugares asociados a un estado del proceso; las transiciones se interpretan como condiciones o eventos que en caso de ser verdaderos

o de ocurrir, obligan al sistema a cambiar de un estado a otro con un tiempo de procesamiento probabilista; así, una st-IPN tiene la capacidad de modelar las secuencias de órdenes dadas a los actuadores y a las señales del sensor emitidas cada vez que se alcanza un nuevo estado. Todo esto permitirá dar interpretación física al modelo, que conducirá a una mejor aplicación a procesos industriales reales.

El modelo propuesto pretende servir de modelo de referencia para el diagnóstico en SED estocásticos y además busca ser determinista. Para ser determinista construye una función de salida isomórfica que garantiza determinismo en el generador, adicionando un elemento que dotará a la st-IPN de memoria en la función de salida. Se trata del *diferencial del símbolo de salida*, (dy). dy representa el comportamiento histórico de las señales de salida, permitiendo un mejor entendimiento del estado del proceso.

Definición 21 (Diferencial de símbolo de salida, dy)**.** Dados dos símbolos de salida $y_j, y_{j-1} \in Y$, generados en los instantes τ_i y τ_{i-1} respectivamente, dy_j se define como:

$dy_j(y_j \times y_{j-1}) \rightarrow \{-1, 0, 1\}$, donde $dy_j = y_j - y_{j-1}$ con $dy_j \in \{-1, 0, 1\}$; así los posibles valores de dy_j son: $0 \times 0 \rightarrow 0$; $0 \times 1 \rightarrow 1$; $1 \times 0 \rightarrow -1$; $1 \times 1 \rightarrow 0$.

Esto significa que, dado que se trabaja con señales binarias, dy representa el cambio del valor de la señal de salida, de 0 a 1, o de 1 a 0; por lo tanto, dy representa el comportamiento histórico de las señales de un instante τ_{i-1} a un instante τ_i. De esta manera la función de salida está compuesta por el valor de las señales de salida y_j y su diferencial dy_j, es decir y_j/dy_j. Esta condición de la función de salida, genera una característica particular de las señales, la cual se ha denominado característica de direccionalidad de las señales.

Por otro lado, se ha incluido el tiempo de permanencia en las transiciones, el cual es el tiempo que transcurre desde entre dos eventos consecutivos, [96]. Además, es importante tener en cuenta que los sistemas reales se ven afectados por una cantidad de aspectos tales como: la variabilidad en el flujo de material, la fluctuación en el suministro de energía, el deterioro de las máquinas e instrumentos, etc. Esta variabilidad implica que el tiempo entre eventos varíe, así como de un modo de operación a otro. Por esta razón, la st-IPN que se propone, incluye una función de densidad de probabilidad para el tiempo de disparo de transiciones para cada transición y para cada modo de operación.

Teniendo en cuenta el diferencial y empleando funciones temporales de densidad de probabilidad en las transiciones, una st-IPN se define de la siguiente manera:

Definición 22 (Red de Petri Interpretada Temporizada Estocástica, st-IPN)**.** Una st-IPN es una estructura representada por: $stQ = (Q, \Omega, \delta, OM)$. Donde $Q = (N, U_o, Y, \lambda, \varphi)$ es una IPN, N tienen el mismo significado que en la Definición 9, $U_o = \left\{u_0, \cdots, u_{|2^{m_o}|-1}\right\}$ es el alfabeto de entrada con u_s como un símbolo de entrada y m_o el número de entradas observables, $Y = \left\{y_0, \cdots, y_{|2^n|-1}\right\}$ es el alfabeto de salida con y_j como un símbolo de salida y n el número de salidas, $\lambda : TR \rightarrow U_o \times \delta$ es una función de etiquetado que asigna un símbolo de entrada y una función de densidad de probabilidad a cada transición y φ es la función de salida definida como:$\varphi : (RQ, M_0) \rightarrow Y/dY$, φ es isomórfica sobre Y/dY.

$\Omega := (U_o \times Y) .\delta$ es el alfabeto del sistema. Un símbolo del alfabeto es $\omega^i = (u_s y_j) .f\left(t_{r,o_{om}}\right)$, donde $(u_s y_j)$ es el símbolo de E/S, $f\left(t_{r,o_{om}}\right)$ es la función de densidad de probabilidad del tiempo de la transición r, asociada al símbolo de E/S, en el modo de operación om.

$\delta := TR \times OM \rightarrow f(t_{TR \times OM})$ es una función de asignación de las

funciones de densidad de probabilidad del tiempo de disparo de cada transición, por cada modo de operación.

$OM = \left\{o_0, \cdots o_{|2^{n_{ec}}|-1}\right\}$ es el conjunto de modos de operación.

Dado un evento $\omega^i = (u_s y_j) .t^{ev}_{o_{om}}$, una transición $tr_r \in TR$ con $\lambda(tr_r) = u_s.f\left(t_{r,o_{om}}\right)$ está habilitada en o_{om}, si $\forall p_q \in \bullet tr_r$, $m\left(p_q\right) \geq I\left(p_q, tr_r\right)$ y si $\left(a \leq t^{ev}_{o_{om}} \leq b\right)$ siendo $\int_a^b f\left(t_{r,o_{om}}\right) \geq (1-\alpha)$, donde $1-\alpha$ es el nivel de confiabilidad y $[a,b]$ el intervalo de confianza.

Si $f\left(t_{r,o_{om}}\right) = N\left(\mu, \sigma\right)$, entonces $a = \mu - z_{\alpha/2} * \sigma$ y $b = \mu + z_{\alpha/2} * \sigma$. Una tr_r está habilitada sí cada lugar de entrada cumple con los requerimientos de marcado y si $t^{ev}_{o_{om}} \in [a,b]$.

Si $\lambda\left(tr_r\right) = u_s.f\left(t_{r,o_{om}}\right)$ está presente y si tr_r está habilitada, entonces tr_r se dispara y se alcanza un nuevo marcado M_{k+1} calculado a partir de la Ecuación 2.3.1; pero, $y_j/dy_j = \varphi\left(M_k\right)$, así cada lugar de la st-IPN representa no solo el estado actual, sino que también incluye información de direccionalidad, por lo tanto la red evoluciona a partir de la ecuación de estado:

$$\begin{array}{rcl} M_{k+1} & = & M_k + I.\overrightarrow{tr_r} \\ y_j/dy_j & = & \varphi\left(M_k\right) \end{array} \tag{4.2.1}$$

Definición 23 (Lenguaje de disparo de una st-IPN)**.** Sea σ la secuencia de disparo de transiciones de una st-IPN, (stQ) tal que $\sigma \in \mathcal{L}_F(stQ)$, (Definición 11), cuyo disparo genera una secuencia s, tal que $s \in \Omega^*, s = \omega^0, \cdots, \omega^k$, a instantes $\tau_0 \leq \cdots \leq \tau_k$, siendo $\omega^i = (u_s y_j) .f\left(t_{r,o_{om}}\right)$ donde $u_s \in U_o$, $y_j \in Y$ y $f\left(t_{r,o_{om}}\right)$ es la función de densidad de probabilidad del tiempo en la tr_r en el o_{om}. El conjunto de todas las secuencias de disparo de transiciones $\sigma \in \mathcal{L}_F\left(Q\right)$, cuya probabilidad sea mayor a $(1-\alpha)$ es el lenguaje

de disparo de la st-IPN así:

$$\mathcal{L}_F\left(stQ\right) = \{\sigma \in \mathcal{L}_F(Q) : Prob\left(\sigma|m_0\right) \geq (1-\alpha)\} \qquad (4.2.2)$$

Y el lenguaje generado por una st-IPN es $\mathcal{L}(stQ) = \{s|s \in \Omega^*\}$.

A continuación se relacionan las secuencias de símbolos de E/S temporizados con el disparo de secuencias de transiciones y las secuencias de marcado generadas, para definir el lenguaje de entrada y el lenguaje de salida de una st-IPN.

Definición 24 (Lenguaje de entrada y Lenguaje de salida de una st-IPN)**.** Sea σ la secuencia de disparo de transiciones de una st-IPN, tal que $\sigma \in \mathcal{L}_F(stQ)$, (Ecuación 4.2.2), el lenguaje de entrada se define como la secuencia de las funciones de etiquetado de las transiciones que pertenecen a $\mathcal{L}_F(stQ)$, es decir $\mathcal{L}_{in}(stQ) = \{\sigma|\sigma = \lambda\left(tr_1\right)\cdots\lambda\left(tr_k\right)\}$ y el lenguaje de salida son las secuencias de marcado alcanzadas $\mathcal{L}_{out}\left(stQ\right) = \{\varphi\left(m\left(p_0\right)\right)\cdots\varphi\left(m\left(p_k\right)\right)\}$.

Por lo tanto $\mathcal{L}\left(stQ\right) = \left\{\omega^0, \cdots, \omega^k\right\}$, con $\omega^i = \left(u_s y_j\right).f\left(t_{r,o_{om}}\right)$, comprende un lenguaje de entrada y un lenguaje de salida.

Aplicando la Definición 6, el lenguaje de entrada se define como: $\mathcal{L}_{in}\left(stQ\right) = \left\{P_{U_o.\delta}\left(s\right)|s \in \mathcal{L}(stQ)\right\}$, entonces $\mathcal{L}_{in}\left(stQ\right) = \left\{u_s^0.f\left(t_{r,o_{om}}\right), \cdots, u_s^k.f\left(t_{r,o_{om}}\right)\right\}$ y el lenguaje de salida como: $\mathcal{L}_{out}\left(stQ\right) = \left\{P_Y\left(s\right)|s \in \mathcal{L}\left(stQ\right)\right\}$, entonces $\mathcal{L}_{out}\left(stQ\right) = \left\{y_j^0, \cdots, y_j^k\right\}$.[1]

Definición 25 (Post-lenguaje)**.** Dada una secuencia de eventos temporizados estocásticos $s = \omega^0, \cdots, \omega^k$, el post-lenguaje de s será: $st^\bullet = \mathcal{L}(stQ)/s = \{t \in \Omega^*|s \in \mathcal{L}\left(stQ\right)\}$.

[1]Nota: los superíndices se adicionan con el objeto establecer el orden de los eventos

4.2.1. Modelado de Sistemas basado en st-IPN

Como una st-IPN es un generador de lenguaje legal, ésta se construye a partir del lenguaje legal ordenado del sistema $\mathcal{L} = \left\{\omega^0, \cdots, \omega^k\right\}$.

El modelo inicia con cero transiciones, $ntr = 0$ y con un lugar, $np = 1$, (p_0). Se toma ω^0 y se calcula y_j como $y_j = Pc_Y(\omega^0)$ y $dy_j = \vec{0}$ como una cadena de ceros de tamaño n (número de salidas); por lo tanto, la función de salida en p_0 es $\varphi(m(p_0)) = y_j/dy_j$.

Luego de manera ordenada se toman uno a uno los eventos ω^i de $\mathcal{L}$, se calcula la función de salida como: $y_j/y_j^i - y_j^{i-1}$, se verifica si existe un lugar con esa función de salida, si no existe se genera un nuevo lugar p_{q+1} con una función de salida $\varphi(m(p_{q+1})) = y_j/y_j^i - y_j^{i-1}$ y también se genera una nueva transición $ntr = ntr + 1$ con $\lambda(tr_{ntr}) = Pc_U(\omega^i)$, si por el contrario existe, se verifica si existe la transición que va hacia el lugar ya modelado, para ello se analizan los vectores $\overrightarrow{pre}$, $\overrightarrow{post}$ y $\lambda(tr_{ntr})$, si no existe se añade una nueva transición.

Luego se actualizan las matrices *Pre y Post*; este procedimiento conforma el Algoritmo 4.1. (*np*: número de lugares, *np′*: índice temporal de lugar, *ntr*: número de transiciones, *ntr′*: índice temporal de transición, *pa*: lugar actual.

El resultado del algoritmo es la estructura de la red que genera el mismo lenguaje que se pretende modelar.

Proposición 1. *Sea* $\mathcal{L}^o = \left\{\omega^0 \cdots \omega^k\right\}$ *el lenguaje temporizado legal de un sistema a instantes* $\tau_0 \leq \cdots \leq \tau_k$*, si stQ es una st-IPN construida a partir del Algoritmo 4.1, entonces* $\mathcal{L}^o = \mathcal{L}(stQ)$.

Demostración. Dadas s, s_1 dos secuencias de eventos temporizados legales; tal que: $s = \omega^0 \cdots \omega^k$ y $s_1 = \omega^p$ donde $s, s_1 \in \mathcal{L}^o$,

Algoritmo 4.1 Construcción de una st-IPN

Entradas: $\mathcal{L} = \left\{ s \in (U_o.Y)^* .\delta \right\}$, con $s = \omega^0, \cdots, \omega^k$ y $\omega^i = (u_s, y_j) .f\left(t_{om}^{ev}\right)$.
Salidas: stQ.
Condiciones iniciales: $i = 0$; $\omega^0 = (u_s, y_j) .f\left(t_{o_{om}}^0\right)$, $\varphi\left(m\left(p_0\right)\right) = y_j/dy_j$; $dy_j = \overrightarrow{0}_n$.
Variables: $np = 1$, np', $ntr = 0$,ntr', $Pre = []$, $Post = []$, $pa = 0$;

1. $i = i + 1$; $\omega^i = (u_s, y_j) .f\left(t_{om}^{ev}\right)$; $np^{'} = np$; lugar temporal.
2. $\varphi\left(m\left(p_{np'}\right)\right) = y_j^i / \left[y_j^i - y_j^{i-1}\right]$; función de salida en el lugar temporal.
3. Si $\varphi\left(m\left(p_{np'}\right)\right) \neq \varphi\left(m\left(p_q\right)\right)$ $\forall q$, $q = 0 \cdots np - 1$, entonces;
   - *a*) $ntr = ntr + 1$; nueva transición con $\lambda\left(tr_{ntr}\right) = u_s.f\left(t_{om}^{ev}\right)$;
     1) Calcular vectores *pre* y *post* para tr : $np^{'} = np$; $\overrightarrow{pre} = ceros(np)$; $pre(pa) = 1$; $\overrightarrow{post} = ceros(np)$; $post(np) = 1$;
     2) Actualizar Pre y $Post$: $Pre = Pre + \overrightarrow{pre}$; $Post = Post + \overrightarrow{post}$.
     3) $\varphi\left(m\left(p_{np}\right)\right) = \varphi\left(m\left(p_{np'}\right)\right)$; $pa = np$;
   - *b*) De lo contrario, (el lugar ya existe).
     1) $pa = k$; $ntr^{'} = ntr + 1$; se genera una transición temporal con $\lambda\left(tr_{ntr'}\right) = u_s.f\left(t_{om}^{ev}\right)$;
     2) Calcular los vectores *pre* y *post* para la tr temporal: $\overrightarrow{pre} = ceros(np)$; $pre(np) = 1$; $\overrightarrow{post} = ceros(np)$; $post(k) = 1$;
     3) Para $r = 1 \cdots ntr$; Si $\overrightarrow{pre} = Pre(:, r) \wedge \overrightarrow{post} = Post(:, r) \wedge \lambda\left(tr_{ntr'}\right) = \lambda\left(tr_r\right)$;
        - a' entonces $tr\_existe = 1 \wedge$ saltar a paso 5;
        - b' de lo contrario $tr\_existe = 0$;
     4) Si $tr\_existe = 1$, entonces $Pre = Pre$; $Post = Post$;
        - a' de lo contrario; $ntr = ntr^{'}$, número de tr temporal.
     5) Actualizar Pre y$Post$: $Pre = Pre + \overrightarrow{pre}$; $Post = Post + \overrightarrow{post}$; $\lambda\left(tr_{ntr}\right) = u_s.f\left(t_{om}^{ev}\right)$;
   - *c*) fin condicional
4. ejecutar paso 1.

$s_1 \in \mathcal{L}^o/s$ y $s \in \mathcal{L}(stQ)$. Asumiendo que $s_1 \notin \mathcal{L}(stQ)$, entonces siendo el estado actual de la st-IPN $\varphi(m(p_k)) = y_j^k / \left[y_j^k - y_j^{k-1}\right]$ y dado el evento $\omega^p = (u_s, y_j).f\left(t^p_{o_{om}}\right)$ al aplicar el Algoritmo 4.1 un estado es alcanzado en $\varphi(m(p_{q*})) = y_j^p / \left[y_j^p - y_j^k\right]$; si $\exists\varphi(m(p_q)) \mid \varphi(m(p_q)) = \varphi(m(p_{q*}))$, $\forall q$, $q = 1 \cdots k$, entonces $p_{q*} \in P$; de lo contrario, p_{q*} es un lugar nuevo y se genera una transición con $\lambda(tr_{r*}) = u_s^p.f\left(t^p_{r*,o_{om}}\right)$, es decir el evento $\omega^p \in \mathcal{L}(stQ)/s$ y se contradice la suposición; si p_{q*} no es nuevo, $p_{q*} = p_q$ siendo q el índice del lugar repetido, se genera una transición $\lambda(tr_{r*}) = u_s^p.f\left(t^p_{r*,o_{om}}\right)$ si $\lambda(tr_{r*}) = \lambda(tr_r)$, entonces $\omega^p \in \mathcal{L}(stQ)/s$ y se contradice la suposición; de lo contrario tr_{r*} es una nueva transición con $\lambda(tr_{r*}) = u_s^p.f\left(t^p_{r*,o_{om}}\right)$ y $\omega^p \in \mathcal{L}(stQ)/s$ y se contradice la suposición. Lo que permite concluir que sí $s_1 \in \mathcal{L}^o$ y $\mathcal{L}^o$ es legal entonces $s_1 \in \mathcal{L}(stQ)$ por lo tanto $\mathcal{L}^o = \mathcal{L}(stQ)$ y la proposición queda demostrada. □

4.2.2. Propiedades de una st-IPN y de su Lenguaje

Dada una secuencia de eventos s, donde $s = (u_k y_k), (u_x, y_x), (u_1, y_1), \cdots, (u_v, y_v), (u_x y_x), (u_1 y_3)$, con $y_k \neq y_v$. Si φ es una función de salida isomórfica entre el conjunto de marcas y el conjunto de símbolos de salida, entonces la red que genera la secuencia es no-determinística porque el símbolo de entrada u_1 puede ser seguido por dos diferentes símbolos de salida y_3 y y_x. Una de las consecuencias de que una red sea no-determinística es que genere secuencias de eventos que no hacen parte del lenguaje del sistema. La generación de cadenas de eventos no incluidas en el lenguaje, debe ser eliminada sí el generador va a ser usado como modelo para diagnóstico. La proposición 2 prueba que una st-IPN es determinística.

Proposición 2. *Sea stQ una st-IPN definida como en la Definición 22, si φ es isomórfica, entonces stQ es determinística.*

Demostración. Sea s una secuencia de eventos, como la descrita en el párrafo anterior. Dividiendo el símbolo de salida y_x en dos: $y_x^{'}$ $y_x^{''}$, $s = (u_k y_k), (u_x y_x^{'}), (u_1 y_1), \cdots (u_v y_v), (u_x y_x^{''}), (u_1 y_3)$; el diferencial de símbolos de salida en $y_x^{'}$ es $dy_x^{'} = y_x^{'} - y_k$ y en $y_x^{''}$ es $dy_x^{''} = y_x^{''} - y_v$; como $y_k \neq y_v$ entonces $dy_x^{'} \neq dy_x^{''}$; por lo tanto, la función de salida es diferente en los dos estados resultantes de la división de y_x, así se elimina el no-determinismo, incrementando el número de estados y φ es isomórfica entre el conjunto de marcas y el conjunto de símbolos de salida y_j/dy_j. □

La inclusión del concepto de dy en el modelo del sistema permite definir su número de estados máximo.

Definición 26 (Número máximo de estados, stQ_{max})**.** El estado de un sistema está determinado por los símbolos de salida y su comportamiento histórico; por lo tanto, el número máximo de estados identificables está relacionado con sus señales de salida, así:

$$stQ_{max} = 2^n \times 3^n \tag{4.2.3}$$

donde n es el número de símbolos de salida.

Por ejemplo, en un sistema con un solo sensor binario, $stQ_{max} = |2|^1 \times |3|^1 = 6$, un sistema con dos sensores binarios será: $stQ_{max} = |2|^2 \times |3|^2 = 36$.

Proposición 3. *Sea stQ una st-IPN y $\mathcal{L}(stQ)$ el lenguaje generado por stQ y sea $s = \omega^i \cdots \omega^k$ una secuencia de eventos a instantes $\tau_i, \cdots \tau_k$. Sí $s \in \mathcal{L}(stQ)$ y $\varphi(m(p_i)) = \varphi(m(p_k))$, entonces $s^* \in \mathcal{L}(stQ)$, es decir, s modela un ciclo.*

Demostración. Como una st-IPN es determinística (ver Proposición 2), si $\varphi(m(p_i)) = \varphi(m(p_k))$, entonces $m(p_i) = m(p_k)$, así s puede ser generada infinitamente. □

Proposición 4. *Sean stQ_1 y stQ_2 dos st-IPNs, con matrices de incidencia I_1 y I_2 y sus marcas iniciales $m(p_0^1)$ y $m(p_0^2)$, y sean $\mathcal{L}(stQ_1)$, $\mathcal{L}(stQ_2)$ los lenguajes generados por stQ_1 y stQ_2 respectivamente con $\varphi_1(m(p_0^1)) = \varphi_2(m(p_0^2))$ y $m(p_0^1) = m(p_0^2)$. Entonces $\mathcal{L}(stQ_1) = \mathcal{L}(stQ_2)$ sí $I_1 = I_2$.*

Demostración. El sistema evoluciona a partir de la ecuación de estado (Ecuación 4.2.1).

Suponiendo que $\mathcal{L}(stQ_1) = \mathcal{L}(stQ_2) = \mathcal{L}(stQ)$, pero $I_1 \neq I_2$, existe una secuencia $s = \omega^0, \cdots, \omega^k \in \mathcal{L}(stQ)$ a instantes $\tau_0 \leq \cdots \leq \tau_k$, que es generada por una secuencia de disparo $\sigma = tr_1 \cdots tr_k$.

Sí $I_1 \neq I_2$, entonces $\exists k > 0 \mid m(p_0^1) \overset{tr_1}{\rightarrow} \cdots \overset{tr_k}{\rightarrow} m(p_k^1)\,;\, m(p_0^2) \overset{tr_1}{\rightarrow} \cdots \overset{tr_k}{\rightarrow} m(p_k^2)$, para cada st-IPN respectivamente. $\mid m(p_i^1) = m(p_i^2)$ pero, $m(p_k^1) \neq m(p_k^2)$ y como φ is isomórfica entonces $\varphi(m(p_k^1)) \neq \varphi(m(p_k^2))$.

Siendo $\mathcal{L}(stQ_1) = \mathcal{L}(stQ_2)$, entonces $\mathcal{L}_{out}(stQ_1) = \mathcal{L}_{out}(stQ_2)$.

Si $\mathcal{L}_{out}(stQ_1) = \{\varphi(m(p_0^1)), \varphi(m(p_1^1)), \cdots, \varphi(m(p_k^1))\}$ y $\mathcal{L}_{out}(stQ_2) = \left\{\varphi(m(p_0^2)),\ \varphi(m(p_1^2)), \cdots,\ \varphi(m(p_k^2))\right\}$, entonces $\varphi(m(p_0^1)) = \varphi(m(p_0^2)) \cdots\ \varphi(m(p_k^1)) = \varphi(m(p_k^2))$.

Por lo tanto, $m(p_k^1) = m(p_k^2)$, con lo cual $I_1 = I_2$, lo cual contradice la condición inicial. □

Esto indica que la st-IPN que genera un lenguaje es única y la secuencia de disparo es fija.

Proposición 5. *Sea $\varphi : R(N, M_0) \rightarrow Y/dY$ una función de salida y sea $s = \omega^i \cdots \omega^k$ tal que $s \in \mathcal{L}(stQ)$ una secuencia que modela*

un ciclo $(\varphi(m(p_i)) = \varphi(m(p_k)))$*, entonces* $\sum_i^k dy_j = 0$ *para cada* $j = 1, \cdots, n$*, donde* n *es el número de sensores.*

Demostración. El sistema evoluciona a partir de los estados de la st-IPN, (Ecuación 4.2.1):

$s = \left\{\omega^i \cdots \omega^k\right\} = \{\lambda(tr_i)\varphi(m(p_i)) \cdots \lambda(tr_k)\varphi m(p_k)\}$ observada a instantes $\tau_i, \cdots, \tau_k$; entonces $\mathcal{L}_{out}(stQ) = \{\varphi(m(p_i)), \cdots, \varphi(m(p_k))\}$ $\mathcal{L}_{out}(stQ) = \{y_i/dy_i, \ldots, y_k/dy_k\}$.

De acuerdo a la condición que en un ciclo $\varphi(m(p_i)) = \varphi(m(p_k))$, $(i \neq k)$, el comportamiento de cada señal puede obedecer a: *i)* No hay cambios, entonces, el valor de dy_j es siempre cero durante el ciclo; por lo tanto, $\sum_i^k dy_j = 0$; *ii)* cambia, entonces, el valor de $\varphi(m(p_i))$ puede ser 1 o 0: Si inicia en 1, el valor a lo largo del ciclo será $1 \to 0 \to 1$ por lo tanto, $dy_j : 0 \to -1 \to 1 : \Sigma dy_j = 0$. Si inicia en 0; el valor a lo largo del ciclo será $0 \to 1 \to 0$ por lo tanto, $dy_j : 0 \to 1 \to -1 : \Sigma dy_j = 0$. Es decir, después de $2n * +1$ pasos ($n*$: número de salidas cuyos valores han cambiado), el valor de las salidas debe retornar a su valor inicial cuando se alcanza $m(p_k)$. □

4.2.3. Observabilidad

La observabilidad implica la posibilidad de detectar si un evento no-observable ha ocurrido o no, analizando la secuencia de eventos generados.

El conjunto de eventos de un sistema está compuesto por eventos observables y eventos no-observables: $\Omega = \Omega_o \cup \Omega_{uo}$. Un evento temporizado es de la forma: $\omega^i = (u_s y_j).t_{o_{om}}^{ev}$. La proyección basada en la Definición 1, $P_{\Omega_o}(\Omega)$, elimina los Ω_{uo} de una secuencia de eventos.

Un evento no-observable se genera cuando existe una entrada no-observable a la planta. Este evento puede obedecer a una entrada en funcionamiento normal pero no observable o una entrada de fallo, cuya ocurrencia provoca que el sistema tenga un funcionamiento no deseado.

Definición 27 (Evento *n-observable*)**.** Un evento ω es n-observable con respecto a una proyección $P(\Omega)$ si $\forall t \in \omega t^{\bullet}$, donde $|t| > n$, $P(s\omega t) \neq P(st)$.

Es decir, un evento ω es n-observable si puede ser detectado con certeza dentro de un número n, de eventos observables después de su ocurrencia. Esto implica que un evento ω es observable si para cada traza s de eventos que finalicen en ω, existe una traza t, de tamaño n; tal que, la secuencia st no genere el mismo registro de eventos observables conteniendo a ω.

Esta definición implica eventos ordenados sin tener en cuenta el tiempo que transcurre entre ellos. Para el enfoque del presente trabajo es importante tener en cuenta el tiempo, puesto que se modelan procesos estocásticos.

Para realizar test de observabilidad en sistemas estocásticos con base en su lenguaje, a continuación se propone una definición de observabilidad temporal.

Definición 28 (Observabilidad temporal)**.** Sea s y t secuencias de eventos, tal que $s \in \mathcal{L}(stQ)$ y t pertenezca al post-lenguaje de s, $st^{\bullet} = \mathcal{L}(stQ)/s$. Dado un evento temporizado $\omega^p = (u_s y_j).t^p_{o_{om}}$ con $\mathcal{L}_{in}(\omega^p) = u^p_s.t^p_{o_{om}}$ y $\mathcal{L}_{out}(\omega^p) = y^p_j$; ω^p es observable si $Pt_{U_o}(s\omega^p t) \neq Pt_{U_o}(st)$ o $Pt_Y(s\omega^p t) \neq Pt_Y(st)$, en n pasos después de su ocurrencia ($n \leq |t|$).

Donde Pt_{U_o} y Pt_Y se calculan con base en la Definición 6.

Para ilustrar la importancia del tiempo entre eventos, en un test de observabilidad se presenta el siguiente ejemplo:

Ejemplo 5. Sea un proceso de llenado de un tanque que opera en un solo modo de funcionamiento, el cual está compuesto por dos válvulas: una de entrada y otra de salida, dos sensores que miden la altura mínima y máxima del contenido del tanque. Se asume que existe un caudal de entrada que no se encuentra controlado y una entrada a la planta que se considera un fallo cuando el tanque se está vaciando.

Las órdenes de control son: $Cc = \{cc_1, cc_2\}$ con $cc_1 = v_e, \bar{v_e}$ y $cc_2 = v_s, \bar{v_s}$. $EnExP^{uo} = \{euop_1, euop_2\}$, donde $euop_1 = f_1, \bar{f_1}$ y $euop_2 = q_e, \bar{q_e}$.

Por lo tanto, las entradas a la planta son: $U = \{euop_1, euop_2, cc_1, cc_2\}$ y las salidas son las lecturas sensoriales: $Y = \{sr_1, sr_2\}$, con $sr_1 = s_1, \bar{s_1}$ y $sr_2 = s_2, \bar{s_2}$.

El alfabeto de entrada observable es $U_o = u_o, u_1, u_2, u_3$ y $Y = y_o, y_1, y_2, y_3$ es el alfabeto de salida.

Sea s_1 una secuencia de eventos del sistema, tal que: $s_1 \in \mathcal{L}$; $s_1 = \omega^0 \cdots \omega^5$, donde $s_1 = (u_0, y_0).t^0\ (u_2, y_2).t^1\ (u_2, y_3).t^2\ (u_1, y_2).t^3\ (u_1, y_0).t^4\ (u_0, y_0).t^5$ es una secuencia normal de llenado y vaciado del tanque.

Sea $s_2 \in \mathcal{L}$; una secuencia de eventos para el llenado del tanque, asumiendo que existe el caudal de entrada adicional: $s_2 = \omega^0 \cdots \omega^7$, donde $s_2 = (u_0, y_0).t^{0'}\ (u_2, y_2).t^{1'}\ (u_6, y_2).t^{2'}\ (u_2, y_2).t^{3'}\ (u_2, y_3).t^{4'}\ (u_1, y_2).t^{5'}\ (u_1, y_0).t^{6'}\ (u_0, y_0).t^{7'}$, donde $(u_6, y_2).t^{2'}$ es el evento no-observable relacionado con $euop_2$.

Aplicando la Definición 1 sobre los eventos observables sin importar el tiempo:

$P(s_1) = (u_0, y_0)\ (u_2, y_2)\ (u_2, y_3)\ (u_1, y_2)\ (u_1, y_0)\ (u_0, y_0)$ y

$P(s_2) = (u_0, y_0)\ (u_2, y_2)\ (u_2, y_3)\ (u_1, y_2)\ (u_1, y_0)\ (u_0, y_0)$; las proyecciones son iguales por lo tanto el evento no-observable no se pueden detectar.

Teniendo en cuenta el tiempo, aplicando la Definición 6:

$Pt\,(s_1) = (u_0, y_0)\,.t^0\ (u_2, y_2)\,.t^1\ (u_2, y_3)\,.t^2\ (u_1, y_2)\,.t^3\ (u_1, y_0)\ .t^4$ $(u_0, y_0)\ .t^5$ y t^2 es el tiempo de llenado.

$Pt\,(s_2) = (u_0, y_0)\,.t^{0'}\ (u_2, y_2)\,.t^{1'}\ (u_2, y_3)\,.\left(t^{2'} + t^{3'} + t^{4'}\right)(u_1, y_2)\,.t^{5'}$ $(u_1, y_0)\,.t^{6'}\ (u_0, y_0)\,.t^{7'}$; $t^{0'} = t^0$; $t^{1'} = t^1$; $t^{5'} = t^3$; $t^{6'} = t^4$; $t^{7'} = t^5$; $\left(t^{2'} + t^{3'} + t^{4'}\right)$ es el tiempo de llenado; $\left(t^{2'} + t^{3'} + t^{4'}\right) < t^2$ puesto que existe un caudal adicional en un periodo de tiempo; por lo tanto, las proyecciones no son iguales puesto que $(u_2, y_3)\,.t^2 \neq (u_2, y_3)\,.\left(t^{2'} + t^{3'} + t^{4'}\right)$, entonces se puede detectar el evento no-observable cuando se proyecta incluyendo el tiempo.

Por otro lado, es importante analizar el comportamiento estadístico del tiempo. Sí el tiempo de llenado es una función de densidad de probabilidad $t^2 \sim f\,(t^2)$, y se ha definido un nivel de confiabilidad $1-\alpha$, se genera un rango de confianza $[a, b]$ dentro del cual un nuevo tiempo se considera aceptable como comportamiento normal. Suponiendo que $t^2 \sim N\,(2, 0{,}05)$ con un nivel de confiabilidad del 95 %; entonces $a = 1{,}902$ y $b = 2{,}98$; si $a \leq \left(t^{2'} + t^{3'} + t^{4'}\right) \leq b$, el evento no-observable se puede detectar; de lo contrario no.

Por lo tanto, la detección de un evento no-observable, depende de la variabilidad del tiempo asociado a cada evento y de la duración del evento no-observable, a mayor variabilidad y menor duración, menor posibilidad de detectarlo.

Ahora, suponiendo una entrada de fallo $euop_1$, cuando el tanque se está vaciando, una secuencia de eventos podría ser: $s_3 = (u_0, y_0)\,.t^0\,(u_2, y_2)\,.t^1\,(u_2, y_3)\,.t^2\,(u_1, y_2)\,.\ t^3\,(u_1, y_2)\,.\,(t^4 \sim \propto)$, t^4 tiende a $\propto$ cuando la estrategia de control se basa en las respuestas de los sensores o $s_4 = (u_0, y_0)\,.t^0\ (u_2, y_2)\,.t^1\ (u_2, y_3)\,.t^2\ (u_1, y_2)\,.t^3$ $(u_1, y_2)\,.t^{*4}\ (u_0, y_2)\,.t^{*5}$, cuando la estrategia de control se basa en

tiempo (t^{*4}) para emitir la orden de control, siendo $(u_1, y_2)\,.t^{*4}$ el evento de fallo relacionado con la entrada no-observable $euop_1$.

$Pt\,(s_3) = (u_0, y_0)\,.t^0\ (u_2, y_2)\,.t^1\ (u_2, y_3)\,.t^2\ (u_1, y_2)\,.\ (t^4 \sim\infty)$; como $Pt\,(s_1) \neq Pt\,(s_3)$, el fallo se puede detectar.

$Pt\,(s_4) = (u_0, y_0)\,.t^0\ (u_2, y_2)\,.t^1\ (u_2, y_3)\,.t^2\ (u_1, y_2)\,.t^3\ (u_1, y_2)\,.t^{*4}$ $(u_0, y_2)\,.t^{*5}$. $Pt\,(s_1) \neq Pt\,(s_4)$; por lo tanto, el fallo se puede detectar.

Este análisis permite inferir respecto a las condiciones para que un evento no-observable sea observado en una st-IPN, lo cual se demuestra en la Proposición 6.

Proposición 6. *Sea $\mathcal{L}^o$ el lenguaje de un sistema y stQ una st-IPN que representa a $\mathcal{L}^o$. Dada una secuencia s de eventos temporizados tal que, $s \in \mathcal{L}(stQ)$ y sea $\omega^k \in \Omega_{uo}$, $\omega^p = \left(u_s^p, y_j^p\right).\left(t_{o_{om}}^p\right)$[2] tal que, $\omega^p \in \mathcal{L}/s$; ω^p es observable si $u_s^p \in \mathcal{L}_{in}(stQ)$ y $y_j^p \notin \mathcal{L}_{out}(stQ)$ o si $u_s^p \notin \mathcal{L}_{in}(stQ)$ y $y_j^p \in \mathcal{L}_{out}(stQ)$ o si $u_s^p \in \mathcal{L}_{in}(stQ)$, $y_j^p \in \mathcal{L}_{out}(stQ)$ y $t_{o_{om}}^p \notin [a, b]$ donde $\int_a^b f\,(t^a) \geq (1 - \alpha)$.*

Demostración. Sea $\mathcal{L} = s\omega^p t$ el lenguaje del sistema, donde $s = \omega^0 \cdots \omega^k \omega^p$, $t = \omega^{p+1}$ y $\omega^p = \left(u_s^p, y_j^p\right).t_{o_{om}}^p$ es un evento no-observable, tal que $s \in \mathcal{L}\,(stQ)$ y $t \in \mathcal{L}\,(stQ)\,/s$; entonces $Pt_{U_o}\,(st) = u_s^0.t_{o_{om}}^0 \cdots u_s^k.t_{o_{om}}^k\, u_s^{p+1}.t_{o_{om}}^{p+1}$ y $Pt_Y\,(st) = y_j^0 \cdots y_j^k y_j^{p+1}$; dado $\omega^p \in \mathcal{L}$ si:

i) u_s^p es observable y y_j^p es no-observable, $Pt_{U_o}\,(\omega^p) = u_s^p.t_{o_{om}}^p$ y $Pt_Y\,(\omega^p) = \varepsilon$; entonces, $Pt_{U_o}\,(st) = Pt_{U_o}\,(s\omega^p t)$ y $Pt_Y\,(st) \neq Pt_Y\,(s\omega^p t)$ de acuerdo a la Definición 28, ω^p es observable.

ii) u_s^p es no-observable y y_j^p es observable, $Pt_{U_o}\,(\omega^p) = \varepsilon$ y $Pt_Y\,(\omega^p) = y_j^p$; entonces, $Pt_{U_o}\,(st) \neq Pt_{U_o}\,(s\omega^p t)$ y $Pt_Y\,(st) = Pt_Y\,(s\omega^p t)$ de acuerdo a la Definición 28, ω^p es observable.

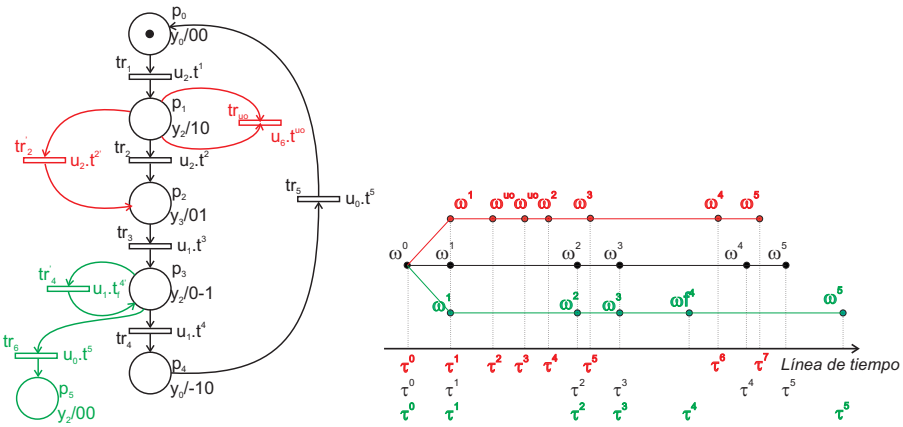
[2]Se ha añadido el superíndice en los símbolos del lenguaje, para identificar el evento al que pertenecen

iii) u_s^p es no-observable y y_j^p es no-observable, $Pt_{U_o}(\omega^p) = \varepsilon$ y $Pt_Y(\omega^p) = \varepsilon$; entonces, si $t_{o_{om}}^p \notin [a, b]$ ω^p es observable; de lo contrario ω^p es no-observable. Por lo tanto, la proposición queda demostrada. □

A continuación se explica la prueba de observabilidad temporal a partir de la comparación del lenguaje normal con lenguajes que contengan eventos no-observables legales o por fallo en el Ejemplo 5.

Dadas las secuencias s_1, s_2, s_4, donde la secuencia s_1 representa el comportamiento normal, la secuencia s_2 representa el comportamiento con un evento no-observable legal y la secuencia s_4 es el comportamiento con evento de fallo.

La Figura 4.2.1a, presenta la st-IPN para el comportamiento normal del Ejemplo 5; pero se han añadido los eventos no-observables para analizar el concepto de observabilidad de una st-IPN. El evento no-observable legal, $euop_2$ se muestra en color rojo y el fallo eup_2, en color verde.



(a) Modelo del tanque **(b)** Trayectorias del lenguaje del sistema

Figura 4.2.1: Comportamientos del tanque

Se puede observar que el evento no-observable legal, altera el tiempo de disparo de la transición tr_2 y el fallo no permite medir el instante en el cual el tanque se termina de vaciar y la red permanece en el estado «vaciando» sin evolucionar y si existe una estrategia del controlador de cerrar la válvula la red evolucionará a un estado no normal (p_5).

En la Figura 4.2.1b, se describen las trayectorias del lenguaje para el comportamiento normal (en negro), con evento no-observable legal (en rojo) y con fallo (en verde) del sistema, proyectadas en una línea de tiempo.

El tiempo de los eventos se calcula a partir de $t^{ev}_{o_{om}} = |\tau_i| - |\tau_{i-1}|$ (Ver Definición 4). Para s_1 $\delta = \{t^0, t^1, t^2, t^3, t^4, t^5\}$ y los tiempos calculados en función de s_1 para s_2 y s_4 son: para $s_2 = \left\{t^0, t^1, t^{uo}, t^{*2}, t^3, t^4, t^5\right\}$, $(t^{uo} = |\tau_3| - |\tau_2|)$, $(t^{*2} = t^2 - t^{uo})$ y para $s_4 = \left\{t^0, t^1, t^2, t^3, t^{*4}_f, t^{*5} \geq t^5\right\}$ $t^{4'}_f = |\tau_4| - |\tau_3|$.

La proyección de los lenguajes de entrada y salida sobre cada trayectoria son:

$Pt_{U_o}(s_1) = u_0.t^0u_2.t^1u_2.t^2u_1.t^3u_1.t^4u_0.t^5$, $Pt_Y(s_1) = y_0y_2y_3y_2y_0y_0$.

$Pt_{U_o}(s_2) = u_0.t^0u_2.t^1u_2.t^{*2}u_1.t^3u_1.t^4u_0.t^5$, $Pt_Y(s_1) = y_0y_2y_3y_2y_0y_0$.

Si $t^{*2} \notin [a, b]$, donde $\int_a^b f(t^2) \geq (1 - \alpha)$ entonces, $Pt_{U_o}(s_1) \neq Pt_{U_o}(s_2)$ y $Pt_Y(s_1) = Pt_Y(s_2)$. El evento $euop_1$ se puede detectar en un paso $(n = 1)$ después de su ocurrencia.

$Pt_{U_o}(s_4) = u_0.t^0u_2.t^1u_2.t^2u_1.t^3u_1.t^{*4}u_0.t^{*5}$, $Pt_Y(s_1) = y_0y_2y_3y_2y_2y_2$.

Si $t^{*4} \notin [a, b]$, donde $\int_a^b f(t^4) \geq (1 - \alpha)$ o $t^{*5} \notin [a, b]$ con $\int_a^b f(t^5) \geq (1 - \alpha)$, entonces $Pt_{U_o}(s_1) \neq P_{U_o}(s_4)$ y $P_Y(s_1) \neq P_Y(s_4)$. El fallo en el sensor se detecta en un paso $(n = 1)$. Es de aclarar que si el controlador no emite la orden de cerrar la válvula de salida, el fallo no puede ser detectado puesto que la red no evolucionaría hacia un estado observable.

A partir de las Proposiciones 1 y 6, se resuelven las preguntas planteadas para el modelado del sistema, puesto que se comprueba que una st-IPN modela el lenguaje legal de un SED y su lenguaje permite detectar algunos eventos no-observables legales y algunos eventos de fallo. Por lo tanto, con base en el modelo de un SED temporizado estocástico representado como una st-IPN se puede realizar diagnóstico de fallos.

4.2.4. Ejemplo de Modelado de un sistema

Ejemplo 6. Modelo del sistema de tanques acoplados como st-IPN. Modelando el ejemplo de la Figura 4.1.2a, como una st-IPN, la red resultante se muestra en la Figura 4.2.2.

Como se puede observar la red mostrada en la Figura 4.2.2 es más compleja que la presentada en la Figura 4.1.2; pero la st-IPN modela solo los comportamientos legales observados; es decir, genera el lenguaje observado, muestra información en los estados respecto al comportamiento del sistema y además es una red determinista.

4.3. Definición de un Sistema de Gran Escala

Un sistema de gran escala, es un sistema compuesto por una gran cantidad de dispositivos distribuidos independientes o no, que generan concurrencia de estados a partir de la ocurrencia de eventos de manera asíncrona. La generación de un modelo único en estos sistemas ha llevado a que se presente una explosión en el número de estados, lo cual conlleva una serie de inconvenientes al realizar análisis sobre ellos test de observabilidad, o para detectar comportamientos no normales. Para reducir la complejidad, muchos

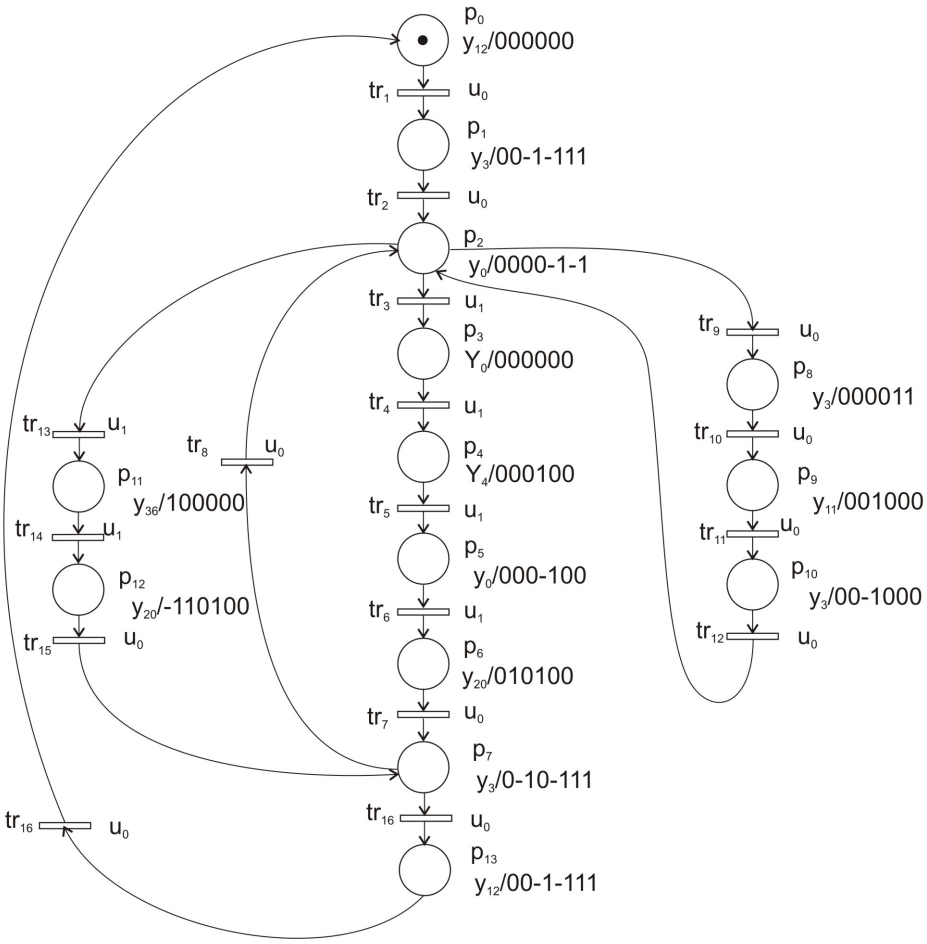


Figura 4.2.2: st-IPN para tanques acoplados

enfoques de modelado han propuesto la división del sistema en partes más pequeñas, [44].

En esta propuesta, el modelado de este tipo de sistemas se realiza a partir de la división en subsistemas, donde un subsistema es una parte del sistema que posee un comportamiento particular. Esta división es llevada a cabo de acuerdo al diseño físico o a criterios de las relaciones de funcionamiento de los dispositivos y las señales de E/S, coherentes con ese funcionamiento en cada subsistema.

La estructura a considerar, de un SED de gran escala, se muestra en la Figura 4.3.1.

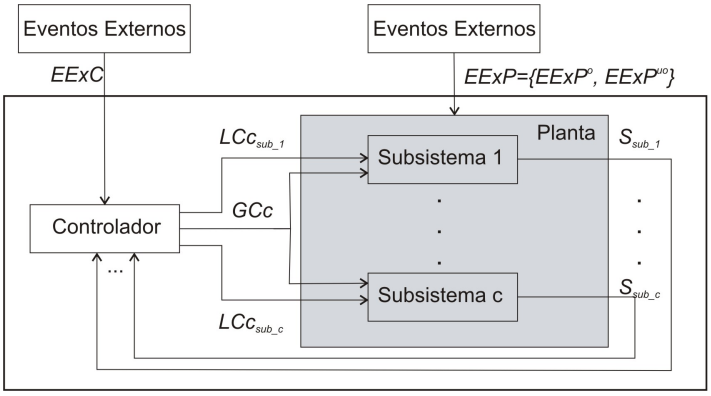


Figura 4.3.1: Esquema de un sistema de gran escala

Las entradas al SED de gran escala son: $Entradas = \{EnExC, EnExP, EnInP\}$; $EnExC$ y $EnExP$ tienen el mismo significado que para sistemas sencillos.

Para la aplicación del método de modelado a este tipo de sistemas, se han dividido las señales de entrada internas a la planta ($EInP$), en dos subgrupos: $EInP = \{GCc, LCc\}$; GCc está compuesto por órdenes de control globales que afectan a más de un subsistema, donde $GCc = \{gcc_1, \cdots, gcc_{n_{gc}}\}$, n_{gc} es el número de

órdenes de control globales y el conjunto $LCc = \cup LCc_l$, donde LCc_l, está compuesto por órdenes de control que solo afectan el subsistema l, con $l = 1 \cdots c$, c es el número de subsistemas. Entonces, $LCc_l = \left\{cc_{l,1}, \cdots cc_{l,n_{l,cc}}\right\}$, $n_{l,cc}$ es el número de órdenes de control locales en el subsistema l.

El conjunto de salidas es: $Salidas = Y = \cup S_l$, donde $S_l = \left\{sr_{l,n_l}, \cdots, sr_{l,n_{l,sr}}\right\}$, es decir, contiene las lecturas sensoriales asignadas al subsistema l y $n_{l,sr}$ es el número de lecturas sensoriales en el subsistema l.

El conjunto de entradas es $U \subset Entradas$ donde, $U = \{EExP^{uo}, EExP^{o}, GCc, LCc\}$, GCc, LCc,$EnExP^{o}$ constituyen las entradas observables y $EExP^{uo}$ las entradas no-observables.

Las entradas a cada subsistema l son $U_l \subset U$, donde $U_l = \{ExP^{uo}, ExP^{o}, GCc, LCc_l\}$, entonces U_l es el conjunto de todas las combinaciones posibles de entradas al subsistema l. El número total de entradas, no-observables y observables en el subsistema l es: $m_l = m_{l_{uo}} + m_{l_o}$, donde: $m_{l_o} = n_{eop} + n_{gc} + n_{l,cc}$. Una entrada a un subsistema l en particular, $u_{l,s}$, es una cadena de valores binarios, tal que:

$$u_{l,s} = \left[euop_1 \cdots euop_{n_{euop}} eop_1 \cdots ep_{n_{eop}} gcc_1 \cdots gcc_{n_{gc}} cc_{l,1} \cdots cc_{l,n_{cc}}\right] \tag{4.3.1}$$

siendo s la representación decimal de la cadena.

Las salidas de cada subsistema son $Y_l \subset Salidas$, donde $Y_l = \{Sr_l\}$; entonces Y_l es el conjunto de todas las combinaciones posibles de salidas del subsistema l. Una salida de un subsistema l en particu-

lar, $y_{l,j}$, es una cadena de valores binarios, tal que:

$$y_{l,j} = [sr_{l,1} \cdots sr_{l,n_l}] \tag{4.3.2}$$

siendo j una representación decimal de la cadena.

Esta clasificación de señales de E/S permite generar modelos en cada subsistema, lo cual evita la explosión combinacional de estados en sistemas de gran escala. Esta descomposición modular presenta ventajas, puesto que se modelan comportamientos de estados concurrentes en diferentes subsistemas, de recursos compartidos por los subsistemas o funcionamiento colaborativos entre sistemas.

Esta es una ventaja importante cuando el objetivo del modelo es realizar diagnóstico de fallos, como en la presente propuesta.

4.3.1. Modelo del SED de Gran Escala

El modelo global (modelo del sistema) es un conjunto de modelos locales (modelos de los subsistemas) que se representan a partir de st-IPNs y éstas se construyen a partir de su lenguaje observable con base en el Algoritmo 4.1.

Definición 29 (st-IPN para un subsistema)**.** Cuando se modela un subsistema se agrega el subíndice l, a la st-IPN presentada en la Definición 22, correspondiente al subsistema a modelar, es decir: $stQ_l = (Q_l, \Omega_l, \delta_l, OM)$. Con $Q_l = (N_l, U_{ol}, Y_l, \lambda_l, \varphi_l)$; $U_l = \left\{ u_{l,0}, \cdots, u_{l,\left\lvert 2^{(m_{lo})} \right\rvert - 1} \right\}$; $Y_l = \left\{ y_{l,0}, \cdots, y_{l,|2^{n_l}|-1} \right\}$; $\lambda_l : TR_l \rightarrow U_l \times \delta_l$ es la función de etiquetado en cada transición del subsistema l, φ_l es la función de salida definida como: $\varphi_l : R\left(Q_l, M_{l,0}\right) \rightarrow$

$y_{l,j}/dy_{l,j}$; φ_l es isomórfica sobre $y_{l,j}/dy_{l,j}$; $\Omega_l = U_l \times Y_l.\delta_l$ es el alfabeto del subsistema; $\delta_l := TR_l \times OM \rightarrow f(t_{TR \times OM})$ es una función de asignación de las funciones de densidad del tiempo de disparo de cada transición por cada modo de operación de OM en el subsistema l[3]. $OM = \left\{o_0, \cdots o_{|2^{nec}|-1}\right\}$ es el conjunto de modos de operación, n_{ec} es el número de entradas externas al controlador.

4.3.1.1. Evento temporizado en subsistemas

Un evento temporizado en un subsistema l es de la forma: $\omega_l^i = (u_{l,s} y_{l,j}) .t_{l,o_{om}}^{ev}$, i identifica el instante de tiempo en el cual se genera el evento, l es el número del subsistema, $u_{l,s}$ y $y_{l,j}$ se calculan a partir de la Ecuación 4.3.1 y la Ecuación 4.3.2, $t_{l,o_{om}}^{ev}$ es el tiempo transcurrido entre los eventos *ev* y *ev-1* en el subsistema l, para el modo de operación o_{om}. La Figura 4.3.2 explica la relación del tiempo en los subsistemas frente al sistema global.

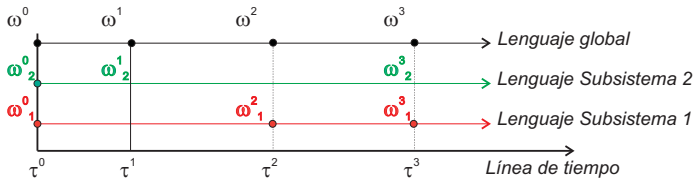


Figura 4.3.2: Lenguajes temporizados

Como se observa en la Figura 4.3.2, en la línea en negro se presenta una secuencia de eventos $\omega^0\omega^1\omega^2\omega^3$ del sistema global en instantes $\tau_0 \leq \cdots \leq \tau_3$; en la línea roja la secuencia de eventos: $\omega_1^0\omega_1^2\omega_1^3$ del subsistema 1 y en verde una secuencia de eventos: $\omega_2^0\omega_2^1\omega_2^3$ del subsistema 2.

[3] $tr_{l,r}$, indica la transición r en el subsistema l. $p_{l,q}$, indica el lugar q en el subsistema l.

El tiempo en los eventos de los subsistemas es: $t^{ev}_{l,o_{om}} = |\tau_{ev\_actual}| - |\tau_{ev\_anterior}|$. Por ejemplo para el evento $\omega^3_2 = (u_{2,s}y_{2,j})\,.t^2_{2,o_{om}}$ el tiempo asociado a este evento será $t^2_{2,o_{om}} = |\tau_3| - |\tau_1|$.

Además, es importante tener en cuenta que cuando hay cambios de órdenes de control o de lecturas sensoriales en más de un subsistema en el mismo instante de tiempo, el evento es generado en todos los subsistemas involucrados.

4.3.2. Operaciones entre Modelo Global y Modelos Locales

El modelo de un sistema complejo presentado como un conjunto de st-IPNs, las cuales representan diversos comportamientos o modos de funcionamiento tiene ventajas frente a un modelo del sistema global, puesto que son más fáciles de interpretar debido su estructura más reducida y porque modelan solo lenguajes activos; por lo tanto, permite establecer en un instante de tiempo, cuál subsistema funciona.

4.3.2.1. Lenguaje local a partir del lenguaje global

El lenguaje de los subsistemas se puede hallar a partir de la operación de proyección de lenguajes definida en la Definición 6.

Dado el lenguaje de un sistema $\mathcal{L} = \omega^0 \cdots \omega^k$, con $\omega^i = (u_s y_j)\,.t^{ev}_{o_{om}}$. Ordenando las señales de E/S: $\omega^i = (u_{1,s} \cdots u_{c,s})\,(y_{1,s} \cdots y_{c,s})\,.t^{ev}_{o_{om}}$ la proyección es: $P_{\Omega_l}(\omega^i) = (u_{l,s}y_{l,j})\,.t^{ev}_{l,o_{om}}$, si ha existido un evento temporizado en *l,* basado en Definición *6;* de lo contrario $P_{\Omega_l}(\omega^i) = \varepsilon.t^{ev}_{l,o_{om}}$ (Definición 6). $u_{l,s}$ y $y_{l,j}$ se organizan bajo la Ecuación 4.3.1 y la Ecuación 4.3.2 y $t^{ev}_{l,o_{om}}$ se halla como en la Definición 6.

Por lo tanto, el lenguaje de cada subsistema se puede hallar a partir del lenguaje global.

El lenguaje del subsistema 1, representado en la Figura 4.3.2, a partir de la proyección sobre lenguaje global es:

$Pt_{\Omega_o}(\omega_1^0\omega_1^2\omega_1^3) = (u_{1,s}y_{1,j}).t^0_{o_{om}}\varepsilon.t^1_{o_{om}}(u_{1,s}y_{1,j}).t^2_{o_{om}}(u_{1,s}y_{1,j}).t^3_{o_{om}}$, donde $\omega_1^0 = (u_{1,s}y_{1,j}).t^0_{l,o_{om}}$, $\omega_1^2 = (u_{1,s}y_{1,j}).t^1_{l,o_{om}}$ con $t^1_{l,o_{om}} = t^1_{o_{om}} + t^2_{o_{om}}$ y $\omega_1^3 = (u_{1,s}y_{1,j}).t^2_{l,o_{om}}$.

4.3.2.2. Lenguaje global a partir de lenguajes locales

Un evento del sistema global ω^i, en el instante τ_i, se obtiene a partir de la *Operación de splice sincronizado temporizado*, definida a continuación.

Definición 30 (Operación de Splice Sincronizado Temporizado)**.** Sea Q un sistema compuesto por c subsistemas. Sea ω_l^i los eventos en el instante τ_i, con $l = 1 \cdots c$, donde $\omega_l^i = (u_{l,s}y_{l,j})\, t^{ev}_{l,o_{om}}$. Un evento del sistema global, en el instante τ_i, puede ser obtenido como:

$$\omega^i = \omega_1^i \oplus \cdots \oplus \omega_c^i \tag{4.3.3}$$

donde $\omega_1^i \oplus \cdots \oplus \omega_c^i = (u_{1,s}y_{1,j}) \oplus \cdots \oplus (u_{c,s}y_{c,j}) = (u_{1,s} \oplus \cdots \oplus u_{c,s})$ $(y_{1,j} \oplus \cdots \oplus y_{c,j})$ y $(u_{1,s} \oplus \cdots \oplus u_{c,s}) = u_{s_g}$, donde u_{s_g} es una representación en binario de s_g, s_g representa todos los símbolos de entrada y $(y_{1,j} \oplus \cdots \oplus y_{c,j}) = y_{j_g}$, donde y_{j_g}es una representación en binario de j_g, j_g representa todos los símbolos de salida. Por lo tanto, $\omega^i = \left(u_{s_g}y_{j_g}\right).t^{ev}_{o_{om}}$ donde $t^i_{o_{om}} = min\left(t^h_{1,o_{om}} \cdots t^h_{c,o_{om}}\right)$.

Entonces, el lenguaje del sistema global se puede construir a partir de la secuencia de eventos sincronizados de sus subsistemas.

Proposición 7. *Dado un sistema global Q, compuesto por c subsistemas, el lenguaje del sistema global $\mathcal{L}(stQ)$ construido por la*

sincronización temporal de los lenguajes de sus subsistemas es igual al lenguaje del sistema, $\mathcal{L}(stQ) = \mathcal{L}(Q)$.

Demostración. Sea Q un sistema de gran escala con c subsistemas. Sea $\mathcal{L}(Q) = \omega^0\omega^1 \cdots \omega^e$ una secuencia de eventos generada por el sistema Q y sea $\mathcal{L}(stQ)$ el lenguaje del sistema global construido por la sincronización de los lenguajes de sus subsistemas, $\mathcal{L}(stQ_l)$ con $l = 1 \cdots c$.

Sea $s = \omega^0\omega^1 \cdots \omega^e$ una secuencia de eventos, tal que $s \in \mathcal{L}(Q)$ y sea $s_1 = \omega^{*0} \cdots \omega^{*e}$ una secuencia de eventos sincronizados de $\mathcal{L}\left(stQ_l\right)$ con $l = 1 \cdots c$.

Asumiendo que $\omega^e \neq \omega^{*e}$.

$P_U\left(s\right) = \{u_s^0 \cdots u_s^e\}$, $u_s^e = \{u_{1,s}, \cdots, u_{c,s}\}$ y $P_Y\left(s\right) = \left\{y_j^0 \cdots y_j^e\right\}$. Como $\omega_l^i = (u_{l,s}y_{l,j})$ para $l = 1 \cdots c$. $\omega_l^e = (u_{l,s}, y_{l,j})$; $\omega_l^e = (u_{1,s}, y_{1,j})$ $\cdots (u_{c,s}, y_{c,j})$. Basado en la Definición 30, $\omega^{*e} = \omega_1^e \oplus \cdots \oplus \omega_c^e$ entonces $\omega^e = \omega^{*e}$, lo cual contradice la condición inicial. □

Por lo tanto el lenguaje del sistema global $\mathcal{L}$, se construye por la sincronización temporal de los lenguajes de los subsistemas $\mathcal{L}_l$ que está compuesto.

Nota: El lenguaje del sistema global se puede reconstruir **si y solo si** se realiza el splice sincronizado temporizado sobre **todos** los subsistemas que integran el sistema global.

4.3.2.3. st-IPN global a partir de st-IPN locales

El modelo del sistema global representado como una st-IPN se puede hallar a partir del Producto Síncrono de st-IPNs, definido a continuación.

Definición 31 (Producto Síncrono de st-IPNs)**.** Dado un conjunto stQ, compuesto por c st-IPNs, $stQ = \{stQ_1, \cdots, stQ_c\}$, cada una definida como en la Definición 22, el producto síncrono de stQ es una st-IPN expresada como:

$$\begin{array}{ll} stQs = & \|_{l=1}^{c} stQ_l \\ stQs = & (Qs, \Omega s, \delta s) \end{array} \tag{4.3.4}$$

donde $Qs = (Ns, Us, Ys, \lambda s, \varphi s)$. $Ns = (Ps, TRs, Pre, Pos, M_0)$ es una PN ordinaria, el conjunto de lugares es: $Ps = \{P1 \times P2 \times \cdots \times Pc\}$, el conjunto de transiciones es $TRs = \left\{2^{|P_s|-1}\right\}$ con$|P_s|$ como el tamaño del conjunto de lugares, c es el número de subsistemas, $Pre : Ps \times TRs \rightarrow Z^+$, $Pos : Ps \times TRs \rightarrow Z^+$); y $Ms_0 = M1_0 \times M2_0 \times \cdots \times Mc_0$, es el marcado inicial. $Us = \{U_1 \cup U_2 \cup \cdots U_c\}$ es el conjunto de entradas al sistema. $Ys = \{Y_1 \cup Y_2 \cup \cdots Y_c\}$ es el conjunto de salidas del sistema. $\lambda s : TRs \rightarrow Us.\delta s$ es la función de etiquetado de las transiciones y φs se define como: $\varphi s : R(Ns, Ms_0) \rightarrow y_{l,j}/dy_{l,j}$; φs siendo isomórfica sobre $y_{l,j}/dy_{l,j}$.

$\Omega s = (Us \times Ys)\,.\delta s$ es el alfabeto del sistema.

$\delta s := TRs \times OM \rightarrow f(t_{TRs \times OM})$ es el conjunto de funciones de densidad de probabilidad para cada modo de operación que pertenece a OM.

Al dividir un sistema en subsistemas, el modelo conserva las propiedades en cuanto a representatividad y a observabilidad; es decir, el modelo de un subsistema representado como una st-IPN es un generador de lenguaje determinístico, capaz de modelar comportamientos estocásticos y de detectar algunos eventos no-observables, bajo las condiciones expuestas en la Proposición 6.

Teorema 1. *Sea $\mathcal{L}(stQ)$ el lenguaje global compuesto por c lenguajes locales, $\mathcal{L}(stQ_l)$. Un evento $\omega^i \in \mathcal{L}(stQ)$ es observable en el sistema global sii $\omega_l^i \in \mathcal{L}(stQ_l)$ es observable en algún subsistema l, con $l = 1 : c$.*

Demostración. Condición necesaria: si un evento ω^i es observable en el sistema global, entonces el evento ω_l^i es observable en el subsistema l.

Suponiendo que ω^i es observable; pero $\exists p \in [1 \cdots c] \mid \omega_p^i$ es no-observable. Como $\omega^i = (u_s y_j) .t_{o_{om}}^{ev} = (u_{1,s} \cdots u_{p,s} \cdots u_{c,s})$ $(y_{1,s} \cdots y_{p,s} \cdots y_{c,s}) .t_{o_{om}}^{ev}$ es observable; $\exists u_{p,s} \wedge y_{p,j}$ tal que $P_{\Omega_l}(\omega_l^i) = (u_{p,s} y_{p,j})$ porque el lenguaje del sistema global se reconstruye a partir de todos los lenguajes locales, entonces $\exists, l, \mid P_{\Omega_l}(\omega_l^i) \neq \varepsilon$. Por lo tanto ω_p^i es observable.

Condición suficiente: si ω_l^i es un evento observable en el subsistema l en algún $l = 1 : c$, entonces ω^i es observable en el sistema global.

Asumiendo que ω_p^i es observable, para algún $p \in [1 \cdots c]$, pero en ω^i no. $\omega^i = \omega_1^i \oplus \cdots \oplus \omega_p^i \oplus \cdots \oplus \omega_c^i$ y $\omega_p^i \subset \omega^i$; por lo tanto, ω_p^i es observable en ω^i, lo cual contradice la suposición y el teorema queda demostrado. □

4.4. Aplicación a un Sistema Robótico

Con el objeto de aplicar la presente propuesta de modelado, se ha escogido el sistema presentado en [37], el cual no es un sistema de gran escala; pero sí es un sistema que permite mostrar las ventajas de la propuesta.

El sistema es un brazo robótico para recoger piezas en las posiciones $pp1$, $pp2$ o $pp3$ y colocarlas en $pp0$. Está compuesto por tres

cilindros, dos horizontales y uno vertical; los cilindros horizontales se extienden o retraen dependiendo de la orden del controlador para ubicar la pieza. El cilindro vertical se desplaza y recoge las piezas por succión a partir de una sola orden de control. Los sensores son: uno para cada posición de los cilindros horizontales, otro incorporado en la unidad de succión, sensores de posición de las piezas y otro para detección de la pieza recogida. En la Figura 4.4.1 se presentan diferentes configuraciones para la interrelación de los componentes mecatrónicos.

Las señales externas al controlador $EExC$, que definen los modos de operación son las señales de presencia de pieza en las posiciones 1, 2, y 3; es decir, $EExC = \{pp_1, pp_2, pp_3\}$, donde $pp_i = \{1, 0\}$ (presencia o no presencia en posición $i = 1 : 3$). El brazo robótico comienza a trabajar cuando se detecta al menos una pieza; por lo tanto, los modos de funcionamiento son 7, $OM = \{o_1, \cdots, o_7\}$. La combinación o_0 no es posible porque no existiría funcionamiento. Por ejemplo $o_1 = [001]$ es el comportamiento del sistema en el cual hay una pieza en la posición 3 ($pp_3 = 1$).

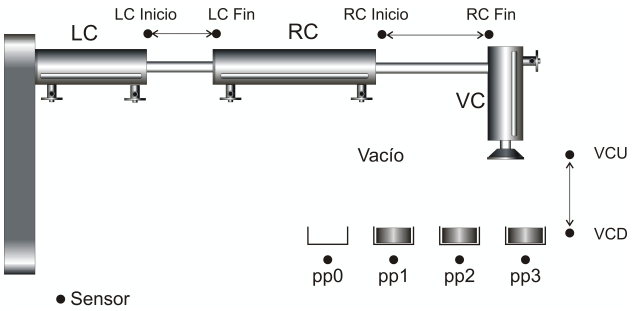


Figura 4.4.1: Brazo robótico

Aunque el sistema no es un sistema de gran escala, con el objeto de aplicar el método propuesto, se ha dividido en subsistemas con base en el número de dispositivos mecatrónicos presentes, así: El

subsistema 1 corresponde al cilindro horizontal de la izquierda, el cual tiene dos sensores de posición: $sr_{1,1} = Lc_{inicio}$ y $sr_{1,2} = Lc_{fin}$ y dos órdenes de control: $cc_{1,1} = Ext_{LC}$ y $cc_{1,2} = Ret_{LC}$. El subsistema 2 corresponde al cilindro horizontal de la derecha, tiene dos sensores de posición: $sr_{2,1} = Rc_{inicio}$ y $sr_{2,2} = Rc_{fin}$ y dos órdenes de control: $cc_{2,1} = Ext_{RC}$ y $cc_{2,2} = Ret_{RC}$. El subsistema 3 corresponde al cilindro vertical, el cual tiene dos sensores de posición: $sr_{3,1} = Vc_{up}$ y $sr_{3,2} = Vc_{down}$, un sensor que indica succión o no: $sr_{3,3} = vacío$ y dos órdenes de control, una para que el cilindro baje $cc_{3,1} = Ext_{VC}$ y $cc_{3,2} = Vac$ que es la orden para vacío del succionador, para recoger no es necesario orden de control puesto que cuando $cc_{3,1}$ no está activa, el cilindro se recoge por la compresión de un resorte.

Se desea modelar el comportamiento del sistema cuando está en modo de operación o_1. Los símbolos se ordenan como $u_{1,s} = [cc_{1,1}cc_{1,2}]$, $u_{2,s} = [cc_{2,1}cc_{2,2}]$, $u_{3,s} = [cc_{3,1}cc_{3,2}]$ y $y_{1,j} = [sr_{1,1}sr_{1,2}]$, $y_{2,j} = [sr_{2,1}sr_{2,2}]$, $y_{3,j} = [sr_{3,1}sr_{3,2}sr_{3,3}]$.

Los lenguajes en cada subsistema $\mathcal{L}_l$, para el modo de operación o_1 son:

$\mathcal{L}_1 = (u_{1,0}, y_{1,0}) .t^0_{1,o_1} (u_{1,2}, y_{1,2}) .t^1_{1,o_1} (u_{1,2}, y_{1,3}) .t^2_{1,o_1} (u_{1,1}, y_{1,2}) .t^3_{1,o_1}$ $(u_{1,1}, y_{1,0}) .t^4_{1,o_1} (u_{1,0}, y_{1,0}) .t^5_{1,o_1}$ en $\tau_0, \tau_1, \tau_2, \tau_{13}, \tau_{14}, \tau_{15}$ instantes.

$\mathcal{L}_2 = (u_{2,0}, y_{2,0}) .t^0_{2,o_1} (u_{2,2}, y_{2,2}) .t^1_{2,o_1} (u_{2,2}, y_{2,3}) .t^2_{2,o_1} (u_{2,1}, y_{2,2}) .t^3_{2,o_1}$ $(u_{2,1}, y_{2,0}) .t^4_{2,o_1} (u_{2,0}, y_{2,0}) .t^5_{2,o_1}$ en $\tau_0, \tau_3, \tau_4, \tau_{10}, \tau_{11}, \tau_{12}$ instantes.

$\mathcal{L}_3 = (u_{3,0}, y_{3,0}) .t^0_{3,o_1} (u_{3,2}, y_{3,4}) .t^1_{3,o_1} (u_{3,2}, y_{3,6}) .t^2_{3,o_1} (u_{3,3}, y_{3,7}) .t^3_{3,o_1}$ $(u_{3,1}, y_{3,5}) .t^4_{3,o_1} (u_{3,1}, y_{3,1}) .t^5_{3,o_1} (u_{3,3}, y_{3,5}) .t^6_{3,o_1} (u_{3,3}, y_{3,7}) .t^7_{3,o_1}$ $(u_{3,3}, y_{3,6}) .t^8_{3,o_1} (u_{3,2}, y_{3,6}) .t^9_{3,o_1} (u_{3,0}, y_{3,4}) .t^{10}_{3,o_1} (u_{3,0}, y_{3,0}) .t^{11}_{3,o_1}$ en τ_0, τ_5, τ_6, τ_7, τ_8, τ_9, τ_{16}, τ_{17}, τ_{18}, τ_{19}, τ_{20}, τ_{21} instantes.

Estos lenguajes representan el comportamiento del sistema. En el instante τ_0, los tres subsistemas están en reposo (el valor de las lecturas de los sensores es de cero y no hay órdenes de control), en el instante τ_1 se genera un evento en el subsistema 1, de extender

el cilindro horizontal izquierdo y la respuesta del sensor $sr_{1,1}$ es inmediata, por lo tanto, el evento en τ_1 es: $(u_{1,2}, y_{1,2})\,.t^1_{1,o_1}$; al cabo de un tiempo, en el instante τ_2 se genera un evento en el subsistema 1 denotado como $(u_{1,2}, y_{1,3})\,.t^2_{1,o_1}$, luego se genera un evento en el subsistema 2 denotado como $(u_{2,2}, y_{2,2})\,.t^1_{2,o_1}$ y así sucesivamente.

Considerando variabilidad en el sistema, el tiempo de los eventos se ajustan a una función de densidad de probabilidad, $t^{ev}_{l,o_{om}} \backsim f\,(t)$. Por ejemplo: $\omega^4_3 = (u_{3,2}, y_{3,6})\,.t^2_{3,o_1}$, el tiempo $t^2_{3,o_1} \sim N\,(2, 0{,}1)$ y para $1 - \alpha = 0{,}95$, el intervalo de confianza será $[1{,}804 - 2{,}196]$.

Para representar este comportamiento se construyen las st-IPNs para cada subsistema, aplicando el Algoritmo 4.1. Cada subsistema inicia con un lugar, cuya función de salida está representada por las lecturas de los sensores, y_j, en τ_0 y el diferencial dy_j como cadenas de ceros; es decir $y_{1,0}/00$, lo cual se muestra en la Figura 4.4.2.

Figura 4.4.2: Proceso de modelado del brazo robótico a partir del lenguaje: estados iniciales

En τ_1 el subsistema 1 evoluciona a otro estado, para representar esta evolución se añade una transición con $\lambda\,(tr_{1,1}) = u_{1,2}.t^1_{1,o_1}$ y se alcanza un nuevo lugar con función de salida $y_{1,2}/10$, $(dy_j = 10 - 00)$; los otros subsistemas no evolucionan en este instante de tiempo, este comportamiento se puede observar en la Figura 4.4.3.

Al final, las st-IPNs que modelan el comportamiento de los tres subsistemas se muestran en la Figura 4.4.4.

Analizando la st-IPN para el cilindro horizontal izquierdo, se observa que se modelan comportamientos que permiten establecer si el cilindro se está extendiendo o retrayendo, en los lugares $p_{1,1}$ y $p_{1,3}$

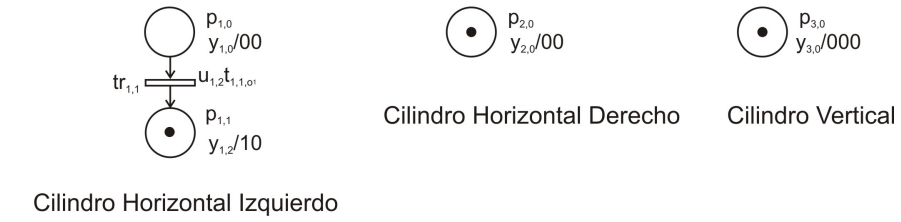


Figura 4.4.3: Proceso de modelado del brazo robótico a partir del lenguaje: paso1

respectivamente; ésta es una ventaja que tiene la presente propuesta frente a un modelo por autómatas; además se puede observar que en las etiquetas de las transiciones se incluye el tiempo para el modo de funcionamiento o_1; pero si se modelan más comportamientos, se adicionaría una tabla para discriminar los tiempos, ya que la red sería la misma.

4.4.1. Evolución de la red

Para explicar la evolución de la st-IPN, se ha tomado como referencia el modelo resultante del subsistema 3. Siendo el estado actual del subsistema, el estado 1, es decir $M_{3,1} = [01000000000000]$, con $\varphi\left(m\left(p_{3,2}\right)\right) = y_{3,4}/100$; dado un evento $\omega_3^6 = \left(u_{3,2}, y_{3,6}\right).t_{3,o_1}^2$, si $\left(u_{3,2}, y_{3,6}\right) \in \mathcal{L}(stQ_3)\ /\omega_3^0\omega_3^5$ y si $Prob\left(1{,}804 \leq t_{3,o_1}^2 \leq 2{,}196\right) \geq 1-\alpha$, se dispara $tr_{3,2}$ con $\lambda_3\left(tr_{3,1}\right) = u_{3,2}.N\left(2,0{,}1\right)$ y con función de salida $\varphi\left(m\left(p_{3,2}\right)\right) = y_{3,4}/100$; alcanzando una nueva marca: $M_{3,2} = [00100000000000]$.

El lenguaje generado por las st-IPNs es el mismo que el de los respectivos subsistemas; además las redes son determinísticas puesto que a partir de un lugar p_q si se dispara una transición tr_r la marca alcanzada por este disparo es única.

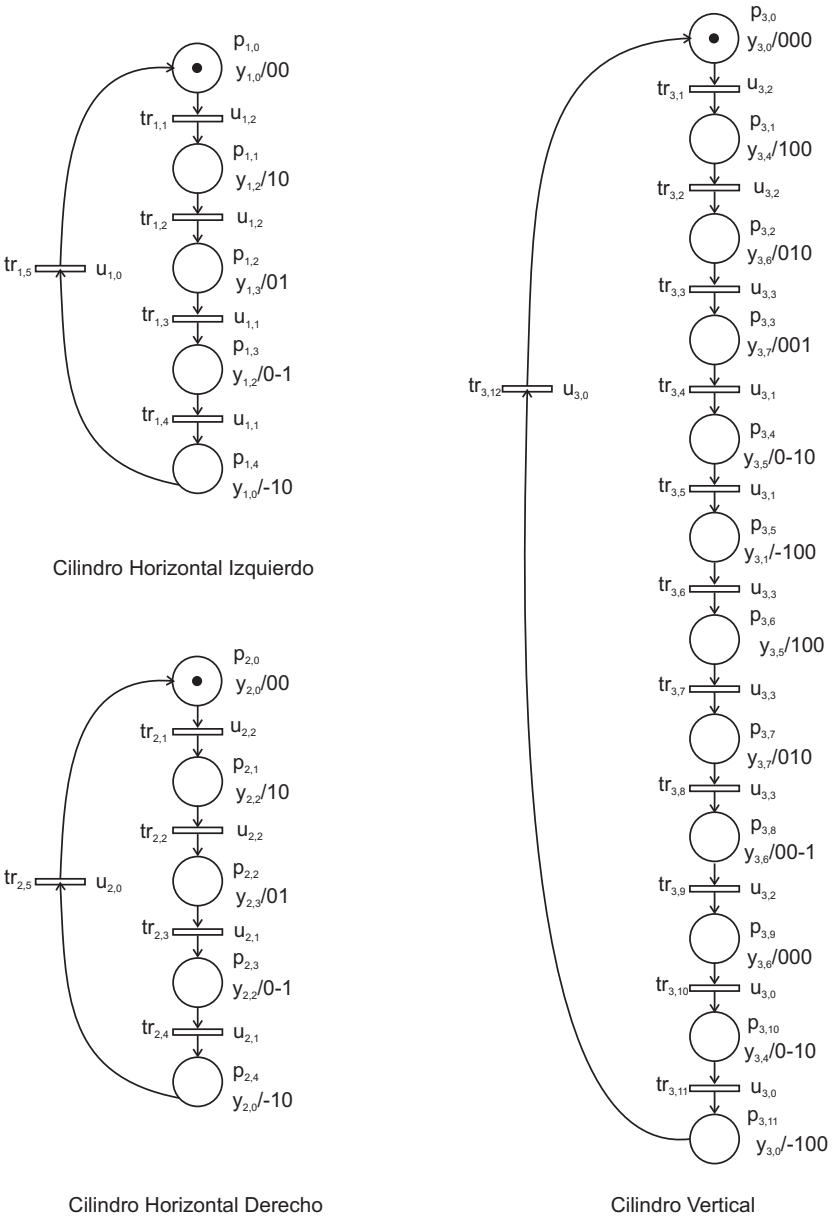


Figura 4.4.4: Modelado del brazo robótico

El lenguaje del sistema global se puede construir por la sincronización temporal de los lenguajes generados por las st-IPNs de los subsistemas, $\omega^i = \omega_1^i \oplus \cdots \oplus \omega_c^i$ (Ver Definición 30).

Y la st-IPN del Sistema Global del Sistema Robótico, aplicando Producto Síncrono de st-IPNs, (ver Definición 31), solo para el modo de operación o_1, tendrá un estructura lineal cíclica con 22 lugares y 22 transiciones que es similar a la unión de las estructuras de los subsistemas; pero cuando se modelan varios modos de operación en un ciclo, la estructura de los subsistemas es más sencilla que la estructura global.

El lenguaje de la st-IPN generada por el producto síncrono de las st-IPNs de los subsistemas es el mismo lenguaje legal del sistema.

4.4.2. Observabilidad del Lenguaje del Brazo Robótico

En esta sección se prueban las ventajas de la propuesta de observabilidad temporal que se expone en la Definición 28. Para ello se han introducido eventos no-observables al sistema del brazo robótico.

El lenguaje en comportamiento normal para el subsistema 3 es: $\mathcal{L}_3 = (u_{3,0}, y_{3,0})\,.t^0_{3,o_1}\ (u_{3,2}, y_{3,4})\,.t^1_{3,o_1}\ (u_{3,2}, y_{3,6})\,.t^2_{3,o_1}\ (u_{3,3}, y_{3,7})\,.t^3_{3,o_1}$ $(u_{3,1}, y_{3,5})\,.t^4_{3,o_1}\quad (u_{3,1}, y_{3,1})\,.t^5_{3,o_1}\quad (u_{3,3}, y_{3,5})\,.t^6_{3,o_1}\quad (u_{3,3}, y_{3,7})\,.t^7_{3,o_1}$ $(u_{3,2}, y_{3,14})\,.t^8_{3,o_1}\ (u_{3,0}, y_{3,4}).t^9_{3,o_1}(u_{3,0}, y_{3,0})\,.t^{10}_{3,o_1}$ en τ_0, τ_5, τ_6, τ_7, τ_8, τ_9, τ_{14}, τ_{15}, τ_{16}, τ_{17} , τ_{18} instantes; es decir $\mathcal{L}_3 = \omega_3^0\omega_3^5\omega_3^6\omega_3^7\ \omega_3^8\omega_3^9\omega_3^{14}\omega_3^{15}$ $\omega_3^{16}\omega_3^{17}\omega_3^{18}$.

Es de aclarar que superíndice de t, indica la secuencia de eventos en el subsistema 3 y el superíndice de ω, es la secuencia en los eventos de todo el sistema.

Considerando una entrada externa a la planta, como un fallo en la unidad del succionador, es decir $EExP = \{f_{suc}\}$, el símbolo de

entrada al subsistema 3 será: $u_{3,s} = [f_{suc} cc_{3,1} cc_{3,2}]$ y el de salida $y_{3,j} = [sr_{3,1} sr_{3,2} sr_{3,3} sr_{3,4}]$.

Sea s una secuencia de eventos normales en el subsistema 3, tal que: $s = (u_{3,0}, y_{3,0}) .t^0_{3,o_1} (u_{3,2}, y_{3,4}) .t^1_{3,o_1} (u_{3,2}, y_{3,6}) .t^2_{3,o_1} (u_{3,3}, y_{3,7}) .t^3_{3,o_1} (u_{3,1}, y_{3,5}) .t^4_{3,o_1}$ en $\tau_0, \tau_5, \tau_6, \tau_7, \tau_8$ y sea t una secuencia de observados eventos tal que $t \in \mathcal{L}_3/s$.

Ahora, asumiendo que en el instante τ_9 se genera el fallo en el succionador, es decir $f_{suc} = 1$, el símbolo de entrada no cambia puesto que el fallo es no-observable; pero las lecturas sensoriales podrían cambiar después de uno o más eventos, es decir en la secuencia t. Entonces, si la $Pt_{U_o}(\mathcal{L}_3) \neq Pt_{U_o}(st)$ y/o la proyección $Pt_Y(\mathcal{L}_3) \neq P_Y(st)$; entonces el evento de fallo es n-observable, donde n es el número de eventos que suceden hasta que el fallo es detectado. Por ejemplo si la secuencia st es:

$st = (u_{3,0}, y_{3,0}) .t^0_{3,o_1} (u_{3,2}, y_{3,4}) .t^1_{3,o_1} (u_{3,2}, y_{3,6}) .t^2_{3,o_1} (u_{3,3}, y_{3,7}) .t^3_{3,o_1} (u_{3,1}, y_{3,5}) .t^4_{3,o_1} (u_{3,1}, y_{3,1}) .t^5_{3,o_1} (u_{3,0}, y_{3,0}) .t^6_{3,o_1} (u_{3,3}, y_{3,4}) .t^7_{3,o_1} (u_{3,3}, y_{3,6}) .t^8_{3,o_1} (u_{3,2}, y_{3,14}) .t^9_{3,o_1} (u_{3,0}, y_{3,4}) .t^{10}_{3,o_1}$ en τ_0, τ_5, τ_6, τ_7, τ_8, τ_9, $\tau_{14}, \tau_{15}, \tau_{16}, \tau_{17}, \tau_{18}$ instantes.

$Pt_Y(\mathcal{L}_3) = Pt_Y(\omega^0_3) Pt_Y(\omega^5_3) \cdots Pt_Y(\omega^{18}_3)$, donde, $Pt_Y(\omega^0_3) = Pt_Y\left((u_{3,0}, y_{3,0}) .t^0_{3,o_1}\right) = Pc_Y(u_{3,0}, y_{3,0}) P\left(t^0_{3,o_1}\right)$, ver Definiciones 3 y 6. Entonces $Pt_Y(\omega^0_3) = y_{3,0}$.

Por lo tanto $Pt_Y(\mathcal{L}_3) = y_{3,0} y_{3,4} y_{3,6} y_{3,7} y_{3,5} y_{3,1} y_{3,5} y_{3,7} y_{3,14} y_{3,4} y_{3,0}$.

$Pt_Y(st) = y_{3,0} y_{3,4} y_{3,6} y_{3,7} y_{3,5} y_{3,1} y_{3,0} y_{3,4} y_{3,6} y_{3,14} y_{3,4}$. De donde se concluye que $Pt_Y(\mathcal{L}_3) \neq Pt_Y(st)$ y el evento de fallo se puede detectar después de 2 eventos de su ocurrencia.

Pero si la entrada no es un fallo, sino un cambio en el tiempo que tarda el cilindro horizontal en recoger el brazo, es decir, $EExP = \{e_{uo_1}\}$, el símbolo de entrada al subsistema 2 no cambia puesto que es un evento no-observable; las lecturas sensoriales tampoco

se verían afectadas, entonces si $Pt_{U_o}(\mathcal{L}_2) \neq Pt_{U_o}(st)$ el evento no-observable puede ser detectado.

El lenguaje en comportamiento normal para el subsistema 2 es $\mathcal{L}_2 = (u_{2,0}, y_{2,0}).t^0_{2,o_1}\ (u_{2,2}, y_{2,2}).t^1_{2,o_1}\ (u_{2,2}, y_{2,3}).t^2_{2,o_1}\ (u_{2,1}, y_{2,2}).t^3_{2,o_1}$ $(u_{2,1}, y_{2,0}).t^4_{2,o_1}\ (u_{2,0}, y_{2,0}).t^5_{2,o_1}$ en τ_0, τ_3, τ_4, τ_{10}, τ_{11}, τ_{18} instantes, es decir $\mathcal{L}_2 = \omega_2^0\omega_2^3\omega_2^4\omega_2^{10}\omega_2^{11}\omega_2^{18}$.

Por ejemplo si la secuencia st es:

$st = (u_{2,0}, y_{2,0}).t^0_{2,o_1}\ (u_{2,2}, y_{2,2}).t^1_{2,o_1}\ (u_{2,2}, y_{2,3}).t^2_{2,o_1}\ (u_{2,1}, y_{2,2}).t^{*3}_{2,o_1}$ $(u_{2,1}, y_{2,0}).t^4_{2,o_1}\ (u_{2,0}, y_{2,0}).t^5_{2,o_1}$.

$Pt_{U_o}(\mathcal{L}_2) = Pt_{U_o}(\omega_2^0)\, Pt_{U_o}(\omega_2^3) \cdots Pt_{U_o}(\omega_2^{18})$, donde $Pt_{U_o}(\omega_3^0) = Pt_{U_o}\left((u_{2,0}, y_{2,0}).t^0_{2,o_1}\right) = Pc_{U_o}(u_{2,0}, y_{2,0})\, P\left(t^0_{2,o_1}\right) = u_{2,0}.t^0_{2,o_1}$, entonces:

$Pt_{U_o}(\mathcal{L}_2) = u_{2,0}.t^0_{2,o_1}u_{2,2}.t^1_{2,o_1}u_{2,2}.t^2_{2,o_1}u_{2,1}.t^{*3}_{2,o_1}u_{2,1}.t^4_{2,o_1}u_{2,0}.t^5_{2,o_1}$.

Si $a \leq t^{*3}_{2,o_1} \leq b$, donde $[a, b]$ es el intervalo de confianza para el tiempo t^3_{2,o_1}, se considera que la variación del tiempo del evento obedece a una variabilidad propia del comportamiento estocástico del sistema, de lo contrario se ha detectado un comportamiento que no es normal.

Lo anterior comprueba la importancia de la proyección de eventos temporizados definida en la Definición 6, para detectar comportamientos no normales a partir de eventos no-observables, en sistemas estocásticos.

4.5. Modelado de SED con Red de Petri Coloreadas

Las CPN (ver Definición 13) son estructuras adecuadas para modelar y validar sistemas en los cuales la concurrencia, comunicación y

la sincronización juegan un papel importante. Las CPN permiten organizar modelos complejos como un conjunto de módulos que permiten incluir el concepto de tiempo en el sistema modelado, entre otras ventajas, [97].

En el contexto del presente trabajo se hace necesario adicionar ciertas características a una CPN convencional. Dentro de estas características se encuentran el modelar bajo señales de E/S, etiquetar transiciones y lugares, definir los lenguajes generados por las CPN y generar una estructura variable.

Para ello a continuación se proponen una serie de definiciones que van cambiando la estructura general hasta conseguir la estructura deseada.

El primer cambio radica en etiquetar una CPN; para ello se hace necesario discriminar las entradas observables del sistema a modelar, dependiendo del tipo de marca, es decir $U_o = \cup U_o^h$; con $h = 1 : ncc$, donde ncc es el número de clases de color y las respectivas señales de salida, $Y = \cup Y^h$.

Definición 32 (Red de Petri Coloreada Interpretada (ICPN))**.** Una ICPN es una estructura $CQ = (CN, U, Y, \lambda, \varphi)$, donde $CN = (P, TR, C, cd, Pre, Post, M_0)$ es una CPN definida como en la Definición 13; U_o, Y son los alfabetos de entrada y de salida respectivamente, $U_o = \cup U_o^h$ y $Y = \cup Y^h$, con $h = 1 : ncc$, ncc es el número de clases de color, donde $U_o^h = \left\{u_0^h, \cdots, u^h_{\left|2^{m_{o,h}}\right|-1}\right\}$ y $Y^h = \left\{y_0^h, \cdots, y^h_{|2^{n_h}|-1}\right\}$ son los alfabetos de entrada y salida para una clase de color h, respectivamente.

$\lambda : TR \times \beta_i \to U_o$ es la función de etiquetado de las transiciones ($\beta_i \in C$); $\varphi : R(CN) \times C \to Y_o$ es una función de salida que asigna símbolos de salida a cada lugar marcado.

Otra característica a adicionar es la temporización de los eventos, para modelar los cambios de estado del sistema en una línea de

tiempo cuando confluyen diferentes marcas de color. El tiempo se asume estocástico y su comportamiento bajo una función de densidad de probabilidad de distribución continua.

Definición 33 (Red de Petri Coloreada Interpretada Temporizada Estocástica (st-ICPN))**.** Una st-ICPN, es una estructura formada por $stCQ = (CQ, \Omega, \delta, OM)$, donde $CQ = (CN, U, Y, \lambda, \varphi)$ es una ICPN, con CN, U, Y definidos como en la Definición 32; pero con $\lambda : TR \times \beta_i \rightarrow U_o.\delta$ y con función de salida φ definida como: $\varphi : R(CQ) \times C \rightarrow Y/dY$, siendo $R(CQ)$ el conjunto de alcanzabilidad de la $stCQ$. La función de salida es isomórfica sobre Y/dY. dY se define como en la Definición 21.

$\Omega := (C \times U_o \times Y).\delta$ es el alfabeto del sistema y $\omega^i = \left(u_s^h, y_j^h\right).f\left(t_{r,o_{om},h}\right)$ es un símbolo del alfabeto.

$\delta := TR \times OM \times C \rightarrow f(t_{TR \times OM \times C})$ es una función de asignación de las funciones de densidad de probabilidad del tiempo de disparo de cada transición por cada modo de operación en cada clase de color.

$OM = \left\{o_0, \cdots o_{|2^{n_{ec}}|-1}\right\}$ es el conjunto de modos de operación.

Una transición $tr_r\left(\beta_i\right) \in TR$ con $\lambda(tr_r) = u_s^h.f\left(t_{r,o_{om},h}\right)$, donde h es el color, está habilitada con respecto a β_i, en el modo de operación o_{om}, si $\forall p_q \in \bullet tr_r\left(\beta_i\right)$, $m\left(p_q\right) \geq I\left(p_q, tr_r\left(\beta_i\right)\right)$ y si $\left(a \leq t_{o_{om},h}^{ev} \leq b\right)$ siendo $\int_a^b f\left(t_{r,o_{om},h}\right) \geq (1-\alpha)$, donde $1-\alpha$ es el nivel de confiabilidad; $[a, b]$ es el intervalo de confianza y $t_{o_{om},h}^{ev}$ es el tiempo asociado al evento. Por lo tanto una transición está habilitada sí cada lugar de entrada cumple con los requerimientos del color de su marca y si el tiempo transcurrido ha alcanzado un cierto nivel de confianza.

4.5.1. Lenguaje de una st-ICPN

Existe un evento cuando se presenta un cambio en las señales de entrada y/o en las señales de salida. Un evento temporizado se definió como $\omega^i = (u_s y_j)\, .t^{ev}_{o_{om}}$ en τ_i, en Definición 4; entonces cuando se añade color y corresponde a un modo de operación específico, se definirá como: $\omega^i = \left(u^h_s, y^h_j\right) .t^{ev}_{o_{om},h}$ en τ_i, siendo h el color asociado. El lenguaje generado por el sistema es una secuencia de eventos temporizados que evolucionan la st-ICPN, así: $\mathcal{L}(CQ) = \left\{\omega^0, \omega^1, \cdots, \omega^k\right\}$ a instantes $\tau_0 \leq \tau_1 \leq \cdots \leq \tau_k$.

A partir de la Definición 6, el lenguaje generado por cada color será:

$$\mathcal{L}^h(CQ) = \left\{ P_{\left(U^h_o . Y^h\right)}(s) \,|\, s \in \mathcal{L}(CQ) \right\}.$$

4.5.2. jemplo de modelado de un Sistema como una st-ICPN

Ejemplo 7. Sea un sistema con dos cilindros neumáticos de doble efecto que se muestra en la Figura 4.5.1; cada uno tiene dos posiciones Pos 1 y Pos 2; en las cuáles hay un sensor.

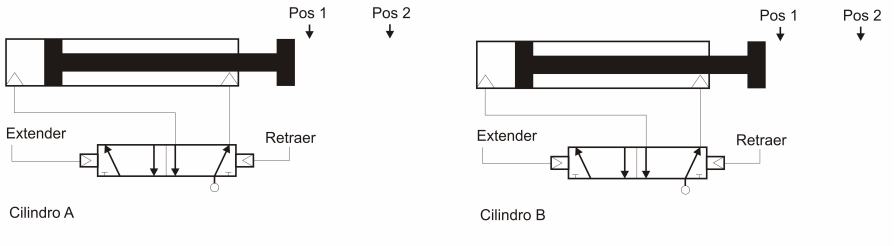


Figura 4.5.1: Cilindros neumáticos

Las señales externas al controlador $E_{e_c} = \{cil_A, cil_B\}$, son las señales que determinan el funcionamiento del cilindro A o del cilindro B. Por lo tanto, los modos de funcionamiento son 4, $OM = \{o_0, o_1, o_2, o_3\}$. La combinación o_0 no es posible porque no existiría funcionamiento.

Las órdenes del controlador son: Extender: E_A y Retraer: $\bar{E}_A$ para el cilindro A y Extender: E_B y Retraer: $\bar{E}_B$ para el cilindro B. La lectura de cada sensor es 1 si detecta algo en su posición y los actuadores inician una acción después de recibir 1 desde el controlador; el sistema parte de un estado inicial en el cual las lecturas de los sensores 1 y 2 miden 0, es decir, $\overline{s}_1, \overline{s}_2$, para el cilindro A e igual para el cilindro B, $\overline{s}_3, \overline{s}_4$. Se supone que los eventos se generan en el tiempo de manera síncrona. Por lo tanto:

- Definición del estado inicial: $\omega^0 = \{\omega_1^0.\omega_2^0\}$;

  $\omega_1^0 = \left[\bar{E}_A, \overline{s}_1\overline{s}_2\right] = [u_{1,0}y_{1,0}]$;

  $\omega_2^0 = \left[\bar{E}_B, \overline{s}_3, \overline{s}_4\right] = [u_{2,0}y_{2,0}]$;
- Función de Salida Inicial

  $\varphi\left(m\left(p_{1,0}\right)\right) = y_{1,0}/\overrightarrow{0}_2$;

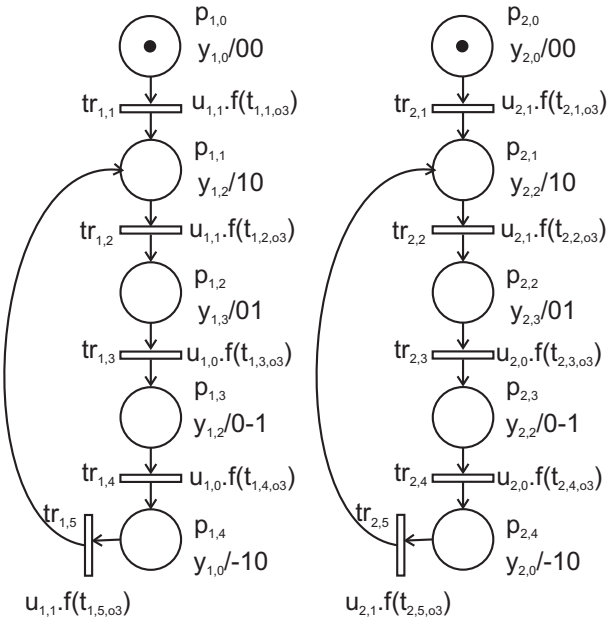
  $\varphi\left(m\left(p_{2,0}\right)\right) = y_{2,0}/\overrightarrow{0}_2$;

Las secuencias de eventos temporizados para los subsistemas involucrados son:

$\mathcal{L}_A = \left(u_{1,0}y_{1,0}\right)\left(u_{1,1}y_{1,2}\right)\left(u_{1,1}y_{1,3}\right)\left(u_{1,0}y_{1,2}\right)\left(u_{1,0}y_{1,0}\right)\left(u_{1,1}y_{1,2}\right)$ a instantes $\tau_0, \tau_1, \tau_3,\ \tau_4, \tau_7, \tau_9$; con $\delta_A = \left\{f\left(t_{1,o_3}^1\right), f\left(t_{1,o_3}^2\right), f\left(t_{1,o_3}^3\right), f\left(t_{1,o_3}^4\right), f\left(t_{1,o_3}^5\right)\right\}$.

$\mathcal{L}_B = \left(u_{2,0}y_{2,0}\right)\left(u_{2,1}y_{2,2}\right)\left(u_{2,1}y_{2,3}\right)\left(u_{2,0}y_{2,2}\right)\left(u_{2,0}y_{2,0}\right)\left(u_{2,1}y_{2,2}\right)$ a instantes $\tau_0, \tau_2, \tau_5,\ \tau_6, \tau_8, \tau_{10}$; con $\delta_B = f\left(t_{2,o_3}^1\right), f\left(t_{2,o_3}^2\right), f\left(t_{2,o_3}^3\right), f\left(t_{2,o_3}^4\right), f\left(t_{2,o_3}^5\right)$.

Las st-IPNs, para cada uno de los cilindros se muestran en la Figura 4.5.2.



(a) Cilindro A **(b)** Cilindro B

Figura 4.5.2: st-IPNs para cilindros neumáticos

Suponiendo que existen dos cilindros A y uno B e integrando las st-IPNs en una st-ICPN, la red se muestra en la Figura 4.5.3.

Por lo tanto, la st-ICPN queda definida como:

$P = \{p_0, p_1, p_2, p_3, p_4\}$; $TR = \{tr_1, tr_2, tr_3, tr_4, tr_5\}$;

$C = \{cilindros, modo\}$; $cilindros = \{A, B\}$; $modo = \{\beta_A, \beta_B, \beta_A \wedge \beta_B\}$; $cd\,(p_q) = \{A, B\}$; $cd(tr_{l,r}) = \{\beta_A, \beta_B, \beta_A \wedge \beta_B\}$;

$U = \{U_A, U_B\}$; $Y = \{Y_A, Y_B\}$; $U_A = \left\{u^A_{1,0}, u^A_{1,1}\right\}$; $U_B = \left\{u^B_{2,0}, u^B_{2,1}\right\}$; $Y_A = \left\{y^A_{1,0}, y^A_{1,1}, y^A_{1,2}, y^A_{1,3}\right\}$; $Y_B = \left\{y^B_{2,0}, y^B_{2,1}, y^B_{2,2}, y^B_{2,3}\right\}$.

$M_0 = \{\langle 2A, B\rangle, 0, 0, 0, 0\}$.

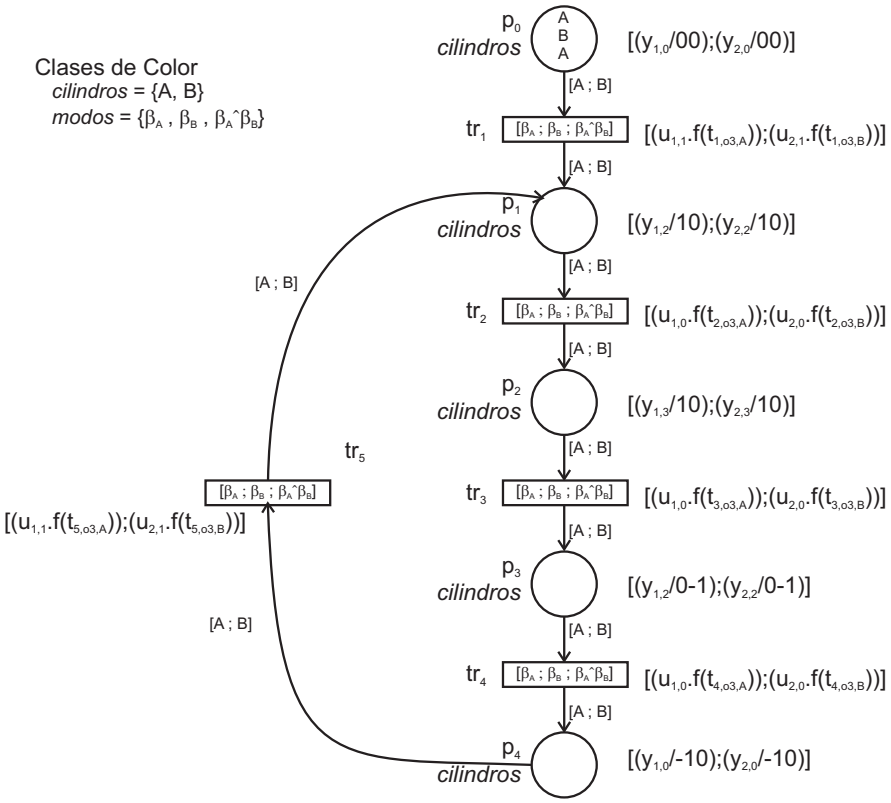


Figura 4.5.3: st-ICPN cilindros neumáticos

El conjunto de asignaciones $Pre\,[p, tr] : cd\,(tr) \rightarrow Bag\,(cd\,(p))$ y $Post\,[p, tr] : cd\,(tr) \rightarrow Bag\,(cd\,(p))$, se representan por vectores como:

$$Pre = \begin{array}{c} \begin{bmatrix} \beta_A \\ \beta_B \\ \beta_A \wedge \beta_B \end{bmatrix} \quad \begin{bmatrix} \beta_A \\ \beta_B \\ ,\beta \wedge \beta_B \end{bmatrix} \quad \begin{bmatrix} \beta_A \\ \beta_B \\ ,\beta_A \wedge \beta_B \end{bmatrix} \quad \begin{bmatrix} \beta_A \\ \beta_B \\ \beta_A \wedge \beta_B \end{bmatrix} \quad \begin{bmatrix} \beta_A \\ \beta_B \\ ,\beta_A \wedge \beta_B \end{bmatrix} \\ \begin{bmatrix} \begin{bmatrix} A \\ B \\ A+B \end{bmatrix} & 0 & 0 & 0 & 0 \\ 0 & \begin{bmatrix} A \\ B \\ A+B \end{bmatrix} & 0 & 0 & 0 \\ 0 & 0 & \begin{bmatrix} A \\ B \\ A+B \end{bmatrix} & 0 & 0 \\ 0 & 0 & 0 & \begin{bmatrix} A \\ B \\ A+B \end{bmatrix} & 0 \\ 0 & 0 & 0 & 0 & \begin{bmatrix} A \\ B \\ A+B \end{bmatrix} \end{bmatrix} \end{array}$$

$$Post = \begin{array}{c} \begin{bmatrix} \beta_A \\ \beta_B \\ \beta_A \wedge \beta_B \end{bmatrix} \quad \begin{bmatrix} \beta_A \\ \beta_B \\ ,\beta \wedge \beta_B \end{bmatrix} \quad \begin{bmatrix} \beta_A \\ \beta_B \\ ,\beta_A \wedge \beta_B \end{bmatrix} \quad \begin{bmatrix} \beta_A \\ \beta_B \\ \beta_A \wedge \beta_B \end{bmatrix} \quad \begin{bmatrix} \beta_A \\ \beta_B \\ ,\beta_A \wedge \beta_B \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ \begin{bmatrix} A \\ B \\ A+B \end{bmatrix} & 0 & 0 & 0 & \begin{bmatrix} A \\ B \\ A+B \end{bmatrix} \\ 0 & \begin{bmatrix} A \\ B \\ A+B \end{bmatrix} & 0 & 0 & 0 \\ 0 & 0 & \begin{bmatrix} A \\ B \\ A+B \end{bmatrix} & 0 & 0 \\ 0 & 0 & 0 & \begin{bmatrix} A \\ B \\ A+B \end{bmatrix} & 0 \end{bmatrix} \end{array}$$

Dado un evento $\omega_1^1 = \left(u_1^A, y_2^A\right).t_{o_2,A}^{ev}$; si $a \leq t_{o_3,A}^{ev} \leq b$, donde $[a,b] = \int_a^b f\left(t_{1,o_3,A}\right) \geq (1-\alpha)$; $tr_1\left(\beta_A\right)$ se dispara y una marca de color A es alcanzada en p_2 y $M_1 = \{\langle A, B\rangle, \langle A\rangle, 0, 0, 0\}$.

4.6. Conclusiones y aportes del capítulo

En este capítulo se presentó una propuesta de generador de lenguaje regular que permite modelar el comportamiento observado de SED estocásticos, representado bajo una estructura de red de Petri denominada st-IPN; la cual es una extensión de las redes de Petri interpretadas, con características relevantes en cuanto a que es determinística, su lenguaje es el mismo lenguaje legal del sistema y permite modelar sistemas con eventos estocásticos. A su vez, se define la estructura de lenguaje regular con base en la organización de las señales de entrada / salida y con base en los modos de funcionamiento. La evolución de la red se genera a partir de eventos temporizados.

Para evitar la explosión combinacional de estados cuando los sistemas crecen se divide el sistema en subsistemas; entonces el modelo del sistema es un conjunto de st-IPNs con las características mencionadas.

Además, se presenta un nuevo concepto de observabilidad que se ha denominado observabilidad temporal, que permite realizar test de observabilidad en sistemas estocásticos.

La propuesta conlleva la generación de definiciones particulares de símbolos, eventos y lenguajes para cumplir con el objetivo de modelado, presentados en la Ecuación 4.3.1, la Ecuación 4.1.3 y la Definición 4 entre otras.

A partir de las Proposiciones 1 y 6, se resuelven la preguntas planteadas en la introducción de este capítulo, puesto que se comprueba que una st-IPN modela el lenguaje legal de un SED y su lenguaje permite detectar algunos eventos no-observables legales y eventos de fallo. Por lo tanto, con base en el modelo de un SED temporizado estocástico representado como una st-IPN se puede realizar diagnóstico de fallos.

5 Identificación

Para lograr el objetivo de la presente investigación se hace necesario definir un método de identificación que permita hallar el comportamiento del sistema libre de fallo, a partir de las señales de E/S obtenidas on-line. De manera general, el problema de identificación de un SED consiste en determinar, a partir de un conjunto ordenado de datos de entrada - salida, (E/S), un modelo tal que sus E/S se aproximen al conjunto original. A partir de este contexto general y teniendo en cuenta que un SED puede ser modelado como un lenguaje regular, de forma específica, el problema de identificación en este trabajo consiste en encontrar una estructura que genere el mismo lenguaje observado.

Considerando las ventajas y desventajas de los métodos de identificación relacionados en el estado del arte, en esta sección se presenta un método de identificación para SED estocásticos a partir de la definición y organización de las señales de E/S del sistema obtenidas on-line.

5.1. Problema de Identificación

El problema central de identificación es el de hallar un modelo apropiado para un SED, como el definido en la Sección 4.1, con una estructura como la presentada en la Figura 4.3.1; con comportamiento estocástico. Dicho modelo debe identificarse a partir de la

observación de señales y bajo la teoría de lenguajes regulares definir una estructura apropiada que represente el lenguaje de E/S que modele el sistema y pueda ser utilizado en procesos de diagnóstico de fallos. El modelo resultante debe ser un generador determinista de lenguaje; es decir, una estructura como la st-IPN definida en la Definición 22.

5.2. Proceso de Identificación

El proceso de identificación de un SED se inicia con la división del sistema en subsistemas, con el objeto de tener información concisa del sistema y que permita analizarla de manera más eficiente.

Cuando se ha dividido el sistema, el siguiente paso es la definición de las señales de entrada tanto internas como externas y de las señales de salida. En esta etapa se requiere de la organización de las señales con el objeto de definir los vectores binarios para los modos de funcionamiento del sistema, los símbolos de entrada y los símbolos de salida en cada subsistema; con base en la Ecuación 4.3.1, la Ecuación 4.3.2 y la Ecuación 4.1.1.

Una vez se han organizado las señales, empieza el proceso de trabajo on-line sobre el sistema, cuyo objetivo es leer las señales en cada instante de tiempo. Las señales leídas llegan a un generador de eventos que transforma las señales en eventos, de acuerdo a características que se explican más adelante.

Los eventos alimentan el algoritmo de identificación, el cual construye paso a paso las st-IPNs correspondientes a los eventos que llegan. El algoritmo termina cuando el sistema se ha identificado totalmente y se pueda explicar estadísticamente su comportamiento.

Como resultado se obtienen las estructuras de las st-IPNs y los

lenguajes observados en cada subsistema. Este procedimiento se puede observar en la Figura 5.2.1

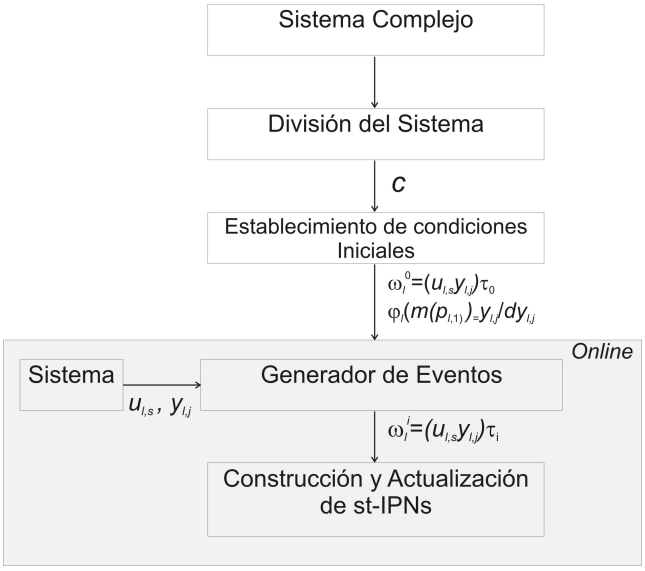


Figura 5.2.1: Proceso de identificación

5.2.1. Definición de las Señales

Un análisis previo del sistema permite discriminar los diferentes tipos de señales que se interrelacionan en un sistema; con base en este análisis, se clasifican en señales de entrada externas, que afectan el funcionamiento del sistema; señales de entrada internas a la planta y las señales de salida que son las respuestas del sistema. Además se clasifican en cada uno de los subsistemas. La señales requeridas para el proceso de identificación, pueden observarse en la Figura 4.3.1.

Como se definió en el capítulo de modelado, un símbolo de entrada externo está conformado por un vector de valores binarios de las

señales externas que entran a la planta y cada vector constituye un modo de funcionamiento: $o_{om} = [ec_1 \cdots ec_{n_{ec}}]$ donde o_{om} es una representación binaria de om. Un símbolo de entrada al subsistema l, es un vector de valores binarios de las señales medibles definido como $u_{l,s} = \left[eop_1 \cdots ep_{n_{eop}} gcc_1 \cdots gcc_{n_{gc}} cc_{l,1} \cdots cc_{l,n_{cc}}\right]$, (solo se toma la parte medible de la Ecuación 4.3.1), donde $u_{l,s}$ es una representación binaria de s. Un símbolo de salida del subsistema l, es un vector de valores binarios definido como $y_{l,j} = [sr_{l,1} \cdots sr_{l,n_l}]$, (ver Ecuación 4.3.2), donde $y_{l,j}$ es una representación binaria de j.

5.2.2. Generador de eventos

El proceso de identificación requiere monitorizar las diferentes señales del sistema, como eventos que afectan el comportamiento del sistema. Un evento externo es un modo de funcionamiento o_{om} en un instante τ_i. El generador de eventos concatena las señales intercambiadas entre la planta y el controlador (señales de E/S) y genera un evento interno de la forma $\omega_l^i = (u_{l,s} y_{l,j})$ en un instante τ_i, (ver Figura 5.2.2).

A partir de un modo de operación o_{om} ($o_{om} \in OM$), el procedimiento de detección de eventos, genera eventos cuando sucede algún cambio en las señales de E/S. Esto es, en cada instante lee el símbolo de entrada, espera la estabilización de las lecturas de los sensores, lee los símbolos de salida, chequea sí hay algún cambio en al menos una de ellas y genera un evento en cada subsistema involucrado. En el Algoritmo 5.1 se programa la generación de evento en un subsistema l.

El evento llega al identificador, el cual continuamente construye st-IPNs que son capaces de generar el mismo evento, en la misma secuencia y al mismo tiempo.

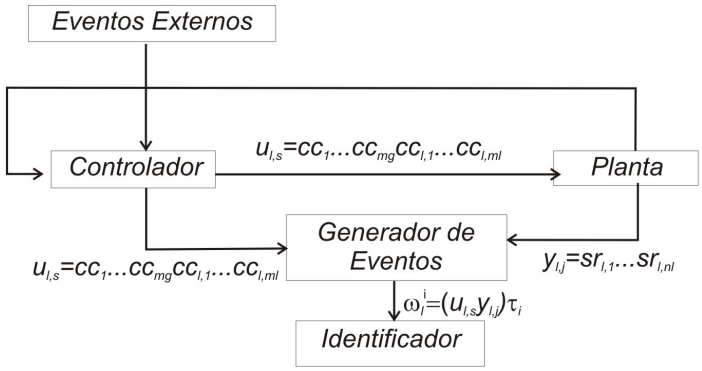


Figura 5.2.2: Generador de eventos

5.2.3. Construcción de las st-IPN

La construcción del modelo se realiza on-line a partir de los eventos generados en cada instante de tiempo. La metodología propuesta permite que se generen simultáneamente una st-IPN por cada subsistema.

El evento inicio para cada subsistema es $\omega_l^0 = (u_{l,s}y_{l,j})\,\tau_0$, donde $u_{l,s}y_{l,j}$ es el símbolo de E/S en el instante en que se inicia la identificación (τ_0). Entonces, cada subsistema inicia con una st-IPN de un lugar, $q_l = 1$, cuya función de salida será: $\varphi_l\,(m\,(p_{l,0})) = y_{l,j}/\overrightarrow{0}$ y la función de entrada será: $\lambda_l\,(tr_{l,0}) = u_{l,s}$ para $l = 1 \cdots c$, donde c es el índice del subsistema, esto significa que el subsistema l inicia sin transiciones $r_l = 0$. (q_l número de lugares en subsistema l, r_l número de transiciones en subsistema l).

La construcción de las st-IPNs, sigue el diagrama de flujo presentado en la Figura 5.2.3. Identifica un cambio de estado de un subsistema cuando hay una diferencia entre $y_{l,j}$ y el símbolo de salida actual en el subsistema l. El proceso permite crear nuevas transiciones y nuevos lugares cuando sea necesario y actualiza las marcas de la st-IPN. δ_l^* es una matriz que va almacenando el

Algoritmo 5.1 Generación de Eventos

Entradas: $\omega_l^{i-1} = (u_{l,s}y_{l,j})\, t_{l,o_{om}}^{ev}$, τ_{i-1}; $u_{l,s}$; $y_{l,j}$ para τ_i; c; i, o_{om}.
Salidas: $\omega_l^i = (u_{l,s}y_{l,j})\, t_{l,o_{om}}^{ev}$; *evento*; l.

1. Leer (l); $u_{l,s}$.
2. Esperar tiempo de establecimiento.
3. Leer $y_{l,j}$.
4. $i^{'} = i + 1$
5. $\omega_l^{i^{'}} = (u_{l,s}y_{l,j})$
6. condición: $\omega_l^{i^{'}} \neq \omega_l^i$, hay evento,
   - entonces $evento = 1$; $i = i + 1$; $t_{l,o_{om}}^{ev} = |\tau_i| - |\tau_{i-1}|$; $\omega_l^i = (u_{l,s}y_{l,j})\, t_{l,o_{om}}^{ev}$.
   - de lo contrario $evento = 0$.

   fin de condicional

tiempo de los eventos en cada subsistema, es una matriz de tres dimensiones: Las filas son las transiciones del subsistema, las columnas el tiempo identificado y las páginas los modos de operación o_{om}. La función de entrada se denota como λ_l^*, ya que dentro del algoritmo aún no tiene el símbolo de entrada total como el definido en la Definición 22; puesto que le hace falta identificar la función de densidad de probabilidad.

La actualización de las matrices *Pre* y *Post*, se realiza con base en el Algoritmo 4.1.

Este proceso se repite hasta que los ciclos de cada subsistemas se hayan identificado. Para ello el vector de tiempo debe contener al

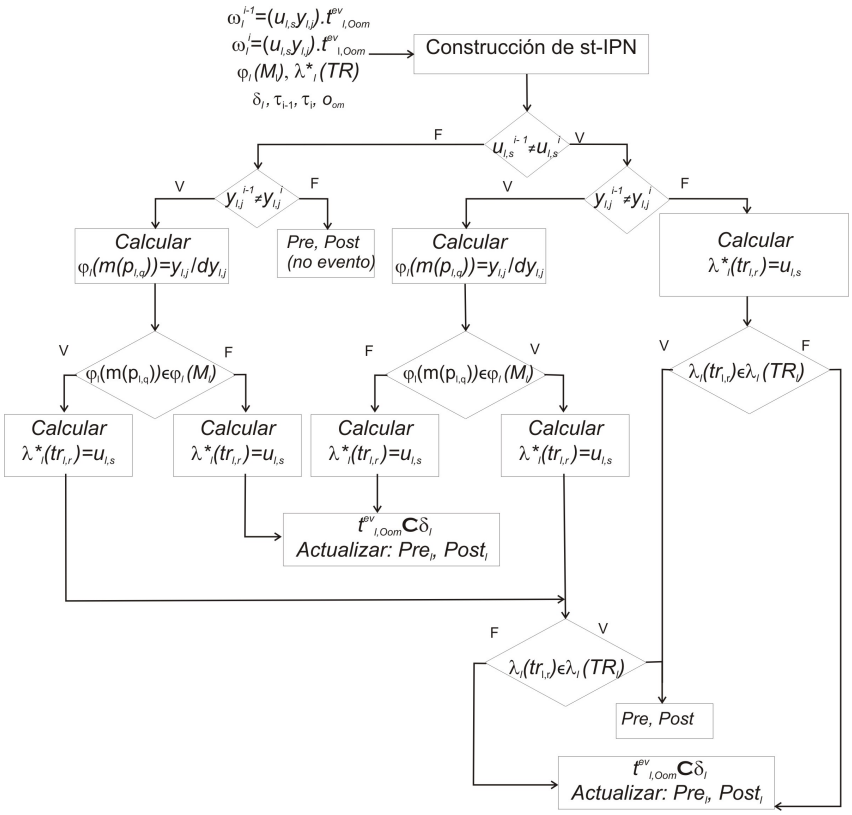


Figure 5.2.3: Proceso de Construcción de una st-IPN.

menos cinco observaciones en cada tr, en cada o_{om} activo y en cada subsistema. Con base en esta información, se determina si el tamaño de los datos constituye una muestra representativa para inferir respecto a su comportamiento estadístico, ya que cada transición posee un vector de tiempos con una marcada aleatoriedad puesto que se identifican sistemas estocásticos. Si el tamaño de muestra no es representativa se continua con la lectura de las señales.

Por lo tanto el siguiente paso es identificar el comportamiento estadístico de estos tiempos. δ_l^* contiene la información para este paso.

5.2.4. Inclusión de la Información Temporizada.

En esta sección se explica la utilización de la información temporizada para solucionar el problema de identificación estocástica.

En general, la información temporizada está relacionada con el tiempo que transcurre entre dos eventos consecutivos, este tiempo depende del modo de operación en el cual el sistema se encuentre, pues la respuesta de los dispositivos se ve afectada por la interacción con otros eventos externos.

Cada nuevo evento tiene un modo de funcionamiento, o_{om}, un número de transición, r_l y el tiempo transcurrido en un subsistema l. La información acumulada en δ_l^* contiene la información que permite inferir sobre el comportamiento de cada transición. Por lo que el sistema debe ser observado por varios periodos de tiempo con el objeto de tener suficientes datos para este análisis.

El procedimiento propuesto para el análisis de datos (Algoritmo 5.2) está basado en la teoría presentada en [26], [27] y [28].

El Algoritmo 5.2 en primer lugar determina si hay suficientes datos para que éstos sean representativos de la población; luego el

Algoritmo 5.3 encuentra el tipo de distribución continua que se ajusta a los datos.

Para encontrar un tamaño de datos representativo, se despeja d, que se considerará como un nivel de precisión del estudio, a partir de la Ecuación 2.4.1 como:

$$d = \left(st_{\alpha/2}\right) \times S/\sqrt{n} \tag{5.2.1}$$

El Algoritmo 5.2 se ejecuta cada vez que se obtiene un nuevo valor en el vector de tiempo (se considera importante empezar a partir de cinco datos); así el tamaño de datos se va incrementando. Se calcula d y se compara con un valor definido como límite superior (d_{op}) (máxima diferencia relativa a aceptar entre la media de la población y la media muestral). Para $n \leq 30$, el estadístico st, se calcula bajo la distribución t de Student, $st_{\alpha/2} = t_{n-1}$ y si $n > 30$, como distribución normal $st_{\alpha/2} = z_{\alpha/2}$. Una vez d_{op} es alcanzado, el Algoritmo 5.2 ya no se aplica a esta transición y al modo de operación, debido a que su distribución puede ser identificada con los datos disponibles.

Para encontrar el tipo de distribución continua, se define un conjunto, CD, el cual es un conjunto de distribuciones continuas específicas: $CD = \left\{F_0^j\left(x_i\right)\right\}$. Entre las distribuciones continuas comúnmente usadas se incluye: la distribución normal $N(\mu, \sigma^2)$, la distribución exponencial $Exp(\lambda)$; la distribución uniforme $U(a, b)$ $\cdots$. A partir del Algoritmo 5.3, se encuentra el tipo de distribución y sus parámetros.

Este algoritmo usa el test de bondad de ajuste (ver [98]), para definir el tipo de distribución con base en el test de Kolmogorov–Smirnov; luego selecciona una función de CD, halla sus parámetros usando el método de máxima verosimilitud y calcula el

Algoritmo 5.2 Identificación estocástica

Entradas:δ_l^* con $k \geq 5$, α, d_{op}; o_{om}; r_l.
Salidas: n_{l,r_l}; $\bar{X}$, S; d
leer $datos = \delta_l^*(:, r_l, o_{om})$;

1. $n_{l,r_l} = |datos|$;
2. calcular $\bar{X} = media\,(datos)$, $S = std\,(datos)$;
3. condición: $n_{l,r_l} \leq 30$ entonces calcular $st = t_{n-1}$
   - de lo contrario: calcular $st = z_{\alpha/2}$

   fin de condicional
4. $d = \left(st_{\alpha/2}\right) \times S/\sqrt{n_{l,r_l}}$;

estadístico $p - value$ como:

$$p - value = sup_{1 \leq i \leq n} \left| \hat{F}_n\,(x_i) - F_0\,(x_i) \right| \tag{5.2.2}$$

donde, x_i es el *i-esimo* valor observado en la muestra (la cual ha sido previamente ordenada de menor a mayor), $\hat{F}_n\,(x_i)$ es un estimador de la probabilidad de observar valores menores o iguales a x_i y $F_0\,(x_i)$ es la probabilidad de observar valores menores o iguales a x_i de la función elegida, [99]; esto es: $F_0\,(x_i) \in CD$.

Como resultado de la aplicación del Algoritmo 5.2 y del Algoritmo 5.3, se obtienen las funciones de densidad de probabilidad de cada transición, de cada subsistema y en cada modo de funcionamiento de las st-IPN.

Algoritmo 5.3 Ajuste de datos

Entradas: *datos* ,n_{l,r_l}, α_{ct} para test de contraste, CD.
Salidas: $F_0(X)$; $p - value$

1. Buscar $p - value_{\alpha_{ct}}$ en tabla de Kolmogorov-Smirnov.
2. para $j = 1 \cdots |CD|$
   a) Diseño de test de bondad de ajuste
   $H_0 : F_0(X) \backsim F_0^j(x_i)$;
   $H_1 : F_0(X) \nsim F_0^j(x_i)$.
   b) Estimación de parámetros de $F_0^j(X)$.
   c) Aplicar test de Kolmogorov-Smirnov
   - Calcular $p - value = sup_{1 \leq i \leq n} \left| \hat{F}_n(x_i) - F_0^j(x_i) \right|$.

   d) condición: $p - value \leq p - value_{\alpha_{ct}}$, entonces $F_0 = F_0^j(X)$; salir;
   de lo contrario $j = j + 1$;
   e) fin del condicional
3. fin del ciclo

5.3. Algoritmo de Identificación

El algoritmo de identificación, (Algoritmo 5.4), enlaza de manera adecuada los algoritmos presentados anteriormente. Con base en las señales de E/S y a partir de un proceso iterativo on-line, el objetivo es obtener los lenguajes en comportamiento normal representados como st-IPN así como las funciones de densidad de probabilidad de cada transición en cada subsistema y en cada modo de operación que esté activo en el sistema.

Como resultado de la aplicación del algoritmo de identificación se obtiene un lenguaje del sistema, representado como una st-IPN, donde los lugares representan los diferentes estados medidos a partir de las lecturas de los sensores y su comportamiento histórico y las transiciones permiten evolucionar la red de un estado a otro a partir de órdenes de control y de funciones de densidad de probabilidad en cada uno de los modos de operación en que se habiliten.

5.4. Error de Modelado

El algoritmo de identificación propuesto es una herramienta para la identificación estocástica de SED, la cual estima las distribuciones de probabilidad de los tiempos de las transiciones. Debido a que no es posible conocer el comportamiento de los datos de la población, existe un error de modelado. Como la media muestral es el estimador de la media de poblacional, entonces el error de modelado está relacionado con la diferencia entre el valor estimado y el verdadero valor de la media, por lo tanto el error estándar de la media es la forma de medir el error de modelado. El error estándar de la media es la desviación estándar de las medias muestrales sobre todas las muestras posibles (de un tamaño dado) extraídas de la población.

Algoritmo 5.4 Identificación de st-IPN

Entradas: $\omega_l^i = (u_{l,s} y_{l,j})\, \tau_i$; $\omega_l^{i-1} = (u_{l,s} y_{l,j})\, \tau_{i-1}$;$\varphi_l\,(M_l)$; $\lambda_l\,(TR_l)$; δ_l; o_{om}; q_l; r_l; k_l.
Salidas: Pre_l; $Post_l$; $\varphi_l\,(M_l)$; $\lambda_l\,(TR_l)$; δ_l.
Condiciones Iniciales: $\varphi_l\,(m\,(p_{l,0})) = y_{l,j}/\overrightarrow{0}$; $\lambda_l\,(tr_{l,0}) = u_{l,s}$; $Pre_l = [1]$; $Post_l = [0]$; $\delta_l = ceros\,(1 \times 1 \times max\,(om))$; $q_l = 1$; $r_l = 0$; $k_l = 0$.

1. Leer $\omega_l^i = (u_{l,s} y_{l,j})\, \tau_i$; l
2. Ejecutar Algoritmo 5.1.
   $\omega_l^i = (u_{l,s} y_{l,j})\, .t_{l,o_{om}}^{ev}$; $evento$; l.
3. Condición: $evento = 1$, entonces ejecutar Algoritmo 4.1
   Pre_l; $Post_l$; $\varphi_l\,(M_k)$; $\lambda_l^*\,(TR_l)$; δ_l.
   fin de condicional.
4. Repetir pasos 1-3 hasta que $|t_{r_l,o_{om}}| = 5$;
5. Ejecutar Algoritmo 5.2
   n_{l,r_l}; $\bar{X}$, S; d,
   a) Condición: $d \leq d_{op}$ entonces ejecutar Algoritmo 5.3
   $F_0(X)$.
   b) de lo contrario continúe paso 1.

Para medirlo, se utiliza la raíz cuadrada del error medio cuadrado (RMSE), donde $RMSE = \frac{S}{\sqrt{n}}$.

5.5. Identificabilidad

Antes de comprobar que la estructura de una st-IPN resuelve el problema de identificación de SED estocásticos, a continuación se definen las condiciones para que una PN se considere un $\alpha - Generator$, o sea un generador de eventos de E/S y las condiciones para que un sistema sea determinísticamente identificable; teniendo en cuenta que el no-determinismo en una PN ocurriría si existe al menos un lugar con al menos dos transiciones con la misma etiqueta que conducen a dos lugares diferentes. (Ver Definición 10).

Definición 34 ($\alpha - Generator$)**.** Sea $U.Y$ un alfabeto y sea $\mathcal{L}(Q)$ el lenguaje generado por el alfabeto; una PN es un $\alpha - Generator$ si dado un símbolo de entrada $u_s \in P : (U.Y)^* \rightarrow U^*$, la PN genera un símbolo de salida y_j tal que la cadena $u_s y_j \in \mathcal{L}(Q_l) \forall u_s \in U$ con una probabilidad mayor o igual a $1 - \alpha$.

5.5.1. Supuestos de un sistema para que sea un sistema determinísticamente identificable (DI).

La Definición 34 permite establecer una serie de supuestos para que un sistema sea determinísticamente identificable.

- Sea U y Y un conjunto de señales de E/S observadas de un sistema; si dado un símbolo de entrada $u_s \in U$; asociado con un tiempo t, la $P : (Y) \rightarrow y_j$ es siempre la misma.

- Por otra parte, dados símbolos de entrada idénticos u_s, pero con tiempo diferente y dos proyecciones de salida diferentes, el sistema puede considerarse determinísticamente identificable; ya que, la duración del tiempo define la marca a alcanzar.

Proposición 8. *Sea* $\mathcal{L}(Q) = \left\{\omega^0\omega^1 \cdots \omega^k\right\}$ *el lenguaje observado de un sistema, a instantes* $\tau_0 \leq \tau_1 \leq \cdots \leq \tau_k$*; si la st-IPN es una PN obtenida con el algoritmo de identificación propuesto, entonces la st-IPN es un* $\alpha - Generator\ de \mathcal{L}(Q)$.

Demostración. Una st-IPN es un $\alpha - Generator$ de $\mathcal{L}(Q)$ si $\forall s \in \mathcal{L}(Q),\ s \in \mathcal{L}(P)$ y si $\forall s \in \mathcal{L}(P),\ s \in \mathcal{L}(Q)$.

Sea P una st-IPN obtenida por el algoritmo de identificación propuesto para $\mathcal{L}(Q)$.

Suponiendo que ss_1 es una cadena tal que $s \in \mathcal{L}(Q) \wedge \mathcal{L}(P)$ y $s_1 \in \mathcal{L}(Q) \wedge s_1 \notin \mathcal{L}(P)$; donde $s = \omega^0, \cdots \omega^k$. Entonces el post-lenguaje de s está dado por $\mathcal{L}(Q)/s = \{s_1 \in (U.Y)^* \,|ss_1 \in \mathcal{L}(Q)\}$. Dado el evento s_1 con $s_1 = \omega^{k+1} = (u_s y_j)^{k+1}.t_{o_{om}}^{k+1}$ en el instante τ_{k+1}, aplicando el Algoritmo 5.4 un nuevo evento genera una transición con $\lambda(tr_{k+1}) = u_s^{k+1}.t_{o_{om}}^{k+1}$ y la marca alcanzada por tr_{k+1}: $m(p_k) \overset{tr_{k+1}}{\rightarrow} m(p_{k+1})$ será $\varphi(m(p_{k+1})) = y_j^{k+1}/y_j^{k+1} - y_j^k$, y como $u_s^{k+1}.y_j^{k+1}$ son vectores de señales observadas al instante τ_{k+1}, basándose en el Algoritmo 5.1, entonces $\omega^{k+1} = (u_s y_j)^{k+1}.t_{o_{om}}^{k+1} \in \mathcal{L}(P)$ por lo que el supuesto es rechazado.

Ahora, asumiendo que ss_2 es una cadena tal que $s \in \mathcal{L}(Q) \wedge \mathcal{L}(P)$ y $s_2 \notin \mathcal{L}(Q) \wedge s_2 \in \mathcal{L}(P)$; donde $s = \omega^0, \cdots \omega^k$. Entonces el post-lenguaje de s es $\mathcal{L}(P)/s = \{s_2 \in (U.Y)^* \,|ss_2 \in \mathcal{L}(P)\}$. Dado el evento s_2 con $s_2 = \omega^{k+1} = (u_s y_j)^{k+1}.t_{o_{om}}^{k+1}$ en el instante τ_{k+1}, la st-IPN evoluciona a un estado a partir de una transición con $\lambda(tr_{k+1}) = u_s^{k+1}.t_{o_{om}}^{k+1}$ y la marca alcanzada por tr_{k+1}: $m(p_k) \overset{tr_{k+1}}{\rightarrow} m(p_{k+1})$ será $\varphi(m(p_{k+1})) = y_j^{k+1}/y_j^{k+1} - y_j^k$, donde

tr_{k+1} no ha sido observada y alcanza estados que no han sido observados, pero con base en el Algoritmo 5.4 la construcción de una st-IPN se realiza con base a estados alcanzados por el lenguaje observado, entonces $\omega^{k+1} = (u_s y_j)^{k+1} . t_{o_{om}}^{k+1} \notin \mathcal{L}(P)$ y la suposición es rechazada y por lo tanto *la proposición 8* es probada. □

Además, $\mathcal{L}(Q)/s = \{s_1 \in (U.Y)^* \,|ss_1 \in \mathcal{L}(Q)\}$ alcanza la marca $\varphi(m(p_{k+1}))$ si $Prob\left(\varphi(m(p_{k+1})), \omega^{k+1}/\varphi(m(p_{k+1}))\right) \geqq (1-\alpha)$ $|\omega^{k+1} \in \mathcal{L}(Q)$ y si $\varphi(m(p_{k+1})) = \varphi(m(p_i))$ para algún $i = 1 \cdots k$, entonces ss_1 es un ciclo (Proposición 3).

Definición 35 (Lenguaje Determinísticamente Identificable (DI))**.** Un $\mathcal{L}(Q)$, donde $\mathcal{L}(Q) = \{s \in (U.Y)^* : Prob(s|\varphi(m(p_0))) \geqq (1-\alpha)\}$ es un lenguaje DI si $\forall s \in \mathcal{L}(Q), \wedge \forall s_1, s_2 \in \mathcal{L}(Q)/s$; donde $s = \omega^0, \cdots \omega^k$, tal que $P_{u_s} : (ss_1) = P_{u_s} : (ss_2)$ entonces $Prob\left(P_{y_j} : (ss_1) = P_{y_j} : (ss_2)\right) \geq 1 - \alpha$, si ambas cadenas inician a partir del mismo estado $\varphi(m(p_{k+1}))$.

En palabras, un $\mathcal{L}(Q)$ es un lenguaje DI, si a partir de un estado, dados dos símbolos de entrada iguales, los símbolos de salida son siempre los mismos.

Teorema 2. *Un lenguaje $\mathcal{L}(Q)$ es un lenguaje DI sii $\exists$ un $\alpha -$ Generator para $\mathcal{L}(Q)$.*

Demostración. Condición necesaria: si $\mathcal{L}(Q)$ es un lenguaje DI, entonces $\exists$ un $\alpha - Generator$.

Un $\alpha - Generator$ determinístico para $\mathcal{L}(Q)$ es una st-IPN. $\mathcal{L}(Q)$ es un lenguaje DI, si el $\alpha - Generator$ es una st-IPN (ver proposición 8).

Condición suficiente: si $\exists$ un $\alpha - Generator$ entonces $\mathcal{L}(Q)$ es un lenguaje DI.

Suponiendo que el $\alpha - Generator$ para $\mathcal{L}(Q)$ es una st-IPN. Sea $\mathcal{L}(P)$ el lenguaje generado por $\alpha - Generator$. Sea $s = \omega^0, \cdots \omega^k$ una cadena tal que $\mathcal{L}(P) = \{s \in (U.Y)^* : Prob\,(s|\varphi\,(m\,(p_0))) \geqq (1-\alpha)\}$. Entonces, dados los eventos s_1 y s_2 tal que $s_1, s_2 \in \mathcal{L}(P)/s$; con $s_1, s_2 = \omega^{k+1}$ y $Prob\left(\varphi\,(m\,(p_{k+1}))\,, \omega^{k+1}/\varphi\,(m\,(p_k))\right) \geqq (1-\alpha)$ si $P_{u_s} : (ss_1) = P_{u_s} : (ss_2)$, en el evento ω^{k+1} entonces $P_{y_j} : (ss_1) = P_{y_j} : (ss_2)$, basado en la Definición 35, $\mathcal{L}(P)$ es un lenguaje DI y de acuerdo a la Proposición 4, $\mathcal{L}(P) = \mathcal{L}(Q)$, por lo tanto $\mathcal{L}(Q)$ es un lenguaje DI. □

Si una st-IPN es un $\alpha - Generator$ determinístico con $\mathcal{L}(Q)$ DI, entonces la st-IPN genera el mismo lenguaje observado, por lo tanto el problema de identificación de SED estocásticos es solucionado.

5.5.2. Reconstrucción del Lenguaje Global a partir del Lenguaje de los Subsistemas.

Como el objetivo es encontrar el lenguaje del sistema global, éste puede ser hallado a partir del lenguaje de los subsistemas. El lenguaje del sistema global se construye a partir de la secuencia de eventos sincronizados de sus subsistemas basándose en la operación de splice (ver Definición 30).

Por lo tanto, el lenguaje del sistema global se construye a partir del splice sincronizado temporizado de los lenguajes de los subsistemas.

La st-IPN del sistema global se halla por medio de la operación de producto síncrono de st-IPNs, definida en la Definición 31.

5.6. Ejemplo de Aplicación

El sistema mostrado en la Figura 5.6.1, es un sistema de calefacción de aire centralizado (AHS). El objetivo es generar calefacción a tres habitaciones. Cada habitación tiene un ventilador que fluye aire caliente, calentado con agua caliente. El flujo de agua es controlado por un sistema de bomba - válvula. Además tiene un calentador central que provee de agua caliente a cada habitación y dos válvulas (v_{c1} y v_{c2}) que controlan el flujo de agua a través de todo el sistema. De acuerdo a estas características, el sistema puede ser dividido en cinco subsistemas: el subsistema 1 comprende el sistema de calefacción central (calentador, bomba principal (p_m) y válvula de reflujo (v_r)). El calentador puede estar en tres estados (0, 1, 2); cada estado es definido por el número de resistencias que ha activado h_1 y h_2, así, el estado 0 no ha activado resistencias, el estado 1 activa una resistencia y en el estado 2 se han activado ambas. El subsistema 2 es el sistema de distribución (v_{c1} y v_{c2}). Y los subsistemas 3, 4 y 5 son los sistemas de calefacción locales.

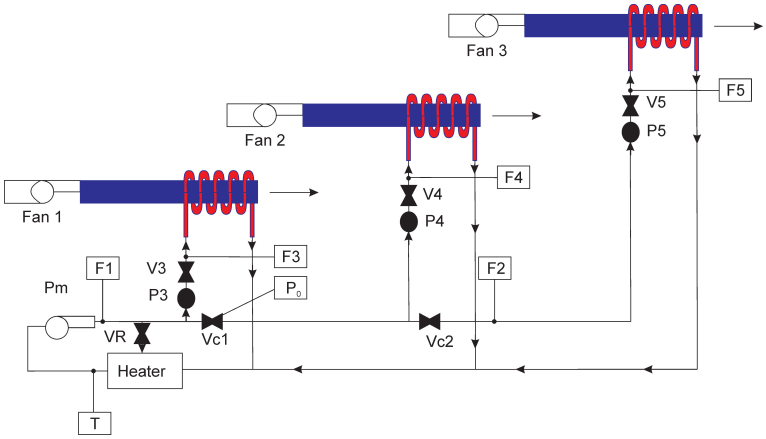


Figura 5.6.1: Sistema AHS

El sistema está equipado con un conjunto de sensores que se detallan en la Tabla 5.6.1. Cada subsistema incluye un sensor de flujo (F_i) que mide la presencia o ausencia de flujo en el subsistema. Sin embargo, el flujo bajo es afectado cuando otro subsistema es activado. Así, un sensor software (NF_i) es diseñado en orden a medir la desviación sobre el flujo de operación normal, teniendo en cuenta la activación de otros subsistemas.

El sistema también incluye sensores binarios de temperatura, sensores de presión y un sensor de posición para la válvula vc_1.

$u_{l,s}y_{l,j}$ es un símbolo de E/S, por ejemplo, $(u_{1,30}y_{1,9})$ es un símbolo de E/S en el subsistema 1, cuyos valores son: $[11110, 1001]$, es decir, $\left[v_r p_g v_g h_1 \overline{h}_2, T_o \overline{T}_1 \overline{T}_2 F_1\right]$.

5.6.1. Funcionamiento del sistema

El sistema inicia globalmente con el evento externo que enciende el sistema: "*Son*". Los subsistemas de calefacción son localmente iniciados con los eventos *"Ca3"*, "Ca*4*" y *"Ca5"*. Estos son eventos externos que cambian la estrategia del controlador (ver Figura 5.6.2); cada combinación temporizada de estos eventos externos constituye un modo de operación del sistema, como el mostrado en la Figura 5.6.3.

Cuando el evento externo *"Son"* es generado, el controlador abre la válvula de reflujo (v_r) e inicia la bomba principal (p_m), el calentador inicia el proceso con un set point de temperatura a *Tp1* y el agua fluye a través del Calentador - Bomba principal - Válvula de Reflujo - Calentador. Cuando algún subsistema es activado (*"Ca*3", *"Ca4"* o *"Ca*5"), el subsistema 1 cierra la válvula de reflujo ($\overline{v}_r$) y cambia el set point de la temperatura a *Tp2 (Tp2>Tp1)*. Si *"Ca4"* y/o *"Ca5"* son iniciadas, entonces, las válvulas 1 y 2 correspondientes

del subsistema 2 (v_{c1},v_{c2}), son abiertas. Por otra parte, las válvulas y bombas locales son iniciadas cuando sea necesario (v_3, v_4, v_5 y p_3, p_4, p_5). Cada subsistema puede ser localmente detenido con los eventos *"$\bar{Ca}3$"*, *"$\bar{Ca}4$"* y *"$\bar{Ca}5$"* ($\overline{Ca} = \bar{Ca}3 \wedge \bar{Ca}4 \wedge \bar{Ca}5$). Luego, el controlador realiza la acción de cierre, cierra las válvulas ($\overline{v}_{c1}, \overline{v}_{c2}\overline{v}_3, \overline{v}_4, \overline{v}_5$) y detiene las bombas ($\overline{p}_3, \overline{p}_4, \overline{p}_5$). Cuando todos los subsistemas están totalmente parados, el sistema puede ser globalmente parado con el evento *"Soff"*. Entonces, el controlador cierra la válvula de reflujo y detiene la bomba principal ($\overline{p}_m$).

Tabla 5.6.1: Lecturas de sensor en el sistema AHS

Sensor	Sub	Lectura	Significado
Temperatura	1	T0, T1, T2,	Si $T < Tp_1$ $T = T_0$, si $Tp_1 < T < Tp_2$, $T = T_1$, si $T_2 > Tp_2$ $T = T_2$.
Flujo	1	F1, NF1,	Flujo, Flujo normal.
Posición	2	Po	Válvula Vc_1abierta o cerrada.
Flujo	2	F2, NF2	Flujo, Flujo normal.
Flujo	3	F3, NF3	Flujo, Flujo normal.
Flujo	4	F4, NF4	Flujo, Flujo normal.
Flujo	5	F5, NF5	Flujo, Flujo normal.
Presión.	3	P3	Presión.
Presión.	4	P4	Presión.
Presión.	5	P5	Presión.

Con el objeto de mostrar las ventajas del método de identificación propuesto, se realizarán una serie de simulaciones sobre el sistema basadas en varios modos de operación del sistema.

Debido a que el sistema no puede trabajar si "*Son*" no es activado, hay nueve posibles modos de operación $OM = \{o_0, \cdots, o_8\}$;

donde $o_0 = Soff, C\bar{a}3, C\bar{a}4, C\bar{a}5$ $o_1 = Son, C\bar{a}3, C\bar{a}4, C\bar{a}5$; $o_2 = Son, Ca3, C\bar{a}4, C\bar{a}5$; $\cdots$.Además cada test incluye las operaciones de inicialización y finalización; por lo tanto la prueba o_x, incluye la secuencia $(o_0 - o_1 - o_x - o_1 - o_0)$.

La primera simulación a realizar se hace cuando solo opera el subsistema 3, (ver Figura 5.6.3).

En la segunda simulación funcionan todos los subsistemas como se observa en la Figura 5.6.4.

Y la tercera simulación presenta todos los posibles modos de operación dispuestos de manera consecutiva en una línea de tiempo, como se observa en la Figura 5.6.5

5.6.2. Identificación de Escenario 1.

5.6.2.1. Generación de Eventos (Algoritmo 5.1)

El número de subsistemas es 5 y las condiciones iniciales son:

- Definición del estado inicial: $\omega^0 = \{\omega_1^0.\omega_2^0.\omega_3^0.\omega_4^0.\omega_5^0\}$;

  $\omega_1^0 = \left[\overline{v}_r\overline{p}_g\overline{v}_g\overline{h}_1\overline{h}_2, T_o\overline{T}_1\overline{T}_2\overline{F}_1\right] = (u_{1,0}y_{1,8})$;

  $\omega_2^0 = \left[\overline{v}_{c1}\overline{v}_{c2}, \overline{P}_o\overline{F}_2\overline{NF}_2\right] = (u_{2,0}y_{2,0})$;

  $\omega_3^0 = \left[\overline{p}_3\overline{v}_3, \overline{F}_3\overline{NF}_3\right] = (u_{3,0}y_{3,0})$;

  $\omega_4^0 = \left[\overline{p}_4\overline{v}_4, \overline{F}_4\overline{NF}_4\right] = (u_{4,0}y_{4,0})$;

  $\omega_5^0 = \left[\overline{p}_5\overline{v}_5, \overline{F}_5\overline{NF}_5\right] = (u_{5,0}y_{5,0})$.
- Función de Salida Inicial

  $\varphi\left(m\left(p_{1,0}\right)\right) = y_{1,8}/\overrightarrow{0}_4$;

  $\varphi\left(m\left(p_{2,0}\right)\right) = y_{2,0}/\overrightarrow{0}_3$;

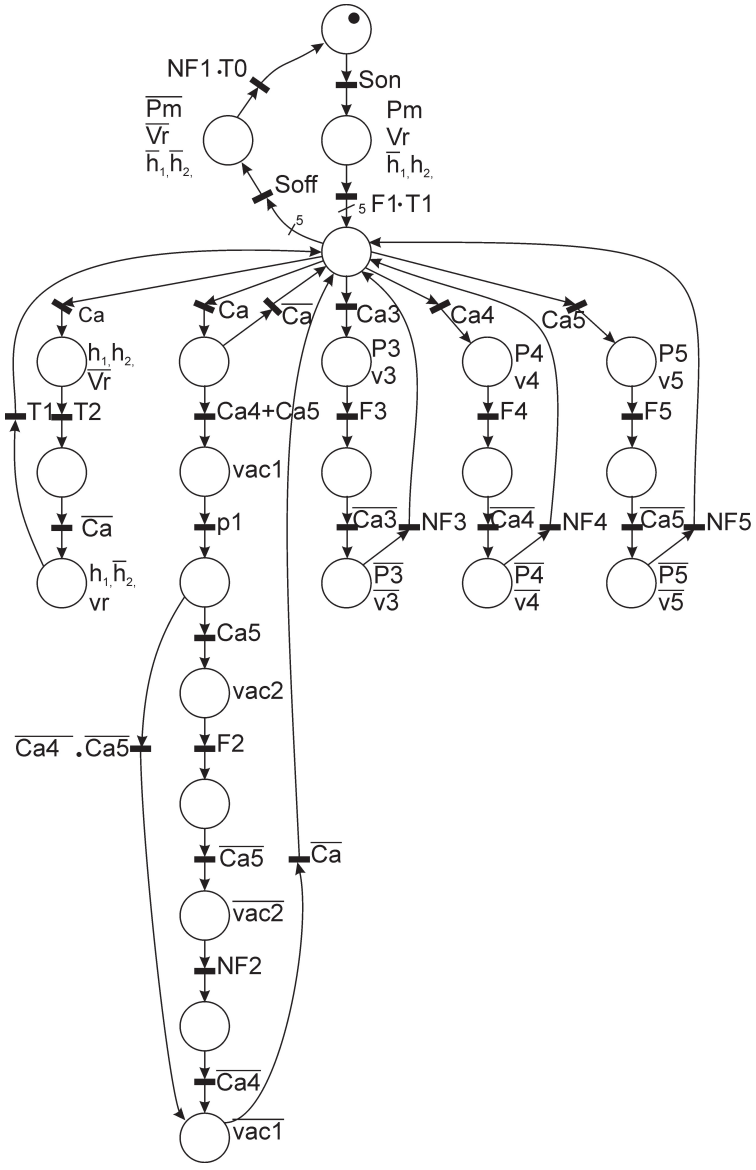


Figura 5.6.2: Controlador del sistema AHS

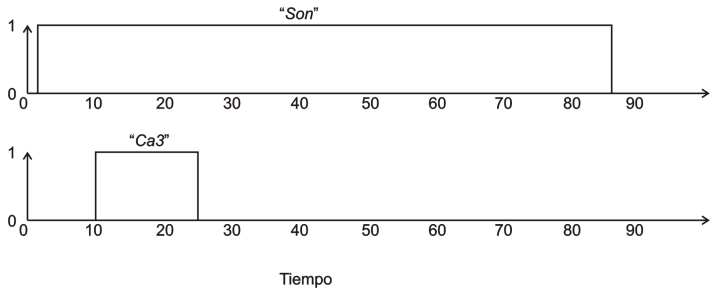


Figura 5.6.3: Escenario1. $o_0 - o_1 - o_2 - o_1 - o_0$

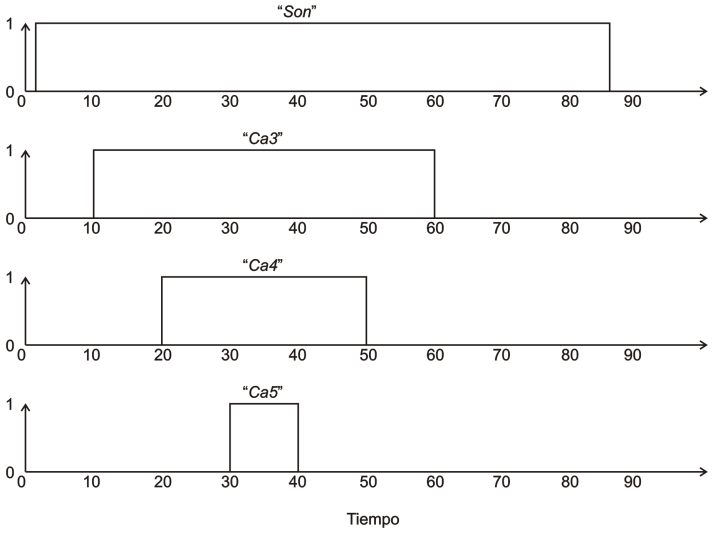


Figura 5.6.4: Escenario 2. $o_0 - o_1 - o_2 - o_5 - o_8 - o_5 - o_2 - o_1 - o_0$

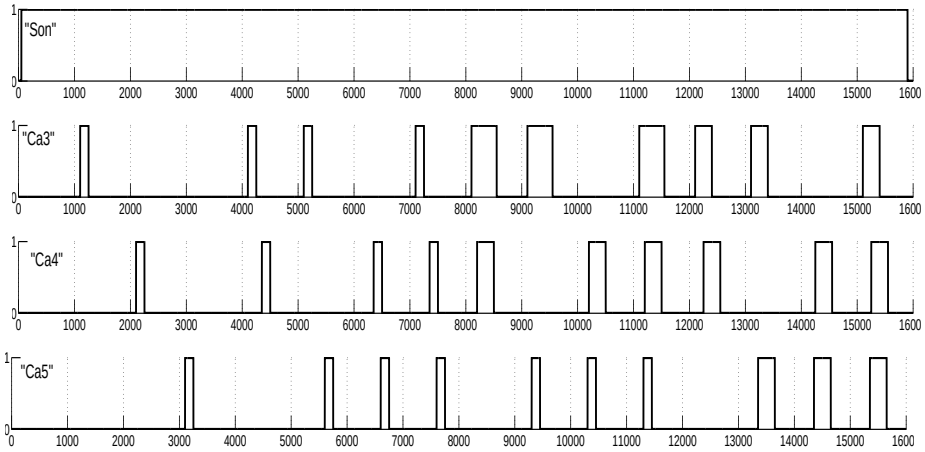


Figura 5.6.5: Escenario de funcionamiento total

$$\varphi\left(m\left(p_{3,0}\right)\right) = y_{3,0}/\overrightarrow{0}_2;$$
$$\varphi\left(m\left(p_{4,0}\right)\right) = y_{4,0}/\overrightarrow{0}_2;$$
$$\varphi\left(m\left(p_{5,0}\right)\right) = y_{5,0}/\overrightarrow{0}_2.$$

El vector de entradas es un arreglo de órdenes de control, basado en el funcionamiento del sistema descrito en la Subsección 5.6.1 y el vector de salidas es un arreglo de lecturas sensoriales, basadas en la Tabla 5.6.1.

El sistema es inicializado externamente con *"Son"*; el controlador genera las órdenes de control cuando evolucionan las entradas externas. Entonces el sistema reacciona y el generador de eventos empieza a generarlos. Es de notar que el sistema inicia en modo de operación o_0, luego cambia a o_1, y, cuando "$Ca3$" es externamente activado, cambia a o_2.

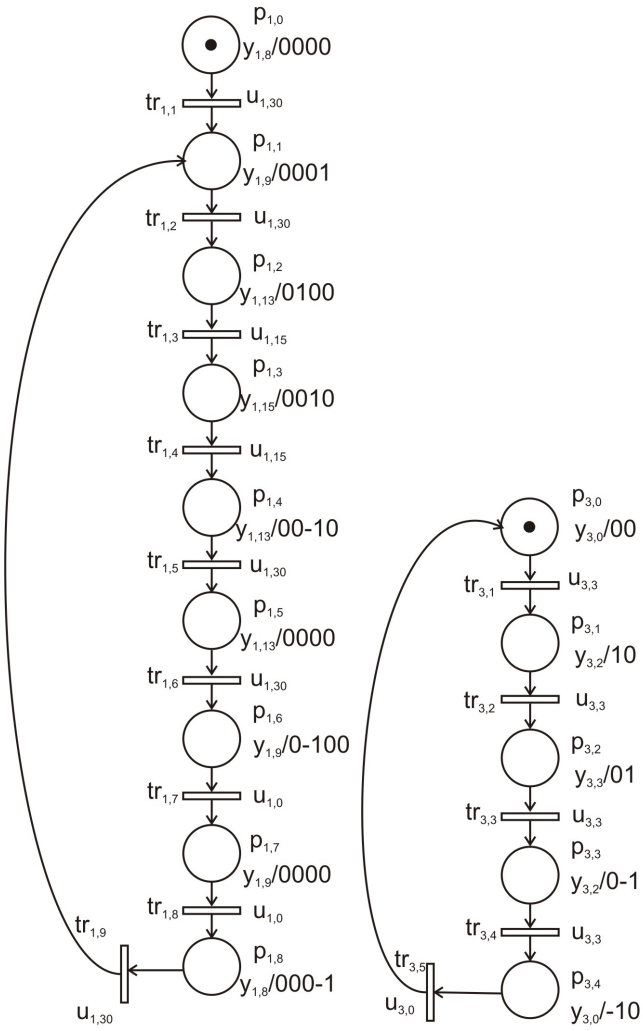
Con cada nuevo evento, el Algoritmo 5.4 crea nuevos lugares y transiciones. En el instante τ_0, los eventos generados son: $(u_{1,0}y_{1,8})$, en el subsistema 1 y $(u_{3,0}y_{3,0})$, en el subsistema 3; las st-IPNs de

estos subsistemas se encuentran en el estado inicial, con sus correspondientes funciones de salida; como se indica en la Figura 5.6.6.

En el instante τ_1, el evento generado es: $(u_{1,30}y_{1,9})\,.t^1_{l,o_1}$, $t^1_{l,o_1} = 21\ seg$, lo que indica que la orden de control es local (solo hay un subsistema involucrado); el subsistema es el 1, se compara frente al evento anterior en el subsistema 1 $(u_{1,0}y_{1,8})$ y se verifica que hay cambio de estado, porque $y_{1,9} \neq y_{1,8}$, se actualiza la st-IPN y el tiempo del evento se adiciona a δ_1 en la posición $(1, 1, 2)$: transición 1, evento número 1 y modo de operación o_1. De esta manera se continua con el proceso de identificación on-line.

Después de completar un ciclo del proceso $Soff$ es generado, el Algoritmo 5.4 crea las st-IPNs para los subsistemas 1 y 3, mostrados en la Figura 5.6.6.

En esta etapa de identificación, lugares y transiciones son definidos y el modelo de las st-IPNs identificadas explican el comportamiento del sistema. Inicialmente el sistema trabaja en modo de operación o_0 porque *"Son"* está activo. El Subsistema 1 inicia en el lugar $p_{1,0}$; cuando *"Son"* es activado el sistema cambia a modo de operación o_1, el controlador abre las válvulas de reflujo y la principal, enciende la bomba general y configura el calentador en modo 1 $(tr_{1,1})$. Un cambio de estado $(p_{1,1})$ es generado, porque el agua fluye a través del ciclo Calentador - PG - VR - Calentador $(F_1 = 1)$ como es mostrado en la Figura 5.6.1. Después de un tiempo $(tr_{1,2})$ la temperatura alcanza el valor de Tp_1 (18^oC), por lo tanto, el sistema evoluciona a otro estado $(p_{1,2})$. El controlador inicia la acción de calentar, cierra la válvula de reflujo y configura el calentador en modo 2 $(tr_{1,3})$, el cambio de estado no puede ser medido por ausencia de sensores que detecten la operación de la válvula de reflujo $(p_{1,3})$. En este instante *"Ca3"* es activado; así el sistema cambia a modo de operación o_2 entonces el controlador enciende la bomba 3 y abre la válvula 3 $(tr_{3,1})$ y el subsistema 3 cambia su estado a $p_{3,1}$ porque el agua está fluyendo a través del ciclo Calen-



(a) Subsistema 1 **(b)** Subsistema 3

Figura 5.6.6: st-IPNs escenario 1de simulación del sistema AHS

tador - Pm - P3 - V3 - Calentador. Después de un tiempo ($tr_{3,2}$), el flujo alcanza un nivel normal (NF_3) y el subsistema 3 cambia de estado a $p_{3,2}$. El subsistema 1 continua con la válvula de reflujo cerrada ($tr_{1,4}$), permanece hasta alcanzar Tp_2 (28^{o}C) y cuando Tp_2 es alcanzada, evoluciona a $p_{1,4}$. El subsistema 3 permanece en $p_{3,2}$ hasta que $Ca3$ es desactivada ($\bar{Ca}3$) y o_1 es alcanzado, entonces el flujo normal cae ($tr_{3,3}$) y $p_{3,3}$ es alcanzado. Ahora, el controlador apaga la bomba 3 y cierra la válvula 3, ($tr_{3,4}$). Como consecuencia el flujo en el subsistema decrece, (cambia a $p_{3,4}$) hasta que el nivel del flujo del subsistema es bajo F_3, ($tr_{3,5}$). Esta transición hace que el subsistema retorne a su estado inicial ($p_{3,0}$). Ahora, el controlador cierra la válvula de reflujo y configura el calentador en modo 1, ($tr_{1,5}$). El subsistema 1 mantiene el estado ($p_{1,5}$) hasta que la temperatura está por debajo de Tp_2, ($p_{1,6}$), después, el controlador apaga la bomba, cierra la válvula general y configura el calentador en modo 0, ($tr_{1,7}$). El subsistema 1 evoluciona a $p_{1,7}$, y después de un tiempo ($tr_{1,8}$) a $p_{1,8}$ y la temperatura decrece hasta alcanzar el estado inicial ($p_{1,0}$). Al finalizar, Son es desactivado ($Soff$) y el sistema cambia a modo de operación o_0.

Sin embargo, cada transición tiene solo un dato de tiempo, por lo que no es posible definir el tipo de distribución a la que se ajustan los datos.

5.6.2.2. Identificación temporizada

De acuerdo a la recomendación presentada en la Subsección 5.2.4, al menos 5 muestras son necesarias para identificar el tipo de distribución, así el escenario 1 se repetirá hasta que cada transición, en cada subsistema y en cada modo de operación sea identificada.

Entonces, después de 5 observaciones, el Algoritmo 5.2 es ejecutado. Como ejemplo se analizan los datos para la transición con

mayor dispersión ($tr_{3,2}$), en este escenario en el modo de operación 2 (o_2). El vector de tiempo está segundos.

- Entradas: $\delta_3(2,:,3) = \left[t^1_{3,o_2} \cdots t^5_{3,o_2}\right]$; $\alpha = 0{,}05$; $\alpha_{ct} = 0{,}05$; $d_{op} = 0{,}5$; $CD = \{normal, exponencial\}$

  $l = 3$, $r_l = 2$.

  - para $n = 5$;

    $\delta_3(2,:,3) = [28, 29, 26, 29, 28]$;

    $\bar{X} = 28$, $S = 1{,}2247$, $d = 1{,}1036$.

Como $d > 0{,}05$, no hay suficientes datos para que la muestra sea representativa.

Después de 21 observaciones, el Algoritmo 5.2 da un resultado positivo: $\bar{X} = 28{,}062$, $S = 1{,}123$; $d = 0{,}49$. Como $d < 0{,}5$, el Algoritmo 5.3 es ejecutado y se calcula el tipo de distribución y sus parámetros, presentando los siguientes resultados:

$\alpha_{ct} = 0{,}05$. $p{-}value_{\alpha_{ct}} = 0{,}28724$ en tabla de Kolmogorov-Smirnov con $n = 21$.

- Test de Hipótesis:

  $H_0 : F_0(X) \backsim N(\mu, \sigma^2)$;

  $H_1 : F_0(X) \nsim N(\mu, \sigma^2)$.

  - Parámetros estimados $F_0 = N(\mu, \sigma^2) = N(28{,}0625, 1{,}1236)$ y $p - value = 0{,}1621$ basado en el test de Kolmogorov-Smirnov.

  Se acepta H_0 ya que $p - value \leq p - value_{\alpha_{ct}}$. Por lo tanto los datos están normalmente distribuidos.

La Figura 5.6.7 muestra el comportamiento de la media a partir de los datos del vector de tiempo para la transición $tr_{3,2}$ versus el número de muestras requeridas en el modo de operación o_2. A

medida que aumenta el número de observaciones que se realizan al proceso, el tiempo medio se estabiliza, constituyendo estos datos en parámetros para determinar el número de muestras necesarias para realizar correctamente la identificación en un proceso estocástico. Se puede observar que a partir de 21 muestras la media se estabiliza.

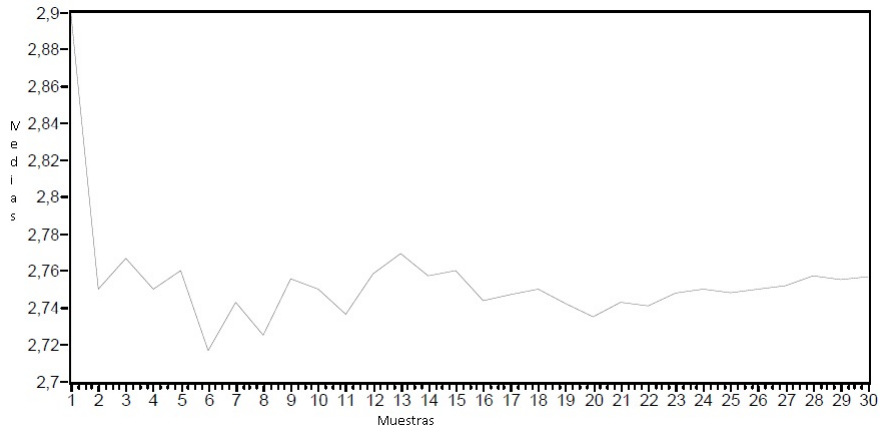


Figure 5.6.7: Comportamiento temporizado de la transición $tr_{3,2}$

En conclusión, para realizar la identificación total del escenario 1, $o_0 - o_1 - o_2 - o_1 - o_0$, es necesario realizar 21 muestras de cada transición temporizada. Esta identificación incluye las estructuras de las st-IPNs y de todas las funciones de distribución. Como el tiempo en las transiciones depende del modo de operación en el cual el sistema se encuentre, porque la respuesta de los dispositivos se ve afectada por la interacción con otros eventos externos, la información de las transiciones se presenta en la Tabla 5.6.2.

Tabla 5.6.2: Identificación de transiciones temporizadas en escenario 1

Transición	o_1	o_2
$tr_{1,1}$	21	
$tr_{1,2}$	$N(3,88;0,33)$	
$tr_{1,3}$		$N(85,12;0,33)$
$tr_{1,4}$	151	
$tr_{1,5}$	4	
$tr_{1,6}$	588	
$tr_{1,7}$	5	
$tr_{1,8}$	$N(7,87;0,33)$	
$tr_{1,9}$	$N(156,12;0,33)$	
$tr_{3,1}$		854
$tr_{3,2}$		$N(28,06;1,12)$
$tr_{3,3}$	$N(115;1,15)$	
$tr_{3,4}$	3	
$tr_{3,5}$	1	

5.6.2.3. Lenguaje Generado

En esta sección se muestra cómo el lenguaje observado en el escenario 1, se obtiene en los subsistemas 1 y 3 y en el sistema global.

El lenguaje observado para cada subsistema es igual a los generados por las st-IPNs:

$\mathcal{L}_1 = (u_{1,0}y_{1,8})\,(u_{1,30}y_{1,9})\,(u_{1,30}y_{1,13})\,(u_{1,15}y_{1,15})\,(u_{1,15}y_{1,13})\,(u_{1,30}y_{1,13})$ $(u_{1,30}y_{1,9})\,(u_{1,0}y_{1,9})\,(u_{1,0}y_{1,8})\,(u_{1,30}y_{1,9})$ a instantes $\tau_0, \tau_1, \tau_2, \tau_4, \tau_9, \tau_{10}$, $\tau_{11}, \tau_{12}, \tau_{13}, \tau_{14}$.

$\mathcal{L}_3 = (u_{3,0}y_{3,0})\,(u_{3,3}y_{3,2})\,(u_{3,3}y_{3,3})\,(u_{3,3}y_{3,2})\,(u_{3,3}y_{3,0})\,(u_{3,0}y_{3,0})$ a instantes $\tau_0, \tau_3, \tau_5, \tau_6, \tau_7, \tau_8$.

La Figura 5.6.8 muestra el resultado del proceso de identificación en caso que el sistema no haya sido dividido en subsistemas. El lenguaje observado es igual al lenguaje generado por la st-IPN. Entonces el lenguaje global es:

$\mathcal{L}_g = (u_0y_{4096})\,(u_{7680}y_{4608})\,(u_{7680}y_{4608})\,(u_{7680}y_{6656})\,(u_{3840}y_{6688})$ $(u_{3888}y_{6688})\,(u_{3888}y_{6704})\,(u_{3888}y_{7728})\,(u_{3888}y_{7712})\,(u_{3840}y_{7680})\,(u_{7680}y_{7680})$ $(u_{7680}y_{6656})$
$(u_0y_{4608})\,(u_0y_{4608})\,(u_0y_{4096})\,(u_0y_{4096})$ a instantes $\tau_0, \cdots, \tau_{15}$.

Un símbolo de entrada global es:

$$u_g = \left\{\overbrace{v_r p_g v_g h_1 h_2}^{u_1}\overbrace{v_{c1}}^{u_2}\overbrace{v_{c2}p_3}^{u_3}\overbrace{v_3p_4}^{u_4}\overbrace{v_4p_5}^{u_5}\overbrace{v_5}\right\}.$$

Y un símbolo de salida global es:

$$y_g = \left\{\overbrace{T_0T_1}^{y_1}\overbrace{T_2F_1P_oF_2}^{y_2}\overbrace{NF_2F_3}^{y_3}\overbrace{NF_3F_4}^{y_4}\overbrace{NF_4F_5}^{y_5}\overbrace{NF_5}\right\}.$$

Es de notar que, por ejemplo, en el instante τ_3, el símbolo de entrada en el lenguaje global es u_{7680} y en el instante τ_4 es u_{3840} (ver Figura 5.6.8). En esos instantes, los símbolos de entrada en los lenguajes locales son $u_{1,30}$ y $u_{3,3}$.

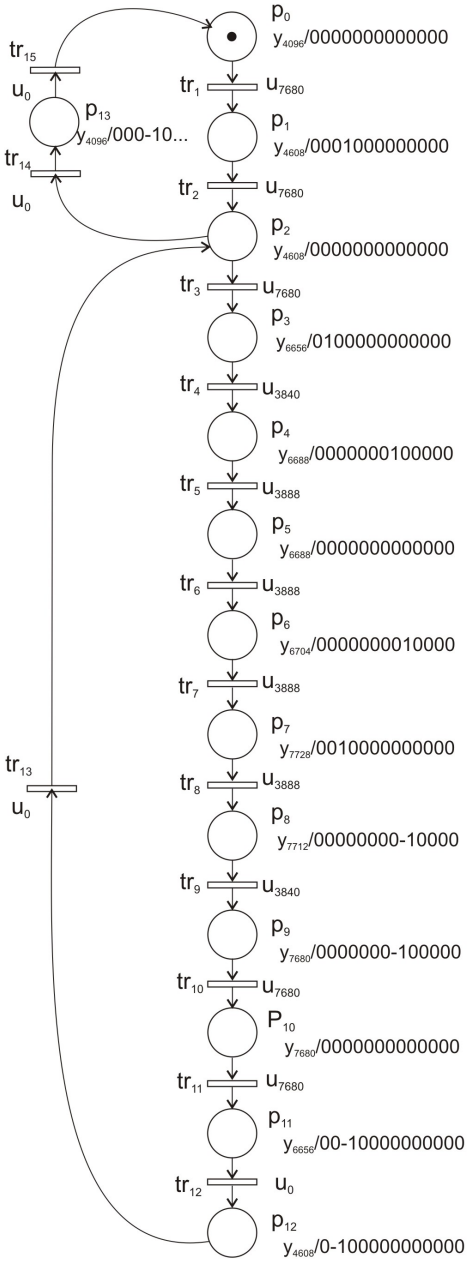


Figura 5.6.8: st-IPN del sistema AHS global

De la misma manera, en los mismos instantes, el símbolo de salida en el lenguaje global es y_{6656} y y_{6688} en τ_3 y τ_4, respectivamente (ver Figura 5.6.8). Y los símbolos de salida en los lenguajes locales son: $y_{1,13}$ y $y_{3,2}$. La Tabla 5.6.3 muestra estos símbolos.

Tabla 5.6.3: Símbolos de E/S en τ_3 y τ_4

	τ_3					τ_4				
u_g	11110	00	00	00	00	01111	00	11	00	00
u_1	11110					01111				
u_2		00					00			
u_3			00					11		
u_4				00					00	
u_5					00					00

	τ_3					τ_4				
y_g	1101	000	00	00	00	1101	000	10	00	00
y_1	1101					1101				
y_2		000					000			
y_3			00					10		
y_4				00					00	
y_5					00					00

De acuerdo a la definición 30, un símbolo global de E/S, ω^i, se obtiene a partir del splice de $\omega^i = \omega_1^i \oplus \omega_2^i \oplus \cdots \oplus \omega_5^i$ para $i = 3, 4$. Por lo tanto, el enfoque global y el de subsistemas representan el mismo lenguaje. Entonces, el splice temporal sincronizado de $\mathcal{L}_1$ y $\mathcal{L}_3$ da $\mathcal{L}_g$.

Además, la ventaja de usar subsistemas es que el método identifica solo lenguajes activos, lo cual permite determinar qué subsistema está funcionando.

5.6.3. Identificación de Escenario 2.

Sí, se cambia el funcionamiento del sistema al escenario 2 (Figura 5.6.4), las st-IPNs identificadas se muestran en la Figura 5.6.9

Cómo se puede observar en la Figura 5.6.9, las st-IPNs para los subsistemas 1 y 3, tienen la misma estructura que las identificadas en la anterior simulación, pero se generan en instantes de tiempo diferentes; en esta figura también se muestran las st-IPNs para los subsistemas 2, (sistema de distribución), 4 y 5.

El método de identificación es aditivo, entonces después de la identificación de un lenguaje asociado a un escenario particular, las redes identificadas crecen con las secuencias de eventos nuevos, si es necesario.

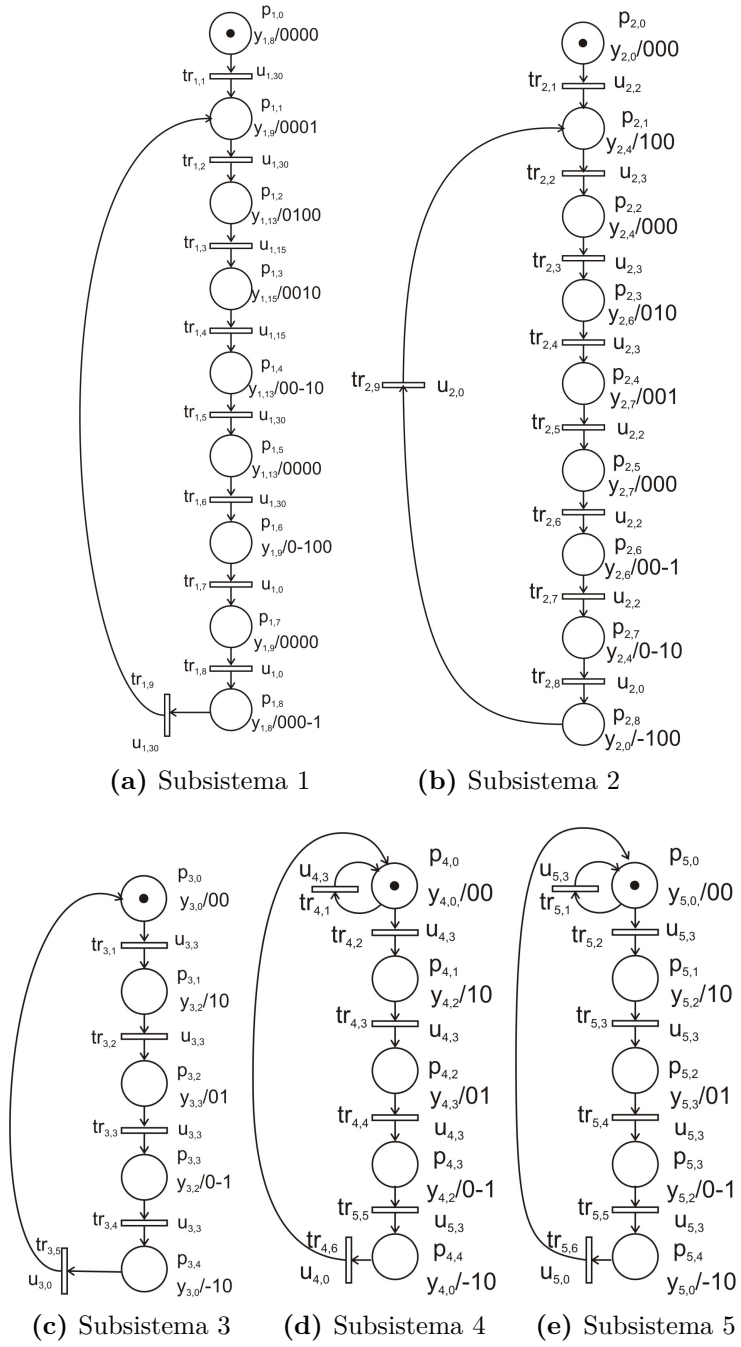
Aunque el lenguaje generado por el sistema es más complicado con esta nueva estrategia, las st-IPNs no han crecido en complejidad, sólo se han incluido los subsistemas que no estaban representados.

La Tabla 5.6.4, muestra los resultados de los tiempos identificados en los diferentes subsistemas en cada uno de los modos de operación involucrados en el escenario 2.

Tabla 5.6.4: Identificación de transiciones temporizadas en escenario 2

Transición	o_0	o_1	o_3	o_7
$tr_{1,1}$	21			
$tr_{1,2}$	$N(3,8;0,41)$			
$tr_{1,3}$		$N(85,19;0,42)$		
$tr_{1,4}$	501			
$tr_{1,5}$	4			
$tr_{1,6}$	238			
$tr_{1,7}$	5			
$tr_{1,8}$	$N(7,96;0,18)$			

Transición	o_0	o_1	o_3	o_7
$tr_{1,9}$	$N(156, 3; 0, 17)$			
$tr_{2,1}$			201	
$tr_{2,2}$				100
$tr_{2,3}$				13
$tr_{2,4}$				$N(26, 2; 0, 41)$
$tr_{2,5}$				$N(60, 8; 0, 42)$
$tr_{2,6}$				1
$tr_{2,7}$				4
$tr_{2,8}$		96		
$tr_{2,9}$			700	
$tr_{3,1}$		109		
$tr_{3,2}$		$N(28, 1; 1, 2)$		
$tr_{3,3}$	$N(464, 8; 1, 4)$			
$tr_{3,4}$	3			
$tr_{3,5}$	1			
$tr_{4,1}$			209	
$tr_{4,2}$			9	
$tr_{4,3}$			$N(29, 57; 0, 5)$	
$tr_{4,4}$		$N(253, 4; 0, 5)$		
$tr_{4,5}$		8		
$tr_{4,6}$		1		
$tr_{5,1}$				307
$tr_{5,2}$				7
$tr_{5,3}$				$N(26, 2; 0, 41)$
$tr_{5,4}$			$N(61, 8; 0, 41)$	
$tr_{5,5}$			4	
$tr_{5,6}$			1	



(a) Subsistema 1 **(b)** Subsistema 2

(c) Subsistema 3 **(d)** Subsistema 4 **(e)** Subsistema 5

Figura 5.6.9: st-IPNs, AHS Escenario 2

5.6.4. Identificación Completa del Sistema

El último test realizado consiste en la generación secuencial de todas las posibles combinaciones de eventos externos (Figura 5.6.5), así el sistema será capaz de generar un lenguaje realmente complejo. Las st-IPNs de los subsistemas se muestran en la Figura 5.6.10.

Como se puede observar la estructura de las st-IPNs varía muy poco; se adicionan algunas transiciones que corresponden a eventos cuando el subsistema debe iniciar nuevamente.

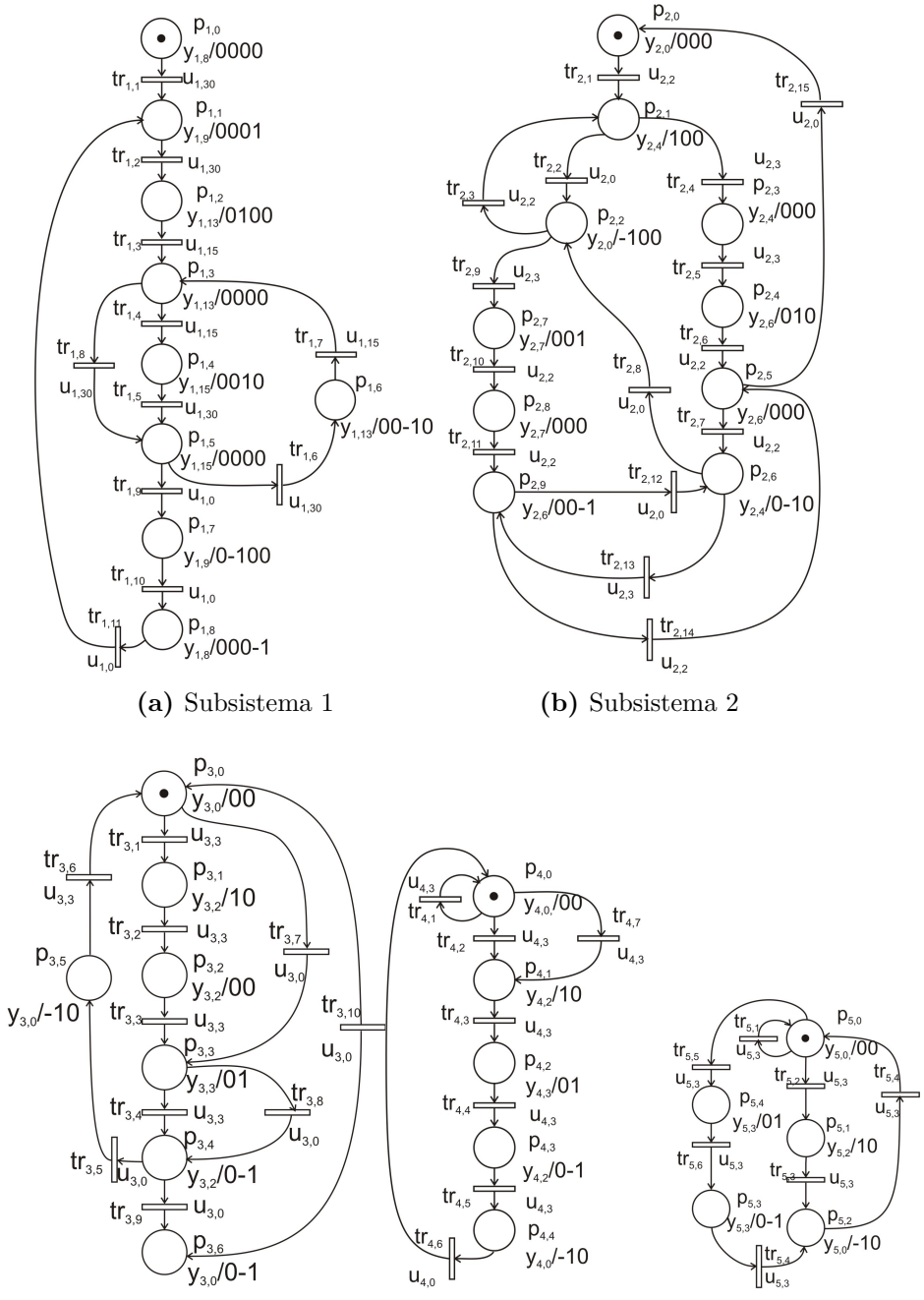
Como el método de identificación es aditivo, después de la identificación de un lenguaje asociado a un modo particular, las redes identificadas crecen con las nuevas cadenas de eventos si fuese necesario.

La ventaja de usar subdivisiones es que el método identifica solo lenguajes activos lo que permite determinar qué subsistemas están operando. Adicionalmente, con el sistema dividido se tiene un mejor manejo de las lecturas sensoriales.

La Tabla 5.6.5 compara el número de estados y de transiciones a medida que los diferentes escenarios se van identificando, con y sin división en subsistemas.

A partir de estos resultados se demuestra la ventaja del método propuesto cuando el sistema se hace más complejo. Hasta el escenario 9, la complejidad (número de lugares y transiciones) de la st-IPN global es similar a las st-IPNs en el caso de identificación en subsistemas. Sin embargo, cuando la complejidad del sistemas se incrementa, la identificación por subsistemas reduce la complejidad.

También se puede observar que la identificación del sistema global, incrementa su complejidad con el número de modos de operación considerados, mientras que la identificación por subsistemas no.



(a) Subsistema 1 **(b)** Subsistema 2

(c) Subsistema 3 **(d)** Subsistema 4 **(e)** Subsistema 5

Figura 5.6.10: st-IPNs de identificación completa

Tabla 5.6.5: Lugares y transiciones

Escenario	Sub		Global	
	Lugares	Tr.	Lugares	Tr.
1	7	2	3	2
2	14	14	14	15
3	22	22	22	23
4	28	32	24	31
5	28	32	24	32
6	28	32	24	32
7	28	32	24	32
8	28	32	24	32
9	28	33	33	42
10	28	33	39	52
11	32	41	44	59
12	32	42	50	68
13	32	43	52	71
14	33	43	54	74
15	34	45	57	78
16	35	48	61	83

Este comportamiento es relativo con el número de estados posibles manejados por la st-IPN. El número máximo de estados en el sistema global es $Q(g)_{max} = 2^{n_t} \times 3^{n_t}$, ($n_t$ es el número total de símbolos de salida) (Ecuación 4.2.3), mientras en la identificación por subsistemas el número máximo de estados es $Q(split)_{max} = Q_{max,1} + Q_{max,2} + \cdots + Q_{max,c}$, es decir, $Q(split)_{max} = 2^{n_1} \times 3^{n_1} + 2^{n_2} \times 3^{n_2} + \cdots + 2^{n_c} \times 3^{n_c}$, donde $n_1 + n_2 + \cdots + n_c = n_t$, se puede comprobar que $Q(g)_{max} > Q(split)$.

Por lo tanto, cuando la complejidad de los sistemas se incrementa, la estrategia de identificación por subsistemas define modelos más simples y más fáciles de interpretar.

5.6.5. Identificación de las transiciones temporizadas en diferentes modos de operación.

Como se mencionó en la Subsección 5.2.4, el tiempo en las transiciones depende del modo de operación del sistema, aunque el dispositivo mantenga su secuencia de operación.

Por ejemplo, el subsistema 3 usa la misma red para cada modo de operación (Figura 5.6.10), pero presenta diferentes distribuciones en cada transición para cada modo de operación, por lo que modela comportamientos complejos.

La Tabla 5.6.6 muestra los parámetros de las distribuciones de la transición $tr_{3,2}$ en el escenario 3, para cada modo de operación.

Tabla 5.6.6: Parámetros de $tr_{3,2}$

Modo de Operación	Distribución
o_0	NA*
o_1	NA
o_2	$N\,(27,5;1,194)$
o_3	NA
o_4	NA
o_5	$N\,(14,4;0,097)$
o_6	NA
o_7	$N\,(3,7;0,049)$

*"No aplica" significa que la transición no se ejecuta en ese modo de operación.

Se puede ver que el comportamiento de los tiempos, obedecen a una distribución normal en todos los modos de operación; pero sus parámetros varían.

5.7. Conclusiones y Aportes del Capítulo

En el presente capítulo se presentó un método de identificación de SED estocásticos a partir de las señales de E/S, las cuáles se convierten en eventos y generan el lenguaje regular que modela el comportamiento del sistema, para ser representado bajo la estructura propuesta en el capítulo de modelado.

El algoritmo de identificación propuesto usa datos E/S, on-line, para identificar el comportamiento de cada subsistema y define las st-IPNs que generan el mismo lenguaje que el observado.

La metodología de identificación es incremental y está directamente relacionada con el objeto de modelado, es decir, modelado de dispositivos que componen el sistema de una manera constructivista. Este enfoque permite identificar todos los comportamientos observables de los dispositivos.

Algunas de las principales ventajas de la metodología, (con respecto a resultados existentes) es que maneja una gran cantidad de entradas y salidas con un bajo costo computacional, no es necesario tener conocimiento previo del número de lugares ni del número de transiciones, genera modelos de tamaño reducido e incluye información temporizada. Además, esta propuesta incluye un procedimiento para determinar si el sistema ha sido totalmente identificado, lo cual no ha sido propuesto en otras investigaciones.

Por lo tanto, el principal aporte del capítulo es el formalismo del método responsable de la identificación, la comprobación de las propiedades de identificabilidad y el algoritmo en sí mismo. Los resultados de la aplicación ilustran el rendimiento del algoritmo en sistemas simulados y la aplicabilidad a sistemas de gran escala.

6 Diagnóstico de Fallos

El diagnóstico de fallos es un tema que ha sido tratado desde diferentes esquemas (Ver Sección 3.3). Teniendo en cuenta las restricciones de los enfoques analizados en el capítulo 3, en esta sección se presenta un método para detectar y aislar fallos on-line, en SED estocásticos sin modelo previo, que mejore los resultados de los enfoques analizados en la Sección 3.4.

La tarea de diagnóstico implica la detección, el aislamiento y la identificación del fallo. La detección en la presente propuesta se lleva a cabo comparando una secuencia de eventos observados con el lenguaje normal identificado con los resultados del método de identificación, desarrollado en el Capítulo 5; el aislamiento se puede hacer por la división modular del sistema que permite localizar de manera más precisa las señales que no han funcionado como se esperaba y la identificación se realiza, en cada caso mediante el conocimiento de los expertos del sistema.

El diagnosticador propuesto es un generador de lenguaje representado como una red de Petri interpretada coloreada, con una estructura particular capaz de representar secuencias de eventos de fallos. Una vez que un evento de fallo es detectado, un algoritmo de aprendizaje modifica la estructura de la red, así el diagnosticador aprende la secuencia de fallo.

6.1. Sistema a Diagnosticar

El sistema a diagnosticar es un SED estocástico cuyo comportamiento se describe por la interrelación de señales de E/S, con la misma estructura que la presentada en la Figura 4.3.1 y con las restricciones presentadas en la Subsección 4.1.1. El comportamiento del sistema se describe bajo lenguajes formales basados en secuencias de eventos; además el sistema se puede dividir en subsistemas, donde cada subsistema es una parte del sistema global con un comportamiento particular.

6.1.1. Flujo Compartido

Un aspecto muy importante a tener en cuenta en el proceso de diagnóstico de un sistema en particular, es analizar si existen sensores que midan flujo compartido relacionado con datos, materiales, o energía, entre subsistemas. Es decir, verificar si existen lecturas sensoriales que midan flujo y que se vean afectadas por el comportamiento de dicho flujo, en otro subsistema. La consecuencia del flujo compartido, desde el punto de vista de diagnóstico de fallo, es que un subsistema que opera con normalidad, en el sentido que ningún fallo ha ocurrido, podría alcanzar un estado erróneo no descrito en su comportamiento normal cuando el subsistema no recibe el respectivo servicio de flujo de otro subsistema conectado a este. El flujo detenido podría ser ocasionado por fallos en otro subsistema arriba o hacia abajo de la línea de proceso, [100]; por lo tanto, el modelo de un subsistema sufre acoplamiento por la interacción entre subsistemas, [100] y por fallos de propagación, [101].

Con el objeto de incluir este hecho en el diagnosticador, se tiene en cuenta la noción de sensores de flujo compartido (SFS_r), donde SFS_r es el conjunto de todos los sensores que miden flujo, discriminados por subsistemas.

6.2. Comportamiento de Fallo

El comportamiento de un sistema puede ser descrito bajo lenguajes formales basados en secuencias de eventos.

El conjunto de eventos del sistema a diagnosticar en esta investigación se divide en eventos observables y eventos no-observables (ver Subsección 4.2.3), $\Omega_l = \Omega_{o_l} \cup \Omega_{uo_l}$, donde Ω_{uo_l} incluye dos subconjuntos: eventos no-observables por fallo y eventos no-observables regulares $\Omega_{uo_l} = \Omega_{f_l} \cup \Omega_{reg_l}$, (adaptado de [102]). Donde cada $\omega_l^i \in \Omega_l$ es un evento compuesto temporizado: $\omega_l^i = (u_{l,s} y_{l,j}) .t_{l,o_{om}}^{ev}$. (Ver Definiciones 2 y 4).

El conjunto de eventos compuestos temporizados normales, se define como: $\Omega_{N_l} \subset \Omega_{o_l}$, subconjunto de los eventos observables de un subsistema l; tal que el lenguaje normal de un subsistema l es: $\mathcal{L}^{N_l} \subset \Omega_{N_l}^*$.

Entonces, el comportamiento de fallo del sistema a diagnosticar se describe a partir de secuencias de eventos en las cuales existe al menos un evento de fallo; teniendo en cuenta que un fallo se define como:

Definición 36 (Evento de Fallo Temporizado)**.** Dada una secuencia de eventos s y un evento de la forma: $\omega_l^{i*} = (u_{l,s} y_{l,j}) .t_{l,o_{om}}^{ev}$; si $s \in \mathcal{L}^{N_l} \wedge s\omega_l^{i*} \notin \mathcal{L}^{N_l}$ entonces ω_l^{i*} es un evento de fallo temporizado.

Un evento, ω_l^i no pertenece al lenguaje normal (basado en la definición 6) $\omega_l^i = \left(u_{l,s}^i y_{l,j}^i\right) .t_{l,o_{om}}^{ev} \notin \mathcal{L}^{N_l}$[1], si

i) $Pc_{\Omega_{N_l}}\left(u_{l,s}^i\right) \notin \mathcal{L}_{in}^{N_l} \wedge Pc_{\Omega_{N_l}}\left(y_{l,j}^i\right) \in \mathcal{L}_{out}^{N_l}$ o

ii) $Pc_{\Omega_{N_l}}\left(u_{l,s}^i\right) \in \mathcal{L}_{in}^{N_l} \wedge Pc_{\Omega_{N_l}}\left(y_{l,j}^i\right) \notin \mathcal{L}_{out}^{N_l}$ o

iii) $Pc_{\Omega_{N_l}}\left(u_{l,s}^i\right) \notin \mathcal{L}_{in}^{N_l} \wedge Pc_{\Omega_{N_l}}\left(y_{l,j}^i\right) \notin \mathcal{L}_{out}^{N_l}$ o

[1]El superíndice "i" se ha adicionado para identificar el instante del evento.

iv) $Pc_{\Omega_{N_l}}\left(u^i_{l,s}\right) \in \mathcal{L}^{N_l}_{in} \wedge Pc_{\Omega_{N_l}}\left(y^i_{l,j}\right) \in \mathcal{L}^{N_l}_{out}$ pero $t^{ev}_{l,o_{om}} \notin [ab]$, donde $[a, b]$ es el intervalo de confianza calculado como $\int_a^b f(t) \geq (1-\alpha)$, donde $1-\alpha$ es el nivel de confiabilidad y $f(t)$ es la función de densidad de probabilidad del evento normal.

El caso *i)* significa que el símbolo de entrada del evento dado no pertenece al comportamiento normal, pero el símbolo de salida si; en el caso *ii)* la situación es la contraria; en el caso *iii)* ninguno de los símbolos pertenece al comportamiento normal y el caso *iv)* ambos símbolos pertenecen al comportamiento normal, pero el tiempo del evento no está dentro del intervalo de confianza.

Por lo tanto, los casos *i), ii)* y *vi)* son eventos de fallo, cuyo efecto se puede observar y en el caso *iii)* el evento de fallo es no-observable, no-detectable.

El lenguaje de fallo se define como:

Definición 37 (Lenguaje de Fallo, $\mathcal{L}^{F_l}$)**.** Dada una secuencia de eventos temporizados s el lenguaje de fallo es $\mathcal{L}^{F_l} = \left\{st \mid s \in \mathcal{L}^{N_l} \wedge t \in \mathcal{L}/s \wedge \exists \omega^{i*}_l \in t \mid \omega^{i*}_l \notin \mathcal{L}^{N_l}\right\}$; es decir, el conjunto de todas las secuencias de eventos con al menos un evento de fallo en el post-lenguaje de una secuencia de eventos normal.

6.3. Método de Diagnóstico

El método de diagnóstico propuesto parte sin conocimiento previo del sistema, ni en comportamiento normal, ni en comportamiento de fallo. Para resolver el problema se propone construir un diagnosticador a partir del comportamiento libre de fallo identificado con el Algoritmo 5.4. El resultado es un conjunto de st-IPN que generan el lenguaje normal observable. $\mathcal{L}^{N_l}$ es el lenguaje generado por la st-IPN identificada para el subsistema l, con $l := 1 \cdots c$, donde

c es el número de subsistemas. La estructura de cada modelo identificado es reformada con el propósito de que permita comparar el comportamiento libre de fallo y el comportamiento observado. Dada una traza de eventos t, si $t \notin \mathcal{L}^{N_l}$, $\forall l, l : 1 \cdots c$; entonces al menos un evento de fallo ha sido detectado, en este caso el algoritmo propuesto crea un modelo de lenguaje reconocedor para esta nueva situación y t es considerada como parte del lenguaje de fallo, $t = \mathcal{L}^{f_i}$.

Una vez un fallo ha sido detectado, un algoritmo de filtrado es ejecutado con el objeto de establecer si el fallo es real o es un falso fallo debido al efecto de propagación de fallos con base en el conjunto de sensores de flujo compartido SFS_r, definido previamente. Cuando un fallo ha sido detectado y no ha sido eliminado por el algoritmo de filtrado, la estructura del diagnosticador propuesta permite aislar el fallo, al mismo tiempo la nueva traza de eventos $\mathcal{L}^{f_i}$, es aprendida por el diagnosticador. Así, las habilidades de diagnóstico del diagnosticador crecen con el tiempo.

Para completar la tarea de diagnóstico se hace necesario identificar la naturaleza del fallo, para ello el algoritmo de diagnóstico genera una caracterización del fallo con la información suficiente para que sea remitida a un experto, quien determinará la criticidad, el tamaño y la importancia del fallo y tomará las medidas adecuadas para su manejo. En este instante se considera que el fallo ha sido diagnosticado.

Con el objeto de explicar el método de diagnóstico propuesto a continuación se explica en detalle la arquitectura del diagnosticador, el modelo del diagnosticador, el proceso de diagnóstico así como el algoritmo de diagnóstico.

6.3.1. Arquitectura para el Método de Diagnóstico

Un conjunto de st-IPNs permiten construir un generador del lenguaje en comportamiento normal para un sistema; pero como un fallo cambia las lecturas esperadas de los sensores o puede bloquear el controlador, se hace necesaria una estructura que no solo genere el comportamiento normal, sino que también detecte y aísle el fallo. Este proceso se implementa coloreando cada red identificada anteriormente como una st-ICPN, con el objeto que realice la comparación de lenguajes y detecte fallos.

El diagnosticador es una *"Red de Petri Coloreada Interpretada temporizada estocástica para Diagnóstico"* (st-DICPN), un ejemplo de su estructura se muestra en la Figura 6.3.1. Esta red tiene una estructura variable, ya que en el proceso de diagnóstico aprende lenguajes de fallos y se incrementan los vectores de marcas de las transiciones y de los arcos.

El conjunto de clases de color está compuesto por: $C_l = \{Comportamiento, VF, IF, Modo\}$; donde cada clase color está compuesta por un conjunto de marcas así: $Comportamiento = \{\langle N_l \rangle, \langle (l,q)\, gf_{gf_l} \rangle\}$, $\langle N_l \rangle$ es una marca normal, $\langle (l,q)\, gf_{gf_l} \rangle$ son marcas de fallo genérico en el subsistema l, q identifica el lugar y el subíndice gf_l es el identificador del fallo. La clase color VF (verificación de fallo), está compuesta por: $VF = \{\langle (l,q)\, gf_{gf_l} \rangle\}$. La clase color IF (identificación de fallo), está compuesta por: $IF = \{\langle entero \rangle\}$. $Modo$ es la clase de color asignada a las transiciones, donde $Modo = \{m\_N, m\_f_{df}\}$, m_N "modo normal", m_f_{df} "modo fallo df".

El conjunto de lugares es dividido en: $P_l = P_{LN_l} \cup P_{VF_l} \cup P_{IF_l}$. P_{LN_l} es el conjunto de lugares de anidamiento latente que representan lugares con comportamiento normal, pero es posible que puedan suceder fallos. P_{VF_l} es el conjunto de lugares de verificación de fallo

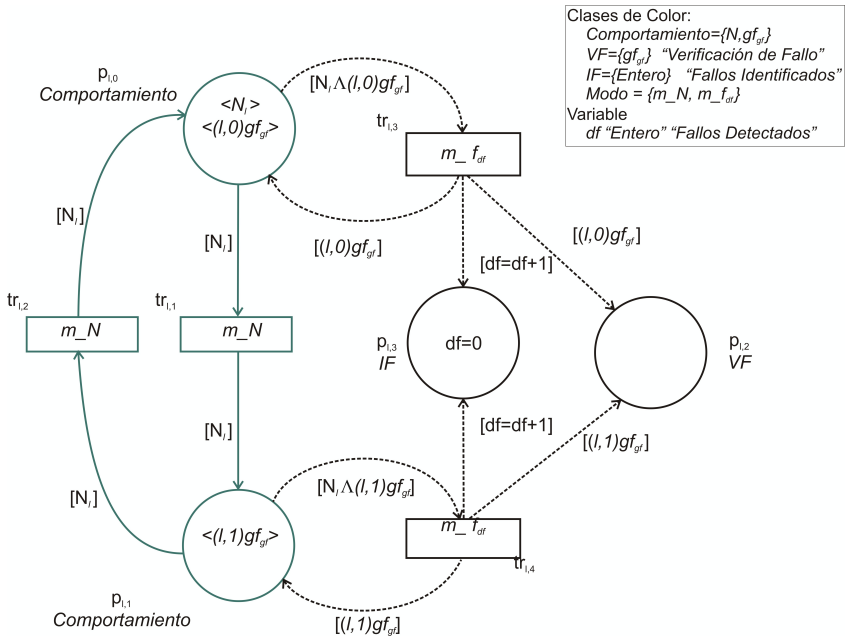


Figure 6.3.1: Arquitectura de una st-DICPN

en los cuales se acumulan marcas de fallo que permiten analizar los fallos detectados. P_{IF_l} es un lugar que contabiliza el número de fallos detectados.

El conjunto de transiciones es divido en: $TR_l = TR_{N_l} \cup TR_{F_l}$. TR_{N_l} es el conjunto de transiciones que se habilitan y disparan con eventos normales. TR_{F_l} corresponde a las transiciones que se habilitan cada vez que un fallo, f_{df}, es detectado; su disparo alcanza marcas de fallo en P_{VF_l} y marcas de tipo "$\langle entero \rangle$" en P_{IF_l}. (f_{df} es el identificador del fallo).

En la Figura 6.3.1, la parte de la red en color verde representa el comportamiento normal, el cual ha sido identificado on-line aplicando el Algoritmo 5.4 a partir de secuencias observadas, sus lugares se consideran de anidamiento latente de fallo. La parte de la red en color negro representa el comportamiento identificado de fallo, aplicando on-line, el algoritmo que se describe más adelante. Los arcos con líneas discontinuas negras se transforman en líneas continuas cuando un fallo es detectado en un lugar P_{LN_l} y la respectiva transición es disparada en modo de fallo f_{df}.

En la Figura 6.3.2 se nota como la estructura cambia cuando se detecta un fallo, ya que el vector de la $tr_{l,3}$ tiene dos marcas, una para el fallo detectado y otra para posibles fallos que se detecten, de igual manera sucede con los arcos que dependen de $tr_{l,3}$.

Las entradas de las matrices de incidencia son representadas por vectores, [25]; las matrices *Pre* y *Post* para la red de la Figura 6.3.2 se observan en la Tabla 6.3.1:

6.3.2. Modelo de Diagnosticador (st-DICPN)

La st-ICPN presentada en la Definición 33, es la base para el modelo de diagnóstico. Para cumplir con el requisito de tener estructura

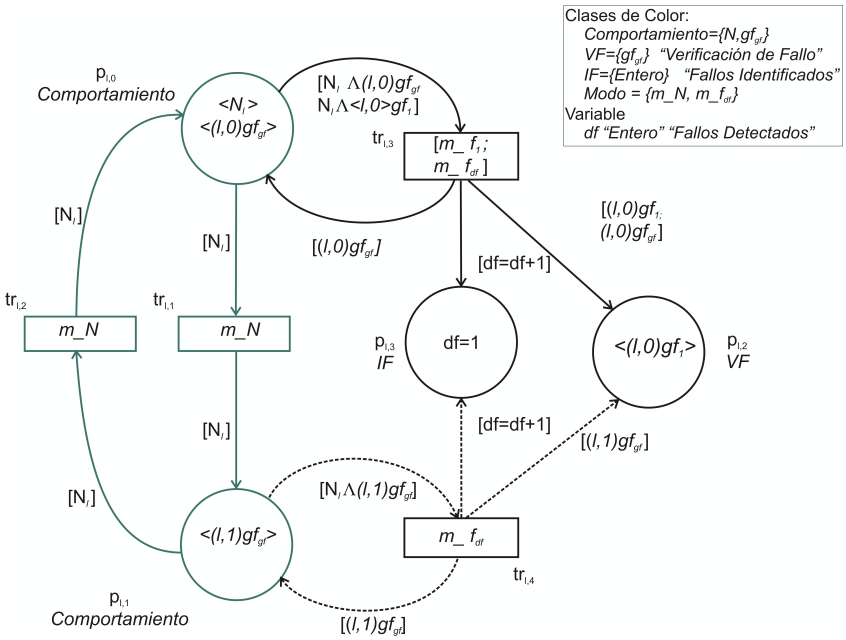


Figura 6.3.2: Arquitectura de una st-DICPN con un fallo.

Tabla 6.3.1: Ejemplo de matrices de incidencia en una arquitectura, st-DICPN

(a) Matriz *Pre*

	$tr_{l,1}$	$tr_{l,2}$	$tr_{l,3}$	$tr_{l,4}$
	$[m\_N]$	$[m\_N]$	$\begin{bmatrix} m\_f_{df} \\ m\_f_1 \end{bmatrix}$	$[m\_f_{df}]$
$p_{l,0}$	N_l	0	$\begin{bmatrix} N_l \wedge (l,0)\, gf_1 \\ N_l \wedge (l,0)\, gf_{gf} \end{bmatrix}$	0
$p_{l,1}$	0	N_l	0	$N_l \wedge (l,1)\, gf_{gf}$
$p_{l,2}$	0	0	0	0
$p_{l,3}$	0	0	0	0

(b) Matriz *Post*

	$tr_{l,1}$	$tr_{l,2}$	$tr_{l,3}$	$tr_{l,4}$
	$[m\_N]$	$[m\_N]$	$\begin{bmatrix} m\_f_{df} \\ m\_f_1 \end{bmatrix}$	$[m\_f_{df}]$
$p_{l,0}$	0	N_l	$N_l \wedge (l,0)\, gf_{gf}$	0
$p_{l,1}$	N_l	0	0	$N_l \wedge (l,1)\, gf_{gf}$
$p_{l,2}$	0	0	$N_l \wedge (l,0)\, gf_1$	$N_l \wedge (l,1)\, gf_{gf}$
$p_{l,3}$	0	0	1	0

variable para diagnosticar nuevos fallos, se adicionan características que se presentan a continuación.

Definición 38 (st-ICPN para Diagnóstico (st-DICPN))**.** Una st-DICPN para un subsistema l, se define como $stDCQ_l = (stCQ_l, fut)$; donde $stCQ_l = (CQ_l, \Omega_l, \delta_l)$ es una st-ICPN definida como en la Definición 33 para un subsistema l, con $P_l = P_{LN_l} \cup P_{VF_l} \cup P_{IF_l}$, $TR_l = TR_{N_l} \cup TR_{F_l}$ y $C_l = \{Comportamiento, VF, IF, Modo\}$. Los conjuntos de dominio de color de lugares son: $cd\left(P_{LN_l}\right) = \{N_l, (l,q)\, gf_{gf_l}\}$, $cd\left(P_{VF_l}\right) = \{(l,q)\, gf_{gf_l}\}$ y $cd\left(P_{IF_l}\right) = \{entero\}$. Los conjuntos de dominio de color de transiciones son: $cd\left(TR_{N_l}\right) = \{m\_N\}$ y $cd\left(TR_{F_l}\right) = \{m\_f_{df}\}$. La función de etiquetado de las transiciones normales es $\lambda_l : TR_{N_l} \times m\_N \rightarrow U_l.\delta_l$. $\delta_l := TR_{N_l} \times OM \rightarrow f\left(t_{TR_{N_l} \times OM}\right)$ es la función que asigna funciones de densidad de probabilidad del tiempo de disparo de las TR_{N_l} por cada OM. La función de salida de los P_{LN_l} es $\varphi_l : R(CQ_l) \times cd\left(P_{LN_l}\right) \rightarrow Y_l/dY_l$. Y fut es la función de actualización definida como $fut : M_l \times Y_l \rightarrow TR_l \times Pre_l \times Post_l$.

Las matrices de incidencia son: $Pre\left[p, tr\right] : cd\left(tr\right) \rightarrow cd\left(p\right)$ y $Post\left[p, tr\right] : cd\left(tr\right) \rightarrow cd\left(p\right)$. Es decir: $Pre\left[p^{LN}_{l,q}, tr^{N}_{l,r}\right] m\_N = \langle N_l \rangle$, $Pre\left[p^{LN}_{l,q}, tr^{F}_{l,r}\right] m\_f_{df} = \langle N_l \rangle \wedge \langle (l,q)\, gf_{gf_l} \rangle$, $Post\left[p^{LN}_{l,q}, tr^{N}_{l,r}\right] m\_N = \langle N_l \rangle$, $Post\left[p^{LN}_{l,q}, tr^{F}_{l,r}\right] m\_f_{df} = \langle (l,q)\, gf_{gf_l} \rangle$, $Post\left[p^{VF}_{l,q}, tr^{F}_{l,r}\right] m\_f_{df} = \langle (l,q)\, gf_{gf_l} \rangle$ y $Post\left[p^{IF}_{l,q}, tr^{F}_{l,r}\right] m\_f_{df} = \{x+1\}$. Esta estructura de matrices de incidencia permite la evolución de la red bajo condiciones normales o trazas de eventos no-predefinidos.

Una transición $tr_{l,j}\left(m\_N\right) \in TR_{N_l}$ con $\lambda(tr_{l,j}) = u^{N}_{l,s}.f\left(t_{r,o_{om},N}\right)$, está habilitada con respecto a m_N, en el modo de operación o_{om}; si $\forall p_{l,q} \in \bullet tr_{l,r}\left(m\_N\right)$, $m\left(p_{l,q}\right) \geq I\left(p_{l,q}, tr_{l,r}\left(m\_N\right)\right)$ y si $\left(a \leq t^{ev}_{l,o_{om},N} \leq b\right)$ siendo $\int_a^b f\left(t_{l,r,o_{om},N}\right) \geq (1-\alpha)$, donde $1-\alpha$ es el nivel de confiabilidad; $[a,b]$ el intervalo de confianza y $t^{ev}_{l,o_{om},N}$ es el

tiempo asociado al evento. Por lo tanto, una transición normal está habilitada sí cada lugar de entrada cumple con los requerimientos del color de su marca y si el tiempo transcurrido ha alcanzado un cierto nivel de confianza.

6.3.3. Proceso de Diagnóstico On-line

El proceso de diagnóstico propuesto en la presente investigación comprende diferentes etapas que se muestran en la Figura 6.3.3.

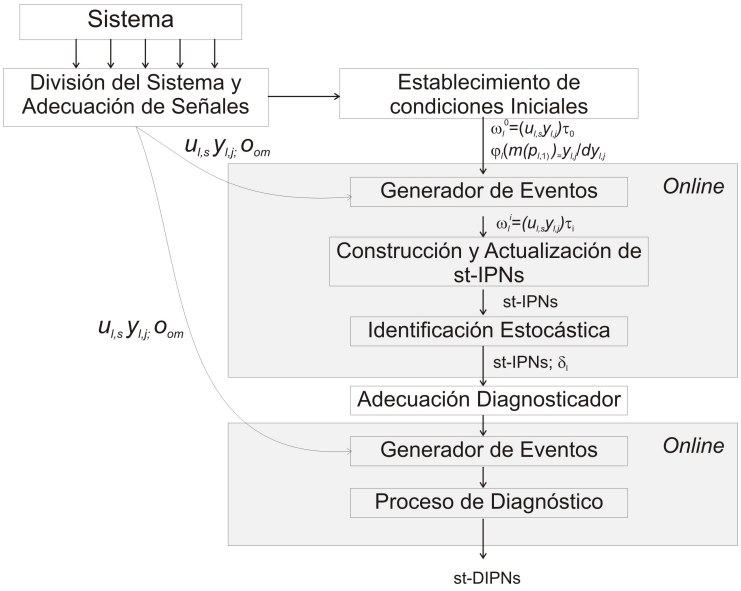


Figura 6.3.3: Proceso de diagnóstico

6.3.3.1. Etapa 1: Definición del sistema a Diagnosticar.

Una primera visión del sistema permite una aproximación del alcance del proceso de diagnóstico. Entonces dado el tamaño del

sistema y bajo criterios de expertos del proceso, un primer paso en esta etapa es la división del sistema en subsistemas con el objeto de reducir la complejidad del sistema y para que sea más fácil localizar en qué estado se ha producido un fallo.

Por otro lado, el sistema a diagnosticar es un sistema con comportamiento dinámico, estocástico del cual no se conoce el modelo; pero, se tiene acceso a señales que permiten inferir respecto al comportamiento del sistema. Por lo tanto, el segundo paso en la definición del sistema es determinar las señales a tener en cuenta en el proceso y clasificarlas en los siguientes grupos:

- Señales de entrada externas que afectan al controlador.
- Señales de entrada internas que van del controlador a la planta, discriminando si van a un solo subsistema o son globales.
- Señales de entrada externas que afectan el comportamiento de la planta.
- Señales de respuesta de la planta a las órdenes del controlador.
- Definición del conjunto de sensores compartidos $SRSr$

Una vez se han definido las señales, sí su lectura no es binaria, el tercer paso es la definición de rangos que permitan su representación binaria. Además organizarlas para que puedan ser leídas como en la secuencia seguida en la Ecuación 4.3.1, la Ecuación 4.3.2 y la Ecuación 4.1.1.

El siguiente paso es el cálculo del tiempo de estabilización de las lecturas sensoriales, con el objeto de evitar eventos erróneos. Si es posible acceder a este tipo de información.

6.3.3.2. Etapa 2: Identificación del Comportamiento Normal on-line.

Esta etapa se realiza monitoreando el sistema on-line cuando está trabajando en comportamiento normal, de una manera iterativa, hasta que todos los eventos se hayan identificado, bajo un criterio de detección estadístico. Para ello se deben cumplir los siguientes pasos:

1. **Establecimiento de las condiciones iniciales en cada subsistema**: Programación del modo de operación, o_{om} ($o_{om} \in OM$) o una secuencia de modos de operación. Evento inicial en cada subsistema l: $\omega_l^0 = (u_{l,s} y_{l,j})$ en τ_0, donde $u_{l,s}$, $y_{l,j}$ se asignan de acuerdo a la Definición 19 en el instante τ_0.
2. **Generación de eventos**: Al ejecutar el Algoritmo 5.1 se obtiene un evento de la forma $\omega_l^i = (u_{l,s} y_{l,j})\, .t_{l,o_{om}}^{ev}$.
3. **Construcción de las st-IPN**: El Algoritmo 5.4 construye con cada nuevo evento las estructuras de las st-IPN para cada subsistema, que representan las secuencias de eventos observados en comportamiento normal.
4. **Identificación Completa**: Los pasos 2 - 3 se repiten hasta que todas las transiciones de todos los subsistemas, en todos los modos de operación programados, hayan cumplido con el criterio de: $d \leq d_{op}$ donde d_{op} es la máxima diferencia relativa a aceptar entre la media de la población y la media muestral y d se calcula a partir de la Ecuación 5.2.1.
5. **Resultado**: Un conjunto de st-IPN que generan el lenguaje normal observado de cada subsistema, es decir: las matrices Pre y $Post$, las funciones de entrada $\lambda_l\,(tr_{l,r})$, las funciones de salida $\varphi\,(m\,(p_{l,q}))$, las funciones de densidad de probabilidad de las transiciones δ_l y el marcado alcanzado con el último evento M_k.

6.3.3.3. Etapa 3: Adecuación de la Arquitectura del Diagnosticador.

Para realizar la transformación de la estructura st-IPN a la estructura st-DICPN; la st-IPN es modificada de la siguiente manera:

- El marcado M_k se multiplica por $\langle N_l \rangle$ y se obtiene el marcado inicial para cada st-DICPN en cada subsistema.
- Se añade el conjunto de lugares de verificación de fallo y de identificación de fallo, P_{VF_l} y P_{IF_l} y las TR_{F_l} a la estructura inicial, así como los arcos con líneas punteadas como se muestra en la Figura 6.3.1.
- Para representar los fallos genéricos, una marca de color definida como $\langle (l,q)\, gf_{gf_l} \rangle$ se debe adicionar en cada lugar de anidamiento latente, P_{LN_l}.

6.3.3.4. Etapa 4: Monitorización del proceso on-line.

El objetivo de esta etapa es obtener una secuencia de eventos observados on-line, con el objeto de detectar y aislar los fallos. Se pretende que el proceso se realice de manera continua al proceso de identificación del comportamiento libre de fallo. Es decir, continuar a partir de la última iteración del método de identificación y de esta manera iniciar el diagnóstico a partir del marcado final M_k, en un estado normal del proceso. El proceso consiste en:

1. Se inicializan las variables $df_l = 0$, $gf_l = 0$ y el conjunto de trazas de fallo identificadas, $\mathcal{L}^{f_{df_l}} = \{\}$.
2. Dado un evento observado ω_l^{i*}, con $\omega_l^{i*} = \left(u_{l,s}^{i*} y_{l,j}^{i*}\right).t_{l,o_{om}}^{ev}$ en el instante τ_{k+1} en un $p_{l,q} \in P_{LN_l}$, a partir del cual:
   - Si $Pc_{\Omega_{N_l}}(\omega_l^i) = \omega_l^i$ (ver Definición 6), entonces $\omega_l^i \in \mathcal{L}^{N_l}$.

Nota 1: Como las st-IPNs identificadas representan solo el lenguaje normal, para comprobar si $\omega_l^{i*} \in \mathcal{L}^{N_l}$; se compara $u_{l,s}^{i*}.t_{l,o_{om}}^{ev}$ con las etiquetas de las transiciones habilitadas desde $p_{l,q}$: si $\exists\lambda\left(tr_{l,r}\right) \mid \left(u_{l,s} = u_{l,s}^{i*}\right) \wedge t_{l,o_{om}}^{ev} \in [a,b]$[2], entonces existe al menos una $tr_{l,r}$ habilitada; luego se compara $y_{l,j}^{i*}/dy_{l,j}^{i*}$ con los marcados alcanzados por las $tr_{l,r}$ habilitadas, si $\exists\varphi_l\left(m\left(p_{l,q}\right)\right) \mid y_{l,j}^{i}/dy_{l,j}^{i} = y_{l,j}^{i*}/dy_{l,j}^{i*}$ entonces al menos una puede ser disparada en modo normal. Por lo tanto, sí se cumplen las dos condiciones, $tr_{l,r}\left(m\_N\right) \in TR_{N_l}$ es disparada en modo normal con $\lambda_l\left(tr_{l,r}\right) = u_{l,s}.f\left(t_{l,r,o_{om}}\right)$ y una marca normal se coloca en $m\left(p_{l,q+1}\right) = \langle N_l\rangle$, con $\varphi_l\left(m\left(p_{l,q+1}\right)\right) = y_{l.j}/dy_{l.j}$.

- De lo contrario si $Pc_{\Omega_{N_l}}\left(\omega_l^i\right) = \varepsilon$ entonces $\omega_l^i \notin \mathcal{L}^{N_l}$, es decir un fallo ha sido detectado.

  Nota 2: $\omega_l^i \notin \mathcal{L}^{N_l}$ sí al menos una de las dos condiciones de la Nota 1 no se cumplen. Entonces una marca de fallo ha sido detectada en l; $M_o\left[\sigma^{F_{sub}} > M_{k+1}\right.$, donde $\sigma^{F_{sub}}$ genera una traza de eventos t que finalizan en un fallo; $df_l = df_l + 1$.

- Una nueva marca en una transición $tr_{l,r} \in TR_{F_l}$ es generada (si no existe) y la $tr_{l,r}$ es disparada en modo $m\_f_{df_l}$ $\left(tr_{l,r}\left(m\_f_{df_l}\right)\right)$ y $gf_l = gf_l + 1$. Una marca de fallo genérico se coloca en $m\left(p_{l,VF_l}\right) = m\left(p_{l,q_l}\right) = \langle(l,q)\, gf_{gf_l}\rangle$ y otra marca de tipo entero en $m\left(p_{l,IF_l}\right) = \langle df_l\rangle$.

Este proceso se observa en el Algoritmo 6.1.

3. Cuando un fallo es detectado y aislado, el siguiente paso es determinar si se debe eliminar por falsas alarmas causadas

[2] $[a.b]$es el intervalo de confianza del tiempo de la transición $tr_{l,r}$ en modo de operación o_{om}.

Algoritmo 6.1 Proceso de detección y aislamiento

Entradas: $\mathcal{L}^{N_l}$; $\omega_l^{i*} = \left(u_{l,s}^* y_{l,j}^*\right).t_{l,o_{om}}^{ev}$; $M_{l,k}$; Pre_l; $Post_l$; $\lambda_l\left(TR_l\right)$; $\varphi_l\left(M_k\right)$
Salidas: IF_l; $\mathcal{L}^{IF_l}$; $st - DICPN_l$ actualizadas.
Condiciones Iniciales: ; $IF_l = []$; $\mathcal{L}^{f_{df_l}} = \{\}$; $f\_in = f$; $f\_out = f$.
Esperar evento;
Ejecutar Algoritmo 5.1; leer $\omega_l^{i*} = \left(u_{l,s}^* y_{l,j}^*\right).t_{l,o_{om}}^{ev}$ en τ_i;
Leer $M_{l,k}$; leer $\varphi\left(m\left(p_{l,q}^*\right)\right)$ con $\langle N\rangle$; $sub = l$;
Calcular $tr_{habilitadas} = \left\{tr_{l,r} \mid Pre(p_{l,q}^*, tr_{l,r}) = \langle N\rangle\right\}$; $\lambda_l\left(tr_{habilitadas}\right) = \{\lambda_l\left(tr_{l,r}\right) \mid tr_{l,r} \in tr_{habilitadas}\}$;
Calcular $\varphi_l\left(M_{k,l}\right) \overset{tr_{habilitadas}}{\rightarrow} \varphi_l\left(M_{alcanzado}\right)$; $\varphi_l\left(M_{alcanzado}\right) = \{\varphi_l\left(m\left(p_{l,q}\right)\right) \mid Post(p_{l,q}, tr_{habilitadas}) = \langle N\rangle\}$;
Calcular $\lambda_{\omega_l^{i*}} = u_{l,s}^*.t_{o_{om}}^{ev}$; $\varphi_l\left(\omega_l^{i*}\right) = y_{l,j}^*/dy_{l,j}^*$.

a) Si $u_{l,s}^* \in \lambda_l\left(tr_{habilitadas}\right) \wedge t_{o_{om}}^{ev} \in [a,b]$ entonces, $f\_in = f$; de lo contrario $f\_in = v$.

b) Si $\varphi_l\left(\omega_l^{i*}\right) \in \varphi_l\left(M_{alcanzado}\right)$ entonces, $f\_out = f$; de lo contrario $f\_out = v$.

c) Si $f\_in = f \wedge f\_out = f$ entonces $\omega_l^{i*} = \left(u_{l,s}^* y_{l,j}^*\right).t_{om}^{ev} \in \mathcal{L}^{N_{sub}}$ y $m\left(p_{sub,q_l+1}\right) = \langle N_{sub}\rangle$; de lo contrario $\omega_l^{i*} = \left(u_{l,s}^* y_{l,j}^*\right).t_{om}^{ev}$ es un fallo genérico en $p_{l,q}^*$; $gf_{sub} = gf_{sub} + 1$; entonces una nueva marca en $tr_{l,r}$ es generada así:

1) $pre = zeros\left(size\left(Pre_l(1,:)\right)\right)$: vector de ceros excepto en $pre\left(p_{sub,q_{sub}}\right) = N_l \wedge (sub, q_{sub})\, gf_{gf_{sub}}$.
2) $post = zeros\left(size\left(Post_l(1,:)\right)\right)$: vector de ceros excepto en $post\left(p_{sub,q_{sub}}\right) = post\left(p_{p_{sub,VF_{sub}}}\right) = (sub, q_{sub})\, gf_{gf_{sub}}$ y en $post\left(p_{sub,IF}\right) = df_l$.
3) La nueva $tr_{l,r}$ se dispara alcanzado: $m\left(p_{sub,VF_{sub}}\right) = \langle (sub, q_{sub})\, pf_{pf_{sub}}\rangle$; $m\left(p_{sub,q_{sub}}\right) = (sub, q_{sub})\, gf_{gf_{sub}}$ y $df_l = df_l + 1$. $m\left(p_{l,IF_l}\right) = \langle df_l\rangle$.
4) $f_{df_{sub}} = f_{df_{sub}} + 1$.
5) Ejecutar Algoritmo 6.2.

Actualizar $M_{l,k}$; Pre_l; $Post_l$; $\mathcal{L}^{f_{df_l}} = \{\}$.
Esperar nuevo evento.

por lecturas de sensores afectadas por flujo compartido. Para ello se propone un algoritmo de filtrado de fallos.

El algoritmo trabaja con la información del conjunto de sensores que miden flujo, definida previamente: $SFS_r = \cup sr_{l,n_l}$, donde n_l es el número del sensor de flujo en el subsistema l. Dado un fallo identificado, $\omega_{l,q}^{i*}$ en el instante actual τ_{ac}, el $\mathcal{L}_{out}\left(\omega_{l,q}^{i*}\right) = y_{l,j}$ y $y_{l,j} = sr_{l,1} \cdots sr_{l,n_l}$. Si el fallo se debe a que $y_{l,j}$ no es la lectura esperada en el lenguaje normal y si la lectura del sensor fallida pertenece a SFS_r, se analiza el $\mathcal{L}_{out}\left(\omega_{l,q}^{i*}\right)$ con el conjunto de lecturas de sensor identificadas previamente en eventos de fallo $ISFS_r$ en todos los subsistemas y se determina si el fallo es de propagación. Este proceso se muestra el Algoritmo 6.2.

4. Finalmente, la función *fut* actualiza la st-DICPN. La arquitectura de la st-DICPN se actualiza de la siguiente manera:

   - Las marcas de fallo serán adicionadas en los lugares de verificación de fallos.
   - El marcado es actualizado.
   - Las marcas en las transiciones de fallo son adicionadas de acuerdo los vectores *pre* y *post* definidos en el proceso de detección del fallo.
   - La traza t es incluida en el conjunto de trazas de fallo identificadas, $t \subset \mathcal{L}^{f_{df_l}}$.

   De esta manera la traza de fallo es aprendida por la st-DICPN.

   El aislamiento del fallo es llevado a cabo, una vez la traza de fallo es conocida y aprendida. Con esta traza es posible distinguir el subsistema en el cual sucedió del fallo así como las señales fallidas.

Algoritmo 6.2 Algoritmo de Filtrado de Fallos

Entradas: SFS_r; $s\omega_{l,q}^{i*} \subset \mathcal{L}^{f_i}$; $s\omega_l^p \in \mathcal{L}^{N_l}$.
Salidas: $ISFS_r$.
Condiciones Iniciales: $ISFS_r = \{\}$.

a) Si $P_{\mathcal{L}_{out}^{N_l}}\left(s\omega_{l,q}^{i*}\right) = \mathcal{L}_{out}(s)$ entonces $\mathcal{L}_{out}\left(\omega_{l,q}^{i*}\right) \neq \mathcal{L}_{out}(\omega_l^p)$ y $\exists i \mid sr_{l,i}^* \in y_{l,j}^* \neq sr_{l,i} \in y_{l,j}$, por lo tanto $sr_{l,i} = sr_{l,i}^{ex}$ es la lectura esperada del sensor.

1) Si $sr_{l,i}^{ex} \in SFS_r$ entonces:

   para $l : 1 \cdots c$

   - Si $\exists sr_{l,n_l} \in ISFS_r$ entones, borrar $\omega_{l,q}^{i*}$ porque es un fallo de propagación.
   - de lo contrario, $sr_{l,i}^{ex} \subset ISFS_r$.
   - fin del condicional.

   fin ciclo.

2) fin del condicional.

b) de lo contrario no es fallo de propagación.

c) fin del condicional.

6.3.4. Aplicación a un Proceso Sencillo

Ejemplo 8 (Cilindros Neumáticos)**.** Sea un sistema compuesto por dos cilindros neumáticos, como el que se muestra en la Figura 4.5.1.

Etapa 1: Definición del sistema a Diagnosticar

Cada cilindro se considera un subsistema. Las señales se discriminan de la siguiente manera:

Señales de entrada externas: $EnExC = \{cil_A, cil_B\}$, son las señales que determinan el funcionamiento del cilindro A o del cilindro B

Señales de entrada internas a la planta de cada subsistema: $EnInP_A = \{LCc\} = \{cc_A\}$, $EnInP_B = \{LCc\} = \{cc_B\}$.

Señales de salida de cada subsistema: $Sr_A = \{s_{1,1}, s_{1,2}\}$; $Sr_B = \{s_{2,1}, s_{2,2}\}$.

$o_{om} = [cil_A cil_B]$. Si $cil_A = 1$ y $cil_B = 0$; $om = 2$. Si $cil_A = 0$ y $cil_B = 1$; $om = 1$. Si $cil_A = 1$ y $cil_B = 1$; $om = 3$. Si $cil_A = 0$ y $cil_B = 0$; $om = 0$.

$u_{1,s} = [cc_A]$ si $cc_A = 0$ entonces $s = 0$ de lo contrario $s = 1$. $u_{2,s} = [cc_B]$ si $cc_B = 0$ entonces $s = 0$ de lo contrario $s = 1$.

$y_{1,j} = [s_{1,1} s_{1,2}]$, donde $j = 0, 1, 2, 3$; $y_{2,j} = [s_{2,1} s_{2,2}]$, donde $j = 0, 1, 2, 3$.

Etapa 2: Identificación del Comportamiento Normal online.

El sistema está trabajando en modo de funcionamiento o_2.

1. El evento inicial es: $\omega_1^0 = (u_{1,0} y_{1,0}), 0$ $\omega_2^0 = (u_{2,0} y_{2,0}), 0$ y las funciones iniciales de salida son: $\varphi_1 (m (p_{1,0})) = y_{1,0}/00$; $\varphi_2 (m (p_{2,0})) = y_{2,0}/00$.
2. Se genera el evento $\omega_1^1 = (u_{1,1} y_{1,2}), 2, 3$ en el subsistema 1.

3. Se ejecuta el Algoritmo 5.4, se genera un nuevo lugar con $\varphi_1(m(p_{1,1})) = y_{1,2}/10$ y una transición con $\lambda(tr_{1,1}) = u_{1,1}$,2, 3 se adiciona.

4. Se repiten los pasos 2 y 3; por 13 veces hasta que $d \leq 0{,}05$ y todas las transiciones son identificadas totalmente.

El lenguaje en comportamiento normal identificado es: $\mathcal{L}^{N_A} = (u_{1,0}y_{1,0}) \cdot \left(t^0_{1,o_2}\right)(u_{1,1}y_{1,2}) \cdot \left(t^1_{1,o_2}\right)(u_{1,1}y_{1,3}) \cdot \left(t^2_{1,o_2}\right)(u_{1,0}y_{1,2}) \cdot \left(t^3_{1,o_2}\right)(u_{1,0}y_{1,0}) \cdot \left(t^4_{1,o_2}\right)(u_{1,1}y_{1,2}) \cdot \left(t^5_{1,o_2}\right)$.

$\varphi_1(m(p_{1,0})) = y_{1,0}/00$, $\varphi_1(m(p_{1,1})) = y_{1,2}/10$, $\varphi_1(m(p_{1,2})) = y_{1,3}/01$, $\varphi_1(m(p_{1,3})) = y_{1,2}/0-1$, $\varphi_1(m(p_{1,4})) = y_{1,0}/-10$, $\varphi_1(m(p_{1,1})) = y_{1,2}/10$.

$\lambda_1(tr_{1,1}) = u_{1,0}.f(t_{1,1,o_2})$, $\lambda_1(tr_{1,2}) = u_{1,1}.f(t_{1,2,o_2})$, $\lambda_1(tr_{1,3}) = u_{1,1}.f(t_{1,3,o_2})$, $\lambda_1(tr_{1,4}) = u_{1,0}.f(t_{1,,4,o_2})$, $\lambda_1(tr_{1,5}) = u_{1,1}.f(t_{1,5,o_2})$.

El marcado final es $M_K = [01000]$

La st-IPN resultante se muestra en la Figura 6.3.4. (En modo de operación o_2 solo funciona el cilindro A).

En la Tabla 6.3.2, se presentan las funciones de densidad de probabilidad para el ejemplo.

Tabla 6.3.2: Funciones de densidad de probabilidad de la st-IPN de la Figura 6.3.4

$tr_{l,r}$	o_{om}	Intervalo de Confianza (min)
$tr_{1,1}$	o_2	$[2-2]^*$
$tr_{1,2}$	o_2	$[3,8-4,3]$
$tr_{1,3}$	o_2	$[3,8-4,3]$
$tr_{1,4}$	o_2	$[2,1-2,6]$
$tr_{1,5}$	o_2	$[1-1]^*$

* **Son tiempos determinísticos**

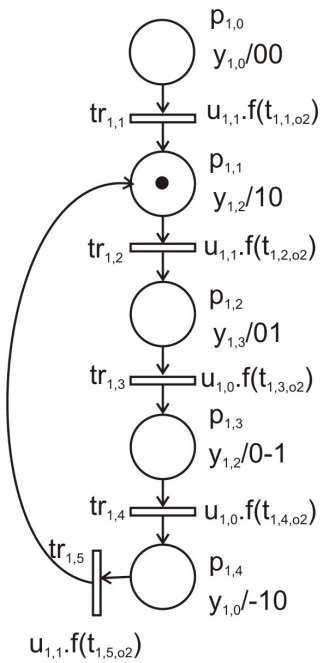


Figura 6.3.4: st-IPN Cilindro A. Comportamiento Normal

Etapa 3: Adecuación de la Arquitectura del Diagnosticador.

Añadiendo la estructura para diagnóstico de fallo, la st-DICPN se muestra en la Figura 6.3.5.

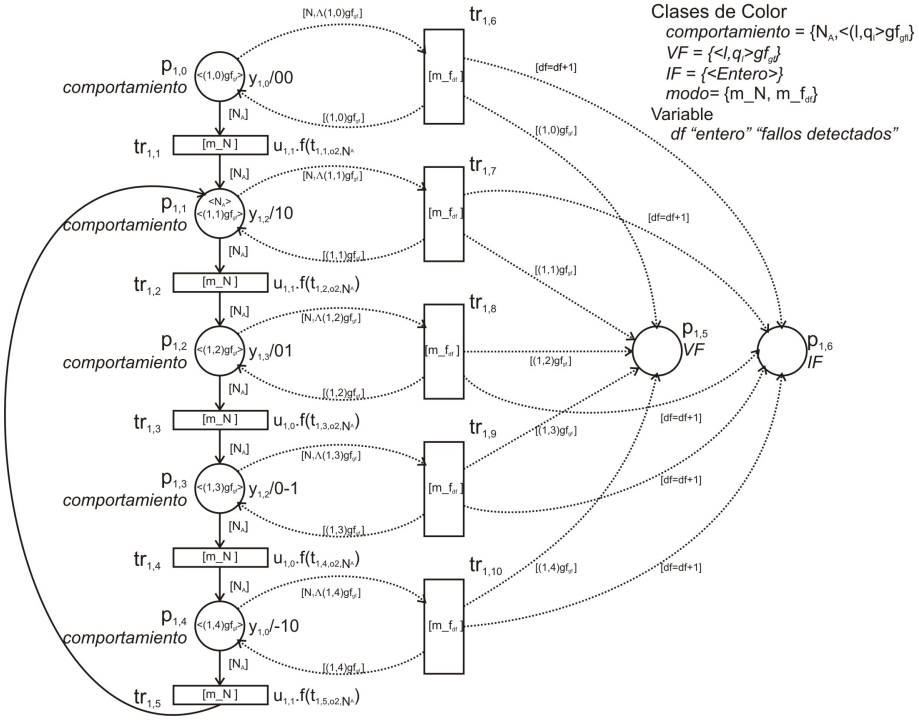


Figura 6.3.5: st-DICPN para diagnóstico de un cilindro neumático

Etapa 4: Monitorización del proceso on-line.

El marcado inicial es $M_K = [0\,\langle N_A\rangle\,000]$; por lo tanto el lugar inicial es $p_{1,2}$.

$df_1 = 0$, $gf_1 = 0$, $\mathcal{L}^{f_{df_1}} = \{\}$

1. Dado un evento $\omega_1^{1*} = \left(u_{1,1}^* y_{1,3}^*\right)$,3, 9. Siendo el estado actual $p_{1,1}$, la transición que se habilita es $tr_{1,2}$; $\lambda_1\left(tr_{1,2}\right) = u_{1,1}.f\left(t_{1,2,o_2,N_A}\right)$, entonces $u_{1,1}^* = u_{1,1}$ y $3,9 \in [3,8-4,3]$. $\varphi_l\left(m\left(p_{1,q_l}\right)\right) = y_{1,3}/01$ y $y_{1,3}^*/dy_{1,3}^* = y_{1,3}^*/01$; por lo tanto $tr_{1,2}\left(m\_N\right)$ se dispara en modo normal y en o_2; entonces una marca normal es alcanzada en $m\left(p_{1,q_l}\right) = \langle N_A \rangle$ con $\varphi_1 m\left(p_{1,1}\right) = y_{1,3}/01$.

2. Dado un evento $\omega_1^{2*} = \left(u_{1,0}^* y_{1,3}^*\right)$,4, 2. Siendo el estado actual $p_{1,2}$, la transición que se habilita es $tr_{1,3}$ con $\lambda_1\left(tr_{1,3}\right) = u_{1,0}.f\left(t_{1,3,o_2,N_A}\right)$, entonces $u_{1,0}^* = u_{1,0}$ y $4,2 \in [3,8-4,3]$. $\varphi_l\left(m\left(p_{1,q_l}\right)\right) = y_{1,2}/0{-}1$ y $y_{1,3}^*/dy_{1,3}^* = y_{1,3}^*/00 \neq \varphi_l\left(m\left(p_{1,q_l}\right)\right)$; por lo tanto se ha detectado un fallo; $df_1 = 1$, $gf_1 = 1$, $tr_{1,8}\left(m\_f_1\right)$ se dispara en modo de fallo y una marca de fallo es alcanzada en $m\left(p_{1,5}\right), m\left(p_{1,2}\right) = \langle (1,2)\, gf_1 \rangle$ y $m\left(p_{1,6}\right) = \langle 1 \rangle$.

La secuencia de disparo $M_{l,0}\lceil\sigma_i > M_{l,k}$ es una secuencia de fallo, donde $\sigma^F = tr_{1,1}tr_{1,2}tr_{1,8}$, la cual genera una traza de fallo $t = \left(u_{1,0}y_{1,0}\right)\left(u_{1,1}y_{1,2}\right)\left(u_{1,1}y_{1,3}\right)\left(u_{1,0}^* y_{1,3}^*\right)$; esta traza se adiciona al lenguaje de fallo del subsistema 1, $\mathcal{L}^{f_1} = \left\{\left(u_{1,0}y_{1,0}\right)\left(u_{1,1}y_{1,2}\right)\left(u_{1,1}y_{1,3}\right)\left(u_{1,0}^* y_{1,3}^*\right)\right\}$.

Al actualizar la st-DICPN con los resultados obtenidos, el diagnosticador cambia su estructura, la cual se muestra en la Figura 6.3.6.

La st-DICPN no se bloquea al detectar el fallo, en este instante la red está en el estado $p_{1,2}$, a partir del cual puede continuar con el proceso de diagnóstico; si las condiciones de la planta lo permiten.

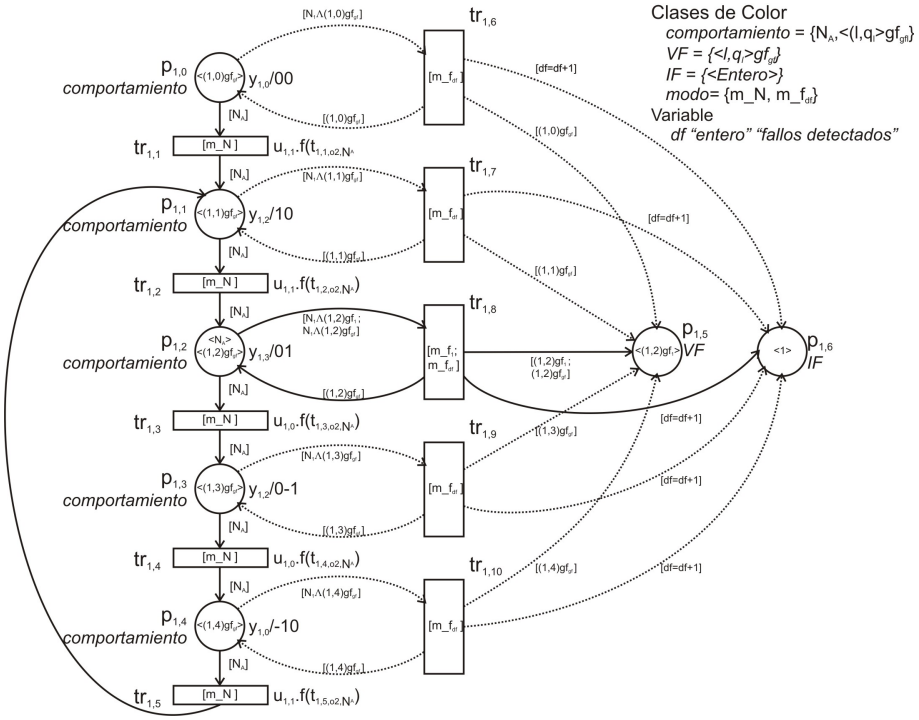


Figura 6.3.6: st-DICPN del cilindro con fallo f_1

6.4. Propiedades del Diagnosticador

La estructura de una red, $stDCQ_l$ es variable porque el vector de entrada de las matrices de incidencia aumenta cada vez que un fallo es detectado y aislado, así como el número de marcas o modos en las transiciones.

El tamaño máximo de cada $stDCQ_l$, está limitado por el número de sensores, ya que el número máximo de fallos que pueden ser detectados se calcula como: $|f_{df}| = 2^{n_{sr}}$, donde n_{sr} es el número de sensores.

Sea σ^{N_l} una secuencia de disparo de transiciones tal que, $\forall tr_{l,r} \in \sigma^{N_l}, tr_{l,r} \in TR_{N_l}$ y sea s la secuencia de eventos generada por σ^{N_l} tal que $s \in \mathcal{L}^{N_l}(stDCQ_l)$, donde $s = \omega_l^0, \cdots \omega_l^k$. Si $\varphi(m(p_{l,0})) = \varphi(m(p_{l,k}))$ entonces $s^* \in \mathcal{L}^{N_l}(stDCQ)$, es decir, s es un ciclo.

Dada una secuencia de eventos $s\omega_l^{i*}$, tal que $s \in \mathcal{L}^{N_l}$ con $s = \omega_l^0, \cdots, \omega_l^k$ a instantes $\tau_1 \leq \cdots \leq \tau_k$ y si $s\omega_{l,q}^{i*} \in \mathcal{L}^{f_i}$, la secuencia de disparo de transiciones σ^{f_i} que genera $s\omega_l^{i*}$ es $\sigma^{f_i} = tr_{l,1}, \cdots tr_{l,k}, tr_{l,k+1}, tr_{l,k}$, donde $tr_{l,1}, \cdots tr_{l,k} \in TR_{N_l}, tr_{l,k+1} \in TR_{F_l}$ y $M_{l,0}\left[\sigma^{f_i} > M_{l,k}\right.$. La marca alcanzada en p_{l,VF_l} es $m(p_{l,VF_l}) = \langle (l,q)\, gf_{gf_l} \rangle$ y la marca alcanzada en $p_{l,k} \in P_{LN_l}$, $m(p_{l,k}) = \langle N \rangle, \langle (l,q)\, gf_{gf_l} \rangle$. Es decir $\mathcal{L}^{f_i}(stDCQ_L)$ es vivo. $\mathcal{L}^{f_i}(stDCQ) = \omega_l^0, \cdots, \omega_l^k \omega_{l,q}^{i*} \omega_l^{k+1}$.

6.4.1. Detectabilidad

El análisis de detectabilidad se realiza con base en el enfoque de lenguajes y se relaciona con el concepto de observabilidad presentado en el capítulo de modelado.

La detectabilidad prueba si un sistema o lenguaje puede detectar la ocurrencia de un fallo en un número finito de eventos observados.

Basado en la Definición 36, a continuación se define la *n-detectabilidad*.

Definición 39 (n-Detectable)**.** Sea s una secuencia de eventos que finaliza en un evento de fallo y sea t una secuencia de eventos observables después de $s\omega_l^{i*}$, con $n = |t|$. Dada $s_1 = s\omega_l^{i*}t$, el evento $\omega_l^{i*} = \left(u_{l,s}^* y_{l,j}^*\right).t_{l,o_{om}}^{ev}$ es n-detectable si $\exists\omega_l^i \in t$, tal que: *i)* $Pc_{\Omega_{N_l}}(u_{l,s}) \notin \mathcal{L}_{in}^{N_l} \wedge Pc_{\Omega_{N_l}}(y_{l,j}) \in \mathcal{L}_{out}^{N_l}$, o *ii)* $Pc_{\Omega_{N_l}}(u_{l,s}) \in \mathcal{L}_{in}^{N_l} \wedge Pc_{\Omega_{N_l}}(y_{l,j}) \notin \mathcal{L}_{out}^{N_l}$, o *iii)* $Pc_{\Omega_{N_l}}(u_{l,s}) \in \mathcal{L}_{in}^{N_l} \wedge Pc_{\Omega_{N_l}}(y_{l,j}) \in \mathcal{L}_{out}^{N_l}$ pero $t_{l,o_{om}}^{ev} \notin [a,b]$[3] en n pasos.

6.4.1.1. Condición Necesaria y Suficiente para Detectabilidad

Un evento de fallo es detectable cuando después de su ocurrencia, existe al menos una señal que no pertenece al comportamiento normal del sistema. Esto significa que una vez ocurra un fallo, éste es detectado sí después de un número finito de pasos al menos un símbolo de entrada o uno de salida o el tiempo, no es el esperado (condición necesaria).

Teorema 3 (Detectabilidad)**.** *Dada una secuencia de eventos que finaliza en un evento de fallo, $s\omega_l^p$, y sean t_1, t_2 dos secuencias de eventos observables, donde $s, t_1 \in \mathcal{L}^{N_l} \wedge t_1, t_2 \in \mathcal{L}/s$; ω_l^p es detectable sii $Pt_{\Omega_{N_l}}(st_1) \neq Pt_{\Omega_{N_l}}(s\omega_l^p t_2)$.*

Demostración. Sea $s = \omega_l^0, \cdots, \omega_l^k$ y $t_2 = \omega_l^q, \cdots, \omega_l^x$. Como $t_1 \in \mathcal{L}^{N_l}/s \wedge t_1 \in \mathcal{L}^{N_l}$ entonces $st_1 \in \mathcal{L}^{N_l}$. Y como ω_l^p es un evento de fallo entonces $s\omega_l^p t_2$ es una traza de fallo de la $stDCQ_l$, ($\mathcal{L}^{f_i} = s\omega_l^p t_2$).

[3] $[a.b]$ es el intervalo de confianza del tiempo de la transición $tr_{l,r}$ en modo de operación o_{om}.

Condición Necesaria: si ω_l^p es detectable entonce $Pt_{\Omega_{N_l}}(st_1) \neq Pt_{\Omega_{N_l}}(s\omega_l^p t_2)$.

Como $t_1 \in \mathcal{L}^{N_l}$ entonces $\forall \omega_l^i \in t_1, \omega_l^i \in \Omega_{N_l}$. Si ω_l^p es detectable entonce $\exists \omega_l^{i*} \in t_2 \mid \omega_l^{i*} \in \Omega_{o_l} \wedge \omega_l^{i*} \notin \Omega_{N_l}$ (ver Definición 39) por lo tanto $t_1 \neq t_2$ y $Pt_{\Omega_{N_l}}(st_1) \neq Pt_{\Omega_{N_l}}(s\omega_l^p t_2)$.

Condición Suficiente: si $Pt_{\Omega_{N_l}}(st_1) \neq Pt_{\Omega_{N_l}}(s\omega_l^p t_2)$ entonces ω_l^p es detectable.

$stDCQ_l$ representa solo lenguajes observables. Si st_1 y $s\omega_l^p t_2$ están representados en una $stDCQ_l$ entonces st_1 y st_2 deben ser secuencia de eventos observados.

Asumiendo que $Pt_{\Omega_{N_l}}(st_1) = Pt_{\Omega_{N_l}}(s\omega_l^p t_2)$ entonces $t_2 \in \mathcal{L}^{N_l}$ y ω_l^p es no detectable. Por lo tanto queda demostrado. □

6.4.2. Caracterización del Fallo

Como la presente propuesta de diagnóstico no presenta modelo de fallo del sistema, es decir, no existe un conocimiento previo de los fallos que puedan suceder en cuanto a nivel de criticidad, tamaño, etc. La identificación del fallo la realiza un experto en el sistema, a partir de la información de la caracterización del fallo, la cual básicamente resume los resultados del proceso de detección y localización del fallo.

La información es reportada como se indica en la Tabla 6.4.1.

Esta información al ser analizada por el experto, se discrimina el fallo en cuanto a criticidad, tamaño e importancia y así completar la tarea de diagnóstico del fallo.

Para el ejemplo 8, la caracterización del fallo se muestra en la Tabla 6.4.2.

En esta tabla se puede observar que cuando el controlador emite la orden de extender, el cilindro no llega a la posición de fin de carrera,

Tabla 6.4.1: Caracterización del fallo

Fallo	Marca de fallo.
Instante	Tiempo en el cual ocurre.
Subsistema	l
Lugar	$p_{l,q}$ donde ocurre el fallo.
Señal esperada	Qué lecturas se esperaban
Señal observada	Qué lecturas se midieron en ese instante.
Tipo de Fallo	Indica si el fallo es por tiempo, por error en las órdenes de control o error por lecturas de sensores.
Señal de fallo $s_{l,i}$	Sensor con lectura de fallo.
$ISFS_r$	Verifica si es un sensor de flujo compartido.
Eliminado?	Indica si se eliminó en el algoritmo de filtrado o no.

Tabla 6.4.2: Caracterización del fallo en el ejemplo 8

Fallo	$\langle (1,2)\, gf_1 \rangle$
Instante	8.1
Subsistema	1
Lugar	$p_{1,2}$
Señal esperada	$y_{1,3}/00$
Señal observada	$y_{1,2}/0-1$
Tipo de Fallo	Fallo por lectura de sensor.
$s_{l,i}$	$s_{1,2}$
$ISFS_r$	No es sensor de flujo compartido.
Eliminado?	No.

por lo tanto puede existir un fallo en la alimentación de aire del cilindro o existe un fallo del sensor. Un fallo en la alimentación de aire del cilindro se considera un fallo grave del sistema, puesto que implica que el funcionamiento del sistema no se genere y el cilindro entre a un estado de bloqueo, si es un fallo del sensor no se considera como fallo grave. El fallo debe ser verificado y reparado para que el sistema pueda recobrar su funcionamiento normal. De esta manera se ha completado la tarea de diagnóstico y se ha empezado a construir un modelo en comportamiento de fallo del sistema.

6.5. Aplicación al Proceso de Calefacción de la Sección 5.6

En esta sección se va a considerar el sistema AHS, identificado en la Sección 5.6, para el escenario 2 (ver figura Figura 5.6.4). Las condiciones iniciales en cada subsistema son:

$\omega_1^0 = \left[\overline{v}_r\overline{p}_g\overline{v}_g\overline{h}_1\overline{h}_2, T_o\overline{T}_1\overline{T}_2\overline{F}_1\right] = (u_{1,0}y_{1,8})$; $\omega_2^0 = \left[\overline{v}_{c1}\overline{v}_{c2}, \overline{P}_o\overline{F}_2\overline{NF}_2\right]$ $= (u_{2,0}y_{2,0})$; $\omega_3^0 = \left[\overline{p}_3\overline{v}_3, \overline{F}_3\overline{NF}_3\right] = (u_{3,0}y_{3,0})$; $\omega_4^0 = \left[\overline{p}_4\overline{v}_4, \overline{F}_4\overline{NF}_4\right]$ $= (u_{4,0}y_{4,0})$; $\omega_5^0 = \left[\overline{p}_5\overline{v}_5, \overline{F}_5\overline{NF}_5\right] = (u_{5,0}y_{5,0})$.

Conjunto de sensores de flujo: $SFS_r = \{sr_{1,4}, sr_{2,1}, sr_{2,2}, sr_{3,1}, sr_{3,2}, sr_{4,1}, sr_{4,2}, sr_{5,1}, sr_{5,2}\}$.

6.5.1. Detección y Localización

Paso 1

Los lenguajes normales identificados en cada subsistema son:

$\mathcal{L}^{N_1} = (u_{1,0}y_{1,8})\,(u_{1,30}y_{1,9})\ (u_{1,30}y_{1,13})\ (u_{1,15}y_{1,15})\ (u_{1,15}y_{1,13})\ (u_{1,30}y_{1,13})\ (u_{1,30}y_{1,9})\ (u_{1,0}y_{1,9})\ (u_{1,0}y_{1,8})\ (u_{1,30}y_{1,9})$ a instantes $\tau_0, \tau_1, \tau_2,\ \tau_4,\ \tau_{23}, \tau_{24}, \tau_{25}, \tau_{26}, \tau_{23}, \tau_{29}$.

$\mathcal{L}^{N_2} = (u_{2,0}y_{2,0})\,(u_{2,2}y_{2,4})\,(u_{2,3}y_{2,4})\,(u_{2,3}y_{2,6})\ (u_{2,3}y_{2,7})\ (u_{2,2}y_{2,7})\ (u_{2,2}y_{2,6})\ (u_{2,2}y_{2,4})\ (u_{2,0}y_{2,0})\ (u_{2,2}y_{2,4})$ a instantes $\tau_0, \tau_6, \tau_{10}, \tau_{12}, \tau_{13}, \tau_{14}, \tau_{15}, \tau_{16}, \tau_{18}, \tau_{28}$.

$\mathcal{L}^{N_3} = (u_{3,0}y_{3,0})\,(u_{3,3}y_{3,2})\,(u_{3,3}y_{3,3})\,(u_{3,3}y_{3,2})\,(u_{3,3}y_{3,0})\,(u_{3,0}y_{3,0})$ a instantes $\tau_0, \tau_3, \tau_5,\ \tau_{20}, \tau_{21}, \tau_{22}$.

$\mathcal{L}^{N_4} = (u_{4,0}y_{4,0})\,(u_{4,3}y_{4,0})\,(u_{4,3}y_{4,2})\,(u_{4,3}y_{4,3})\,(u_{4,3}y_{4,2})\,(u_{4,3}y_{4,0})$ a instantes $\tau_0, \tau_7, \tau_8,\ \tau_9, \tau_{17}, \tau_{19}$.

$\mathcal{L}^{N_5} = (u_{5,0}y_{5,0})\,(u_{5,3}y_{5,0})\,(u_{5,3}y_{5,2})\,(u_{5,3}y_{5,3})\,(u_{5,3}y_{5,2})\,(u_{5,3}y_{5,0})$ a instantes $\tau_0, \tau_{11}, \tau_{12},\ \tau_{13}, \tau_{15}, \tau_{16}$.

Las funciones de densidad de probabilidad del tiempo de disparo de las transiciones (δ_l) se presentan en la Tabla 5.6.4.

Las st-IPNs para esta información se muestran en la Figura 5.6.9

Paso 2:

Se asumen posibles fallos genéricos en todos los lugares de todos los subsistemas involucrados es en este escenario de simulación.

Paso 3: Monitorización On-line

Induciendo fallos en la bomba del subsistema 4 a los 25 min., en la válvula v_{c1} del subsistema 2 a los 40 min., en el calentador a los 45 min. y en la bomba general del subsistema 1 a los 50 min. después que el sistema empieza a operar; los eventos observados a cada instante son:

Evento $\omega_1^1 = (u_{1,30}y_{1,9})$,21.

Aplicando el algoritmo:

$sub = 1$; $Pt_{\Omega_{N_1}}(\omega_1^1) = (u_{1,30}y_{1,9})$,21, entonces $\omega_1^1 \in \mathcal{L}^{N_1}$; por lo tanto $m(p_{1,1}) = \langle N_1 \rangle$.

Evento $\omega_1^2 = (u_{1,30}y_{1,13})$,4.

$sub = 1$; $Pt_{\Omega_{N_1}}(\omega_1^2) = (u_{1,30}y_{1,13})$,4; ya que $(u_{1,30}y_{1,13}) \in \Omega_{N_1} \wedge \left(a \leq t_{l,o_{om}}^{ev} \leq b\right)$, siendo $[a, b]$ el intervalo de confianza de $tr_{1,2}$[4], entonces $\omega_1^2 \in \mathcal{L}^{N_1}$ y $m(p_{1,2}) = \langle N_1 \rangle$.

Evento $\omega_3^3 = (u_{3,3}y_{3,2})$,109.

$sub = 3$; $Pt_{\Omega_{N_3}}(\omega_3^3) = (u_{3,3}y_{3,2})$,109, entonces $\omega_3^3 \in \mathcal{L}^{N_3}$ y $m(p_{3,1}) = \langle N_3 \rangle$.

Evento $\omega_1^4 = (u_{1,15}y_{1,15})$,85.

$sub = 1$; $Pt_{\Omega_{N_1}}(\omega_1^4) = (u_{1,15}y_{1,15})$,85, entonces $\omega_1^4 \in \mathcal{L}^{N_1}$ y $m(p_{1,3}) = \langle N_1 \rangle$.

Evento $\omega_3^5 = (u_{3,3}y_{3,3})$,28.

$sub = 3$; $Pt_{\Omega_{N_3}}\omega_3^5 = (u_{3,3}y_{3,3})$,28, entonces $\omega_3^5 \in \mathcal{L}^{N_3}$ y $m(p_{3,2}) = \langle N_3 \rangle$.

Evento $\omega_2^6 = (u_{2,2}y_{2,4})$,201.

$sub = 2$; $Pt_{\Omega_{N_2}}\omega_2^6 = (u_{2,2}y_{2,4})$,201, entonces $\omega_2^6 \in \mathcal{L}^{N_2}$ y $m(p_{2,1}) = \langle N_2 \rangle$.

Evento $\omega_4^7 = (u_{4,3}y_{4,0})$,209.

$sub = 4$; $Pt_{\Omega_{N_4}}\omega_4^7 = (u_{4,3}y_{4,0})$,209, entonces $\omega_4^7 \in \mathcal{L}^{N_4}$ y $m(p_{4,1}) = \langle N_4 \rangle$.

Evento $\omega_4^8 = (u_{4,3}y_{4,2})$,9.

$sub = 4$; $Pt_{\Omega_{N_4}}\omega_4^8 = (u_{4,3}y_{4,2})$,9, entonces $\omega_4^8 \in \mathcal{L}^{N_4}$ y $m(p_{4,2}) = \langle N_4 \rangle$.

Evento $\omega_4^9 = (u_{4,3}y_{4,3})$,30.

$sub = 4$; $Pt_{\Omega_{N_4}}\omega_4^9 = (u_{4,3}y_{4,3})$,30, entonces $\omega_4^9 \in \mathcal{L}^{N_4}$ y $m(p_{4,3}) = \langle N_4 \rangle$.

[4] $a = 3{,}8064 - 1{,}96 * 0{,}401$ y $b = 3{,}8064 + 1{,}96 * 0{,}401$ (ver Tabla 5.6.4)

Evento $\omega_4^{10} = (u_{4,3}y_{4,2})$,14.

$sub = 4$; $Pt_{\Omega_{N_4}}\omega_4^{10} = \varepsilon$[5], $t_{proy} = 14$, entonces $\omega_4^{10} \notin \mathcal{L}^{N_4}$, por lo tanto ω_4^{10} es un fallo genérico en el subsistema 4; $df_4 = 1$, la transición $tr_{4,9} \in TR_{F_4}$ se dispara en modo m_f_1; $pre = [00\,\langle(4,2)\,gf_1 \wedge N_4\rangle\,000]$, $post = [00\,\langle(4,2)\,gf_1\rangle\,0\,\langle(4,2)\,gf_1\rangle\,\langle x\rangle]$ y $m\,(p_{4,vf_4}) = m\,(p_{4,2}) = \langle(4,2)\,gf_1\rangle$ y $m\,(p_{4,if_4}) = \langle 1\rangle$. Este fallo se adiciona en la Tabla 6.5.1.

Se ejecuta el Algoritmo 6.2. gf_1 no es fallo de propagación.

Evento $\omega_4^{11} = (u_{4,3}y_{4,3})$,62.

$sub = 4$; $Pt_{\Omega_{N_4}}\omega_4^{11} = \varepsilon$[6], $t_{proy} = 14 + 62$, entonces $\omega_4^{11} \notin \mathcal{L}^{N_4}$, por lo tanto ω_4^{11} es un fallo genérico en el subsistema 4; $df_4 = 2$, la transición $tr_{4,9} \in TR_{F_4}$ se dispara en modo m_f_2; con $pre = [00\,\langle(4,2)\,gf_1 \wedge N_4; (4,2)\,gf_2 \wedge N_4\rangle\,000]$, $post = [00\,\langle(4,2)\,gf_1; (4,2)\,gf_2\rangle\,0\,\langle(4,2)\,gf_1; (4,3)\,gf_2\rangle\,\langle x\rangle]$ y $m\,(p_{4,vf_4}) = m\,(p_{4,2}) = \langle(4,2)\,gf_2\rangle$ y $m\,(p_{4,if_4}) = \langle 2\rangle$. Este fallo se adiciona en la Tabla 6.5.1.

Se ejecuta el Algoritmo 6.2, entonces: $sr_{4,i}^{ex} = s_{4,2}$ $s_{4,2} \in SFS_r$ y $s_{4,2} \subset ISFS_r$.

Evento $\omega_2^{12} = (u_{2,3}y_{2,4})$,100.

$sub = 2$; $Pt_{\Omega_{N_2}}\omega_2^{12} = (u_{2,3}y_{2,4})$,100, entonces $\omega_2^{12} \in \mathcal{L}^{N_2}$ y $m\,(p_{2,2}) = \langle N_2\rangle$.

Evento $\omega_5^{13} = (u_{5,3}y_{5,0})$,209.

$sub = 5$; $Pt_{\Omega_{N_5}}\omega_5^{13} = (u_{5,3}y_{5,0})$,109, entonces $\omega_5^{13} \in \mathcal{L}^{N_5}$ y $m\,(p_{5,1}) = \langle N_5\rangle$.

Evento $\omega_2^{14} = (u_{2,3}y_{2,6})$,13; $\omega_5^{14} = (u_{5,3}y_{5,2})$,7.

[5] $(u_{4,3}y_{4,2}) \in \Omega_{N_4}$ pero $14 \notin [a,b]$ (intervalo de confianza del tiempo de la $tr_{4,4}$).

[6] $(u_{4,3}y_{4,2}) \notin \Omega_{N_4}$.

Tabla 6.5.1: Fallos detectados en AHS

Fallo	Instante	Sub sistema	Lugar	Tipo de Fallo	Señal esperada	Señal observada	Señal medida	$s_{l,i}$	$ISFS_r$
$(4,2)\,gf_1$	26.2	4	2	Fallo por tiempo	[10]	[10]			[]
$(4,2)\,gf_2$	30	4	2	Salida	[10]	[11]	F_4NF_4	$s_{4,2}$	$[s_{4,2}]$
$(4,2)\,gf_3$	34.1	4	2	Fallo por tiempo	[10]	[10]			$[s_{4,2}]$
$(2,5)\,gf_1$	40.1	2	5	Salida	[111]	[010]	$\overline{P}_oF_2\overline{NF}_2$	$s_{2,2}$	$[s_{4,2}, s_{2,2}]$
$(4,2)\,gf_4$	40.3	4	2	Salida	[10]	[00]	$\overline{F}_4\overline{NF}_4$	$s_{4,1}, s_{4,2}$	$[s_{4,2}, s_{2,2}, s_{4,1}]$
$(2,5)\,gf_2$	40.5	2	5	Salida	[111]	[000]	$\overline{P}_o\overline{F}_2\overline{NF}_2$	$s_{2,1}, s_{2,2}, s_{2,3}$	$[s_{4,2}, s_{2,2}, s_{2,3}]$
$(1,4)\,gf_1$	45.2	1	4	Salida	[1101]	[1001]	$T_o\overline{T}_1\overline{T}_2F_1$	$s_{1,2}$	
$(3,2)\,gf_1$	53.6	3	2	Fallo por tiempo	[10]	[10]			
$(3,2)\,gf_2$	60.3	3	2	Salida	[10]	[00]	$\overline{F}_3\overline{NF}_3$	$s_{3,1}$	$[s_{4,2}, s_{2,2}, s_{2,3}, s_{3,1}]$

$sub = 2$; $Pt_{\Omega_{N_2}}\omega_2^{14} = (u_{2,3}y_{2,6})$,13, entonces $\omega_2^{14} \in \mathcal{L}^{N_2}$ y $m\,(p_{2,3}) = \langle N_2 \rangle$.

$sub = 5$; $Pt_{\Omega_{N_5}}\omega_5^{14} = (u_{5,3}y_{5,2})$,7, entonces $\omega_5^{14} \in \mathcal{L}^{N_5}$ y $m\,(p_{5,2}) = \langle N_5 \rangle$.

Evento $\omega_2^{15} = (u_{2,3}y_{2,7})$,26; $\omega_5^{15} = (u_{5,3}y_{5,3})$,26.

$sub = 2$; $Pt_{\Omega_{N_2}}\omega_2^{15} = (u_{2,3}y_{2,7})$,26, entonces $\omega_2^{15} \in \mathcal{L}^{N_2}$ y $m\,(p_{2,4}) = \langle N_2 \rangle$.

$sub = 5$; $Pt_{\Omega_{N_5}}\omega_5^{15} = (u_{5,3}y_{5,3})$,26, entonces $\omega_5^{15} \in \mathcal{L}^{N_5}$ y $m\,(p_{5,3}) = \langle N_5 \rangle$.

Evento $\omega_4^{16} = (u_{4,3}y_{4,2})$,41.

$sub = 4$; $Pt_{\Omega_{N_4}}\omega_4^{16} = \varepsilon$[7], entonces $\omega_4^{16} \notin \mathcal{L}^{N_4}$, por lo tanto ω_4^{16} es un fallo genérico en el subsistema 4; $df_4 = 3$, la transición $tr_{4,9} \in TR_{F_4}$ se dispara en modo m_f_3; con $pre = [00\,\langle (4,2)\,gf_1 \wedge N_4; (4,2)\,gf_2 \wedge N_4; (4,2)\,gf_3 \wedge N_4 \rangle\,000]$, $post = [00\,\langle (4,2)\,gf_1; (4,2)\,gf_2; (4,2)\,gf_3 \rangle\,0\,\langle (4,2)\,gf_1; (4,2)\,gf_2; (4,2)\,gf_3 \rangle\,\langle x \rangle]$ y $m\,(p_{4,vf_4}) = m\,(p_{4,2}) = \langle (4,2)\,gf_3 \rangle$ y $m\,(p_{4,if_4}) = \langle 3 \rangle$. Este fallo se adiciona en la Tabla 6.5.1.

Se ejecuta el Algoritmo 6.2. gf_3 no es fallo de propagación.

Evento $\omega_2^{17} = (u_{2,2}y_{2,3})$,60.

$sub = 2$; $Pt_{\Omega_{N_2}}\omega_2^{17} = (u_{2,2}y_{2,3})$,60, entonces $\omega_2^{17} \in \mathcal{L}^{N_2}$ y $m\,(p_{2,5}) = \langle N_2 \rangle$.

Evento $\omega_2^{18} = (u_{2,2}y_{2,2})$,1; $\omega_5^{18} = (u_{5,3}y_{5,2})$,61.

$sub = 2$; $Pt_{\Omega_{N_2}}\omega_2^{18} = \varepsilon$, entonces $\omega_2^{18} \notin \mathcal{L}^{N_2}$, por lo tanto ω_2^{18} es un fallo genérico en el subsistema 2; $df_2 = 1$, la transición $tr_{2,15} \in TR_{F_2}$ se dispara en modo m_f_1; con $pre = [00000\,\langle (2,5)\,gf_1 \wedge N_2 \rangle\,00000]$,

[7] $(u_{4,3}y_{4,2}) \in \Omega_{N_4}$ $t^{ev} = 14{+}62{+}41 = 117$ $t^{ev} \notin [a,b] = 253{,}42 \pm 1{,}96 * 0{,}51$ $[a,b]$ intervalo de confianza de la $tr_{4,4}$

$post = [00000\,\langle(2,5)\,gf_1\rangle\,000\,\langle(2,5)\,gf_1\rangle\,\langle x\rangle]$ y $m\,(p_{2,vf_2}) = m\,(p_{2,5}) = \langle(2,5)\,gf_1\rangle$ y $m\,(p_{2,if_2}) = \langle 1\rangle$. Este fallo se adiciona en la Tabla 6.5.1.

Se ejecuta el Algoritmo 6.2. $sr^{ex}_{2,i} = s_{2,2}$ $s_{2,2} \in SFS_r$ y $s_{2,2} \subset ISFS_r$. Este subsistema no se ha recuperado del fallo.

$sub = 5$; $Pt_{\Omega_{N_5}}\omega_5^{18} = (u_{5,3}y_{5,2})$,61, entonces $\omega_5^{18} \in \mathcal{L}^{N_5}$ y $m\,(p_{5,4}) = \langle N_5\rangle$.

Evento $\omega_4^{19} = (u_{4,3}y_{4,0})$,62.

$sub = 4$; $Pt_{\Omega_{N_4}}\omega_4^{19} = \varepsilon$, entonces $\omega_4^{18} \notin \mathcal{L}^{N_4}$, por lo tanto ω_4^{18} es un fallo genérico en el subsistema 4; $df_4 = 4$, la transición $tr_{4,9} \in TR_{F_4}$ se dispara en modo m_f_4; con $pre = [00\,\langle(4,2)\,gf_1 \wedge N_4; (4,2)\,gf_2 \wedge N_4; (4,2)\,gf_3 \wedge N_4; (4,2)\,gf_4 \wedge N_4\rangle\,000]$, $post = [00\,\langle(4,2)\,gf_1; (4,2)\,gf_2; (4,2)\,gf_3; (4,2)\,gf_4\rangle\,0\,\langle(4,2)\,gf_1; (4,2)\,gf_2; (4,2)\,gf_3; (4,2)\,gf_4\rangle\,\langle x\rangle]$ y $m\,(p_{4,vf_4}) = m\,(p_{4,2}) = \langle(4,2)\,gf_4\rangle$ y $m\,(p_{4,if_4}) = \langle 4\rangle$. Este fallo se adiciona en la Tabla 6.5.1. Este subsistema no se ha recuperado del fallo.

Evento $\omega_2^{20} = (u_{2,2}y_{2,0})$,4; $\omega_5^{20} = (u_{3,3}y_{3,0})$,4.

$sub = 2$; $Pt_{\Omega_{N_2}}\omega_2^{20} = \varepsilon$, entonces $\omega_2^{20} \notin \mathcal{L}^{N_2}$, por lo tanto ω_2^{20} es un fallo genérico en el subsistema 2; $df_2 = 2$, la transición $tr_{2,10} \in TR_{F_2}$ se dispara en modo m_f_2; con $pre = [00000\,\langle(2,5)\,gf_1 \wedge N_2; (2,5)\,gf_2 \wedge N_2\rangle\,00000]$, $post = [00000\,\langle(2,5)\,gf_1; (2,5)\,gf_2\rangle\,000\,\langle(2,5)\,gf_1; (2,5)\,gf_2\rangle\,\langle x\rangle]$ y $m\,(p_{2,vf_2}) = m\,(p_{2,5}) = \langle(2,5)\,gf_2\rangle$ y $m\,(p_{2,if_2}) = \langle 2\rangle$. Este fallo se adiciona en la Tabla 6.5.1.

Se ejecuta el Algoritmo 6.2.$sr^{ex}_{2,i} = s_{2,1}, s_{2,2}, s_{2,3}$. $s_{2,2}, s_{2,3} \in SFS_r$ y $s_{2,3} \subset ISFS_r$. Este subsistema no se ha recuperado del fallo.

$sub = 5$; $Pt_{\Omega_{N_5}}\omega_5^{20} = (u_{3,3}y_{3,0})$,4, entonces $\omega_5^{20} \in \mathcal{L}^{N_5}$ y $m\,(p_{5,5}) = \langle N_5\rangle$.

Evento $\omega_1^{21} = (u_{1,15}y_{1,13})$,342.

$sub = 1$; $Pt_{\Omega_{N_1}}\omega_1^{21} = (u_{1,15}y_{1,13})$,342, entonces $\omega_1^{21} \in \mathcal{L}^{N_1}$ y $m\,(p_{1,4}) = \langle N_1\rangle$.

Evento $\omega_1^{22} = (u_{1,15} y_{1,9})$,6.

$sub = 1$; $Pt_{\Omega_{N_1}} \omega_1^{22} = \varepsilon$, entonces $\omega_1^{22} \notin \mathcal{L}^{N_2}$, por lo tanto ω_1^{22} es un fallo genérico en el subsistema 1; $df_1 = 1$, la transición $tr_{2,14} \in TR_{F_2}$ se dispara en modo m_f_1; con $pre = [0000 \langle (1,4)\, gf_1 \wedge N_1 \rangle\, 000000]$, $post = [0000 \langle (1,4)\, gf_1 \rangle\, 0000 \langle (1,4)\, gf_1 \rangle \langle x \rangle]$ y $m(p_{1,vf_1}) = m(p_{1,4}) = \langle (1,4)\, gf_1 \rangle$ y $m(p_{1,if_1}) = \langle 1 \rangle$. Este fallo se adiciona en la Tabla 6.5.1.

Se ejecuta el Algoritmo 6.2. ω_1^{22} no es fallo de propagación.

Evento $\omega_3^{23} = (u_{3,3} y_{3,2})$,399.

$sub = 3$; $Pt_{\Omega_{N_3}} \omega_3^{23} = \varepsilon$, entonces $\omega_3^{23} \notin \mathcal{L}^{N_3}$, $t_{proy} = 399$; por lo tanto ω_3^{23} es un fallo genérico en el subsistema 3; $df_1 = 1$, la transición $tr_{3,8} \in TR_{F_3}$ se dispara en modo m_f_1; con $pre = [00 \langle (3,2)\, gf_1 \wedge N_3 \rangle\, 0000]$, $post = [00 \langle (3,2)\, gf_1 \rangle\, 00 \langle (3,2)\, gf_1 \rangle \langle x \rangle]$ y $m(p_{3,vf_3}) = m(p_{3,2}) = \langle (3,2)\, gf_1 \rangle$ y $m(p_{3,if_3}) = \langle 1 \rangle$. Este fallo se adiciona en la Tabla 6.5.1.

Se ejecuta el Algoritmo 6.2. ω_3^{23} no es fallo de propagación.

Evento $\omega_3^{24} = (u_{3,3} y_{3,0})$,67.

$sub = 3$; $Pt_{\Omega_{N_3}} \omega_3^{24} = \varepsilon$, entonces $\omega_3^{24} \notin \mathcal{L}^{N_3}$, por lo tanto ω_3^{24} es un fallo genérico en el subsistema 3; $df_1 = 2$, la transición $tr_{3,8} \in TR_{F_3}$ se dispara en modo m_f_2; con $pre = [00 \langle (3,2)\, gf_1 \wedge N_3; (3,2)\, gf_2 \wedge N_3 \rangle\, 0000]$, $post = [00 \langle (3,2)\, gf_1; (3,2)\, gf_2 \rangle\, 00 \langle (3,2)\, gf_1; (3,2)\, gf_2 \rangle \langle x \rangle]$ y $m(p_{3,vf_3}) = m(p_{3,2}) = \langle (3,2)\, gf_2 \rangle$ y $m(p_{3,if_3}) = \langle 2 \rangle$. Este fallo se adiciona en la Tabla 6.5.1.

Se ejecuta el Algoritmo 6.2. $sr_{3,i}^{ex} = s_{3,1}$ $s_{3,1} \in SFS_r$ y $s_{3,1} \subset ISFS_r$.

Por lo tanto $\omega_1^0 \omega_1^1 \omega_1^2 \omega_1^3 \omega_1^5 \omega_1^7$ es una traza de fallo, es decir $\omega_1^0 \omega_1^1 \omega_1^2 \omega_1^3 \omega_1^5 \omega_1^7 \subset \mathcal{L}^{f_1}$.

Los diagnosticadores encontrados para los subsistemas analizados se muestran en las Figuras: Figura 6.5.1, Figura 6.5.2, Figura 6.5.3, Figura 6.5.4 y Figura 6.5.5, respectivamente.

En los diagnosticadores se puede observar que en el lugar de verificación de cada subsistema se presentan las marcas de fallos detectados. Siendo 2, 2, 2, 3 y 0 los fallos detectados para los cinco subsistemas respectivamente.

6.5.2. Identificación del fallo

Con base en la Tabla 6.5.1, $(4,2)\,gf_1$ es un fallo detectado en el subsistema 4 debido a que el subsistema tarda más tiempo que el normal en alcanzar el flujo deseado, aunque el subsistema recupera las señales, éste permanece en modo de fallo, situación que es observable en los instantes 30, 34.1, y 40.3. Estos no son fallos de propagación puesto que anteriormente no se han detectado en la línea de flujo. En el subsistema 2 a los 40,1 min, se detecta un fallo $(2,5)\,gf_1$ de posición en la válvula vc_1 y no se ha alcanzado un flujo normal; luego se bloquea totalmente el flujo, como se observa en el fallo $(2,5)\,gf_2$. En el subsistema 1 a los 45,2 min se detecta un fallo de temperatura $(1,4)\,gf_1$, se asume por fallo en el calentador, el fallo de la bomba general que se indujo a los 50 min no se detecta puesto que el sistema no se ha recuperado del fallo. En el subsistema 3, a los 53,6 min se detecta un fallo temporizado $(3,2)\,gf_1$ ya

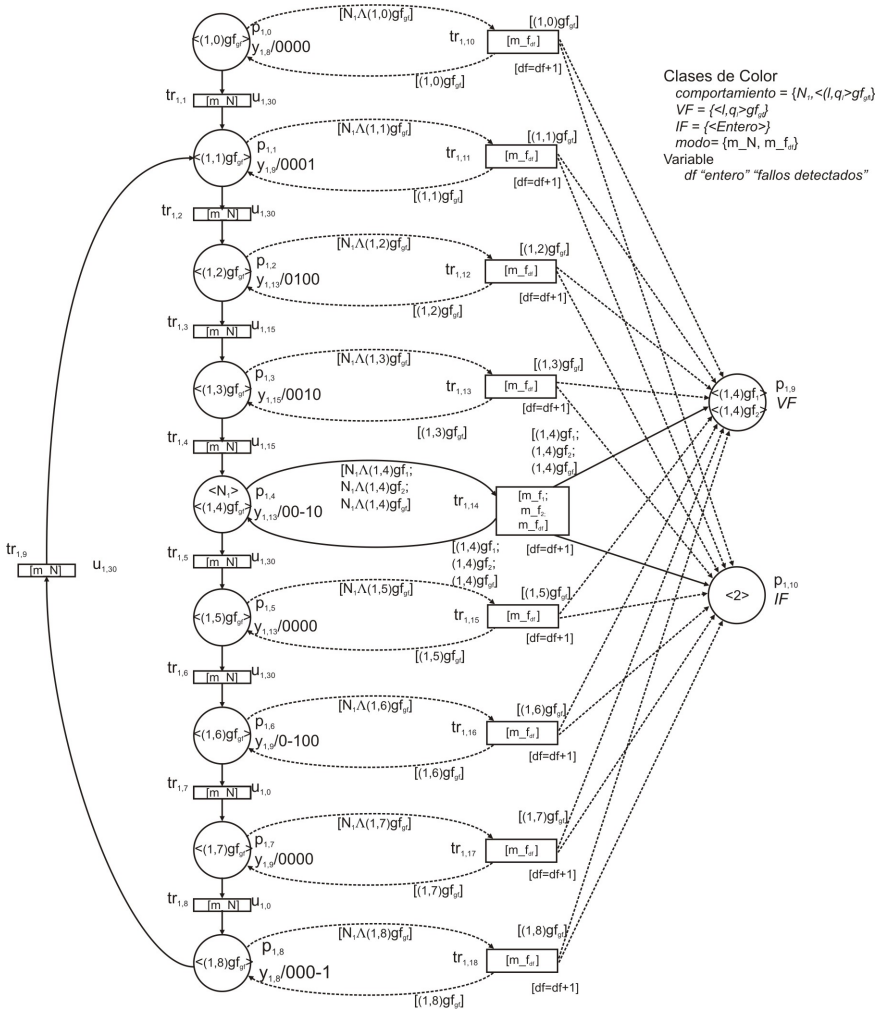


Figura 6.5.1: st-DICPN Subsistema 1

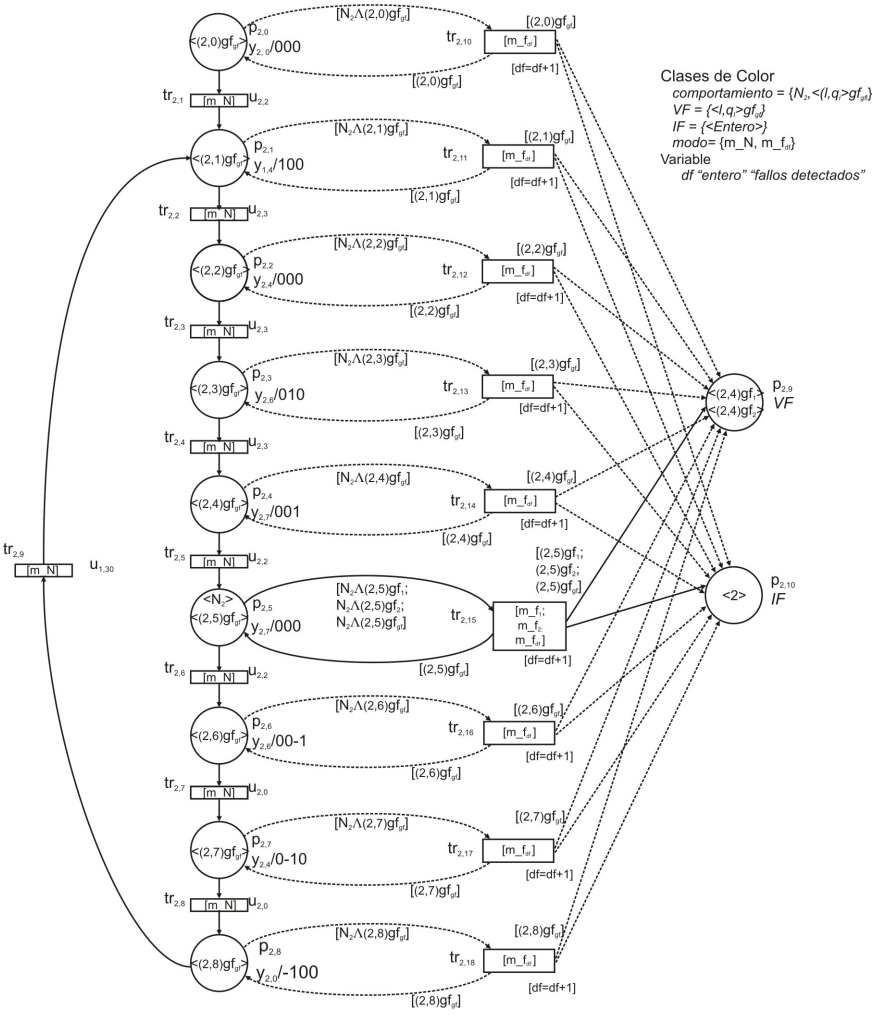


Figura 6.5.2: st-DICPN Subsistema 2

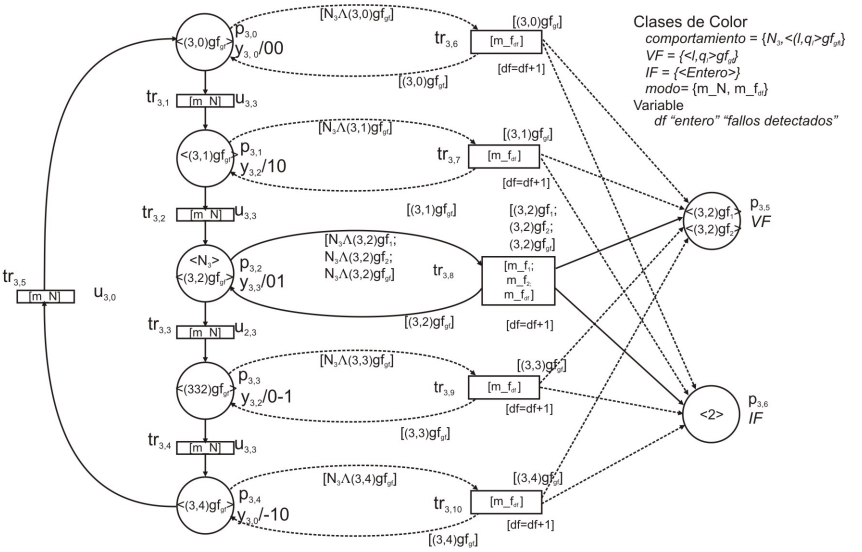


Figura 6.5.3: st-DICPN Subsistema 3

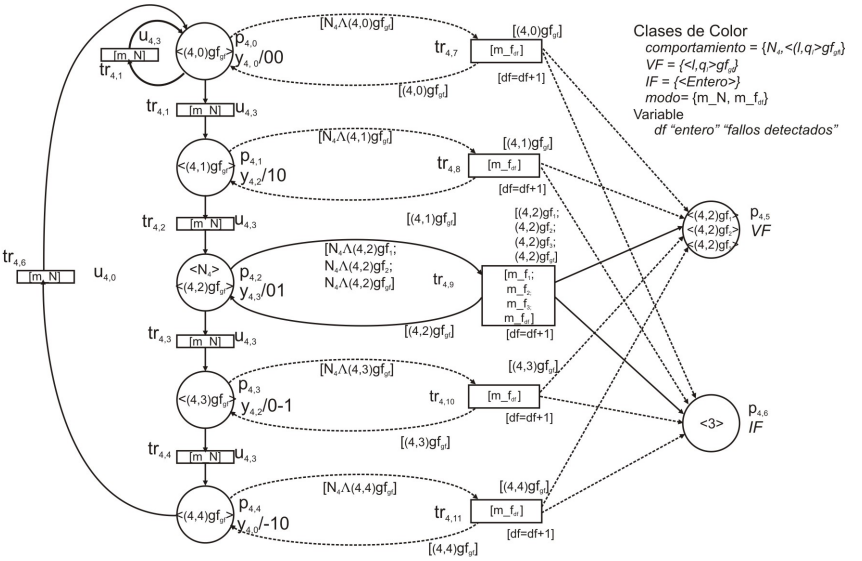


Figura 6.5.4: st-DICPN Subsistema 4

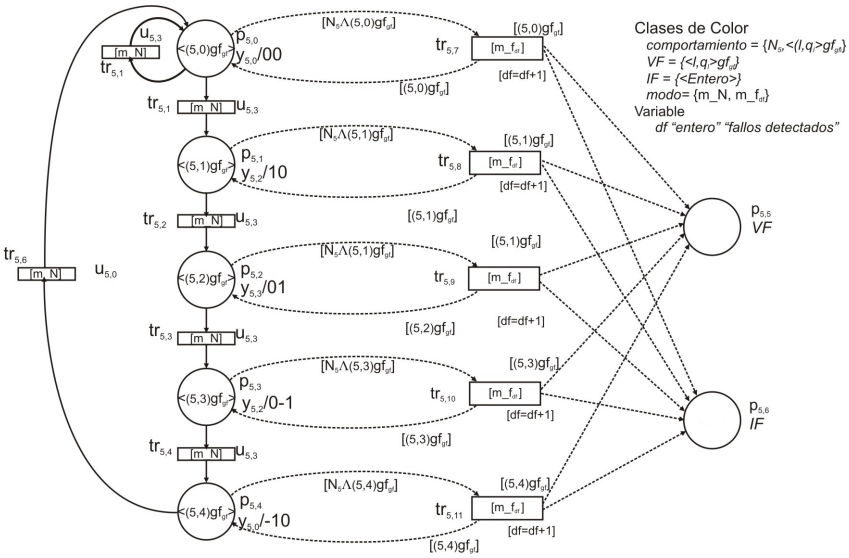


Figura 6.5.5: st-DICPN Subsistema 5

que el flujo no alcanza el nivel deseado en el tiempo normal y en el minuto 60,3 el flujo cae, el fallo detectado es en un sensor de flujo $sr_{3,1}$. $sr_{3,1}$ pertenece a la línea de fallos de flujo y un fallo en un sensor de flujo se ha dado previamente, pero no ha sido detectado; por lo tanto el fallo no es eliminado.

6.6. Comparación con otros Métodos

Con el propósito de mostrar las contribuciones del presente estudio, en esta sección se realiza una comparación de los métodos relacionados en Sección 3.4 con el método de diagnóstico propuesto.

La primera propuesta a comparar es el método de diagnóstico presentado en [1], la cual se denominará *"IPN no-determinista",* la segunda propuesta es el enfoque de residuos presentado por [3], que se denominará *"Diagnosticador NDAAO"* y la presenta propuesta *"st-DICPN".*

Estos enfoques presentan diferentes características en cuanto a: utilización de modelos del sistema en comportamiento de fallo o libre de fallo, utilización del tiempo en el proceso de diagnóstico y herramienta de modelado del diagnosticador.

Para analizar las ventajas y desventajas de cada uno, se propone realizarlo con base a los siguientes criterios adaptados de [103]:

- Habilidad para detectar (Detectabilidad) y diagnosticar un fallo (Diagnosticabilidad).
- Precisión en la localización del Fallo.
- Generación de falsas alarmas.
- Aplicabilidad a sistemas reales.

6.6.1. Detectabilidad y Diagnosticabilidad

La *"IPN no-determinista",* es construida a partir de un conjunto de fallos predefinidos, por lo tanto su habilidad para detectar y diagnosticar fallos depende de si el fallo está contenido dentro de este conjunto; además, cuando se genera el proceso de diagnóstico on-line y dada una secuencia de eventos observados, la estrategia

de detección de fallos se basa en el cálculo de una función diagnosticador que se soluciona como un ILPP y permite decidir si el comportamiento es normal, de fallo o ambiguo; un comportamiento es ambiguo cuando existe duda si el fallo ha sucedido o no, comportamiento que puede ser consecuencia que el modelo de la red es no-determinista; en la solución del ILPP del diagnosticador se incluye la información del tiempo, con el objeto que el algoritmo sea más preciso.

El *"Diagnosticador NDAAO",* no tiene modelo del comportamiento en fallo; un fallo es detectado comparando comportamientos. Su habilidad para detectar fallos, en un proceso on-line, depende de la precisión del modelo identificado a partir del lenguaje observado; esta precisión depende entre otros aspectos del tamaño de las palabras que componen el lenguaje; por lo tanto se pierde precisión en la medida que el tamaño del sistema aumenta, además el modelo identificado es no-determinista. A partir del estado inicial y dada una secuencia de lecturas de E/S y de los "bordes", se compara con el modelo identificado, si la secuencia observada no coincide con el diagnosticador un fallo es detectado. Los fallos detectados se relacionan con fallos en sensores o actuadores.

La *"st-DICPN"* detecta fallos a partir de la comparación del comportamiento observado con el lenguaje normal identificado previamente; por lo tanto, igual que en el método anterior, la habilidad para detectar fallos depende del modelo identificado. El modelo identificado es determinístico, generado a partir del lenguaje compuesto por palabras que se construyen con cada nuevo evento, cuya longitud la determina el comportamiento del sistema. Las transiciones identificadas en el modelo, consideran un rango de tiempo como intervalo de confianza de una función de densidad de probabilidad, en el cual su comportamiento es normal. A partir de un estado normal y dado un evento, la detección del fallo se realiza comparando las etiquetas de las transiciones habilitadas, las fun-

ciones de salida en los lugares alcanzados y del tiempo transcurrido en la estructura de la PN propuesta. Por lo tanto, este método detecta fallos por fallos en actuadores, en sensores y en el tiempo del evento. La tarea de identificación del fallo, para completar el diagnóstico, igual que en el método anterior la realiza un experto en el sistema.

En resumen; La *"IPN no-determinista"* puede detectar los fallos predefinidos bajo un modelo no-determinista, teniendo en cuenta el tiempo; el *"Diagnosticador NDAAO"* puede detectar cualquier tipo de fallo que pueda ser medido por las señales de E/S bajo un modelo no-determinista sin tener en cuenta el tiempo y el método propuesto en esta investigación también puede detectar cualquier tipo de fallo que pueda ser medido por las señales de E/S; pero bajo un modelo determinista y con una estructura de tiempo que permite detectar algunos fallo que no pueden ser medidos por las señales de E/S.

6.6.2. Precisión en la localización del fallo

La *"IPN no-determinista"* garantiza la localización del fallo puesto que son fallos predefinidos, el *"Diagnosticador NDAAO"* y la *"st-DICPN"* presentan propuestas basadas en el cálculo de residuos y en el análisis del lenguaje de salida respectivamente; pero la precisión en la localización no es totalmente garantizada, se localiza la señal que ha fallado; pero no es posible distinguir si el fallo es por un dispositivo o por un fallo de sensor o del actuador.

6.6.3. Generación de falsas alarmas

Las falsas alarmas pueden ser generadas por diferentes causas como fallos en la instrumentación del sistema o fallos por flujo compar-

tido. Un punto importante que deben tener los métodos de diagnóstico es contemplar la relación temporal y causal que tienen los fallos. Los tres métodos analizados no garantizan la detección de falsas alarmas puesto que son modelos incompletos de fallo [103]; la *"st-DICPN"* propone un método de filtrado de fallos, el cual puede eliminar fallos por falsas alarmas generados por flujo compartido entre subsistemas, a partir de la definición temporal de los fallos detectados.

6.6.4. Aplicabilidad del Diagnosticador

La mayoría de los sistemas dinámicos de gran escala pueden considerarse SED en un determinado nivel de abstracción. En este sentido, los métodos de diagnóstico de fallos basados en modelos de eventos discretos son aplicables no solamente a los sistemas específicos clasificables como SED, sino también a los sistemas de naturaleza continua.

Bajo un enfoque centralizado en *"IPN no-determinista",* el tamaño del modelo crece en la medida que el número de fallos definidos aumenta y la solución del diagnosticador incrementa su complejidad; por lo tanto, su aplicación en sistemas complejos se ve afectada; además el conocimiento previo del sistema debe ser muy exhaustivo para que el modelo se ajuste mejor al comportamiento real del sistema. Para la aplicación en sistemas de gran tamaño los autores proponen un enfoque distribuido. El modelo sobre el cual se realiza el diagnóstico es no-determinístico y esta condición no es una condición de los sistemas industriales, ya que el modelo genera un comportamiento ambiguo pero el sistema no.

La *"st-DICPN"* y el *"Diagnosticador NDAAO",* utilizan señales de E/S del sistema observadas directamente el sistema, el conocimiento previo del sistema no es muy profundo; por lo tanto, pueden ser

aplicables a cualquier tipo de sistemas con señales que puedan ser representadas de manera binaria y cuyo comportamiento pueda ser modelado por lenguajes.

El *"Diagnosticador NDAAO",* presenta inconvenientes cuando se aplica a sistemas de gran escala, puesto que la precisión del modelo identificado depende del tamaño de las palabras de lenguaje; además, es no-determinista y no es aplicable en sistemas que presenten variabilidad en la generación de señales. Bajo un enfoque distribuido existe una propuesta de diagnóstico.

La *"st-DICPN"* es un método de diagnóstico con enfoque centralizado, cuando se aplica a sistemas de gran escala el sistema se identifica dividiéndolo en subsistemas; pero cada subsistema se modela de manera independiente; por lo tanto, se conservan las propiedades del enfoque centralizado. Cada modelo es determinista, no solo tiene en cuenta las señales intercambiadas entre el controlador y la planta, sino que además incluye señales externas que afectan al controlador e influyen en el tiempo de los eventos. Cuando una secuencia de fallo es detectada, ésta es aprendida por el diagnosticador en modo de fallo; de esta manera el diagnosticador cada vez que es ejecutado, se aproxima al comportamiento real del sistema.

6.7. Conclusiones y aportes del capítulo

En el este capítulo se presentó una propuesta de diagnóstico de fallos en SED estocásticos. El método detecta y aísla fallos sin conocimiento previo del modelo del sistema. Para hallar el diagnosticador, primero se monitorea el sistema hasta encontrar un modelo, cuya representación presenta las características mostradas en el capítulo de modelado y el proceso de consecución del modelo se realiza con base en el método de identificación propuesto en el capítulo 5; una vez encontrado el modelo se convierte en una red

de Petri coloreada y sobre ella se realiza el proceso de diagnóstico. El diagnosticador presentado es un generador determinista de lenguaje temporizado, con características particulares en cuanto a que puede detectar cualquier tipo de fallo que influya de alguna manera en los valores de las señales del sistema, presenta una estructura sin bloqueos que permite evolucionar la red, una vez el fallo haya sido reparado; permite realizar análisis de detectabilidad basado en la operación de proyección definida en el presente documento. El diagnosticador aprende los lenguajes de fallos detectados.

El método de diagnóstico propuesto, se puede aplicar a cualquier tipo de sistema, independiente de su tamaño.

7 Validación

En este capítulo se presenta la aplicación del método de diagnóstico en sistemas reales.

7.1. Sistema Fotovoltaico (PVS)

La energía solar fotovoltaica es una de las alternativas de generación renovables más extendidas; las instalaciones fotovoltaicas tienen aplicaciones que van desde usos domésticos hasta grandes parques solares, la instalación implica una inversión significativa que debe ser cuidada para preservar la productividad durante su vida útil, que en el caso del panel fotovoltaico puede superar los 20 años. Un conjunto fotovoltaico puede seguir trabajando en estado de fallo aparentando un comportamiento normal con algunas pérdidas de energía; si el estado de fallo se mantiene, un daño permanente o el envejecimiento prematuro en los componentes puede ocurrir, [104]. Por lo tanto, hacer un adecuado mantenimiento y rápidas reparaciones en los componentes del sistema, permiten obtener la máxima generación posible.

La mayoría de las técnicas de diagnóstico de fallos parten de un modelo creado a partir de la ecuación del circuito equivalente de una célula fotovoltaica en función de la irradiancia y temperatura, esto permite obtener la caracterización de la célula, es decir, la curva de comportamiento I-V y por consiguiente la caracterización

del módulo, [105, 106], que incluye valores como: Voltaje de circuito abierto (Voc), Intensidad de corto circuito (Isc), Voltaje máximo (Vmax), Corriente máxima (Imax) y punto de máxima potencia (MPP) para la radiación de condiciones estándar de prueba (STC).

7.1.1. Descripción del Sistema y Adecuación de señales

7.1.1.1. Sistema Fotovoltaico

El generador Fotovoltaico, objeto de estudio, hace parte del laboratorio de recursos energéticos distribuidos (LabDER) del Instituto de Ingeniería Energética de la Universidad Politécnica de Valencia, los módulos se encuentran instalados en la cubierta de la nave, con orientación sur e inclinados a 30º; el sistema está conformado por 11 módulos conectados en serie a un inversor de tipo "en red" de 2.5k W, (Figura 7.1.1).

La Tabla 7.1.1, resume los valores eléctricos del generador fotovoltaico. En el mismo plano de los paneles, se encuentra instalado un sensor de radiación solar que proporciona además la temperatura de la superficie de los módulos y la temperatura ambiente.

Tabla 7.1.1: Valores eléctricos del generador PVS

Parámetro paneles	Valor
Voc	445,2 V
ISc	4,85 A
PMax	2,125 kW
IMax	4,39 A
VMax	354 V

Figura 7.1.1: Sistema fotovoltaico del LabDER, UPV

7.1.1.2. Adquisición de datos

Tanto el inversor solar como el sensor de radiación están provistos de puertos de comunicaciones ModBus serie, el primero trabaja con RS-232 y el segundo con RS-485; por lo que se han instalado tarjetas transductoras que permiten convertir el protocolo serie a TCP/IP para llevarlos mediante conexiones Ethernet a un PC en donde se han programado las comunicaciones y la adquisición de datos usando LabVIEW. El PC cuenta con dos tarjetas de comunicaciones Ethernet, en la primera se crea la red local LAN, que comunica con todos los equipos del LabDER y la segunda se conecta a la intranet de la UPC y a Internet.

La Figura 7.1.2 muestra el esquema del generador fotovoltaico, incluyendo líneas de comunicación de datos.

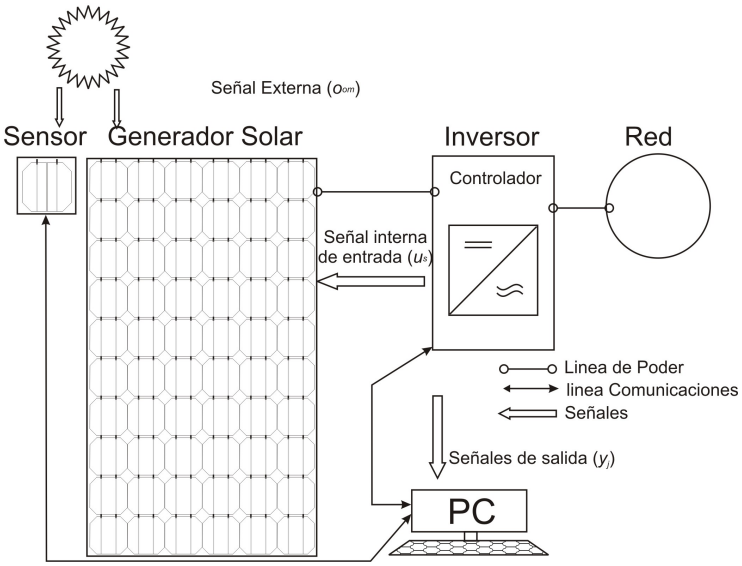


Figura 7.1.2: Esquema generador fotovoltaico

7.1.1.3. Binarización de Señales

Para efectos de identificación y diagnóstico se considera todo el sistema, sin divisiones y las señales a tener un cuenta son:

- La señal de entrada externa es la radiación solar (W/m2).
- La señal de entrada es el voltaje de referencia (V).
- Y las respuestas del sistema son: la potencia DC (W), la corriente (A) y la temperatura superficial (ºC).

Para convertir los valores de las señales a valores binarios se han establecido rangos de operación, de acuerdo al conocimiento de un experto en el sistema. Estos rangos se muestran en la Tabla 7.1.2.

Un ejemplo de un evento de la forma $\omega^i = (u_s y_j)\, .t^{ev}_{o_{om}}$, para el sistema fotovoltaico es $\omega^i = (u_2 y_{584})$,69 lo cual significa que el voltaje de referencia está entre $[360 - 599](u_2 = [00010])$, que la potencia

Tabla 7.1.2: Rangos de operación para PVS

Señal	Rango	Señal	Rango
Radiación Solar (W/m^2)	$[0-20]$		$[1200-1800]$
	$[20-200]$		$[1800-2500]$
	$[200-900]$	Corriente (A)	$[0-0,002]$
	$[>900]$		$[0,002-0,6]$
Voltaje (V)	$[0-180]$		$[0,6-3,8]$
	$[180-300]$		$[3,8-4,4]$
	$[300-360]$	Temperatura de la Superficie de la celda	$[0-16]$
	$[360-599]$		$[16-25]$
	$[\geq 600]$		$[25-60]$
Potencia DC (W)	$[0-600]$		$[>60]$
	$[600-1200]$		

DC está entre $[1200 - 1800]$, la corriente entre $[0,002 - 0,6]$ y la temperatura entre $[0 - 16]$ ($y_{584} = [001001001000]$) y el tiempo entre los eventos ω^{i-1} y ω^{i} es de 69 min.

La función de salida representa un estado del sistema, cuyo comportamiento lo define el estado de cada una de las señales medidas.

La radiación solar es una señal de disturbio del sistema; por lo que se considera una señal de entrada externa. Esta señal define los modos de funcionamiento o_{om} del sistema, por ejemplo $o_2 = [0010]$ significa que la radiación solar se encuentra entre $[200 - 900]$ (W/m2).

7.1.2. Identificación del comportamiento normal

El sistema se observó por varios ciclos de funcionamiento, en un comportamiento normal y la st-IPN identificada se muestra en la Figura 7.1.3.

En la estructura de la st-IPN es posible identificar tres comportamientos diferenciados del PVS. En primer lugar se identifican los estados de inicialización del funcionamiento hasta que se estabiliza, en segundo lugar se muestra el trabajo del sistema en máxima producción y finalmente la reducción de la potencia.

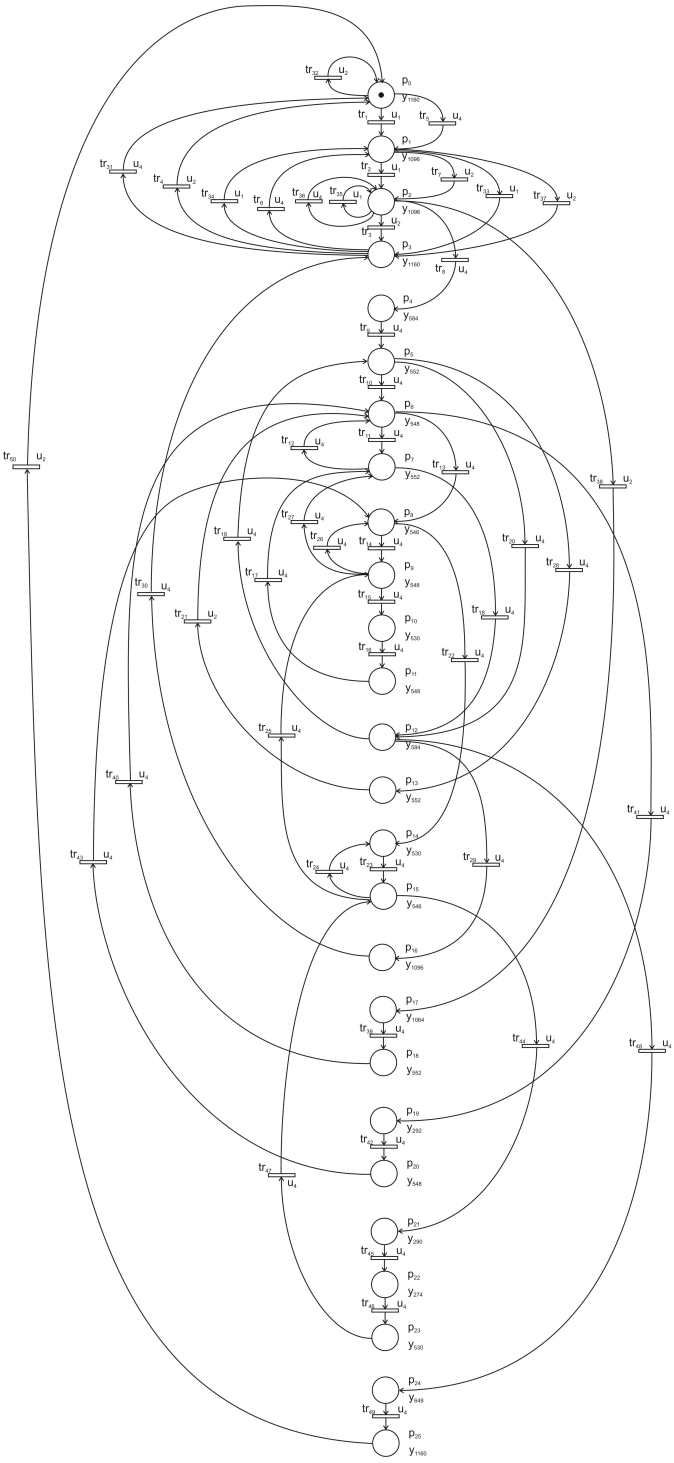


Figura 7.1.3: st-IPN para PVS

La información temporizada de las transiciones, respecto a las funciones de densidad en cada modo de operación se muestra en la Tabla 7.1.3, así como los intervalos de confianza, asumiendo un nivel de confianza del 95 %.

Tabla 7.1.3: Información de la st-IPN para el PVS

λ	φ	Transición		Función de Densidad			
		pre	*post*	o_1	o_2	o_4	o_8
u_1	y_{1096}	0	1	$N\,(17,5;1,2)$	$N\,(56,1;5,2)$		
u_1	y_{1096}	1	2	$N\,(5;0,3)$			
u_2	y_{1160}	2	3	$N\,(70,8;0,1)$	$N\,(4,25;1,25)$		
u_2	y_{1160}	3	0		$N\,(7;1,03)$		
u_4	y_{1096}	0	1		$N(1;0,3)$		
u_4	y_{1096}	3	1		$N\,(3,75;0,82)$		
u_2	y_{1096}	1	2		$N\,(3,6;0,35)$		
u_4	y_{584}	2	4		$N\,(7,5;0,15)$		
u_4	y_{552}	4	5		$N\,(3,4;0,13)$		
u_4	y_{548}	5	6		$N\,(9;1,25)$	$N\,(8;2,05)$	
u_4	y_{552}	6	7		$N\,(12;1,3)$	$N\,(48;5,12)$	
u_4	y_{548}	7	6		$N\,(8,2;2,3)$	$N\,(116;5,8)$	
u_4	y_{546}	6	8			$N\,(24;3,01)$	
u_4	y_{548}	8	9			$N\,(3;0,325)$	
u_4	y_{530}	9	10			$N\,(28;2,04)$	$N\,(10;2,1)$
u_4	y_{548}	10	11			$N\,(20;2,14)$	
u_4	y_{552}	11	7			$N\,(42;2,16)$	$N\,(11;1,3)$
u_4	y_{584}	7	12			$N\,(3,2;0,2)$	
u_4	y_{552}	12	5		$N\,(63;2,03)$	$N\,(7,5;0,18)$	
u_4	y_{552}	5	13		$N\,(42;3,04)$	$N\,(98;4,06)$	
u_2	y_{548}	13	6		$N\,(5;0,2)$	$N\,(3;0,15)$	
u_4	y_{530}	8	14	$N\,(11;1)$		$N\,(3;0,5)$	
u_4	y_{546}	14	15			$N\,(7;0,01)$	

λ	φ	Transición		Función de Densidad			
		pre	*post*	o_1	o_2	o_4	o_8
u_4	y_{530}	15	14		$N\,(9;0,1)$	$N\,(13,5;0,2)$	
u_4	y_{548}	15	9		$N\,(6,5;0,2)$	$N\,(8,5;0,3)$	
u_4	y_{546}	9	8		$N\,(4;0,1)$	$N\,(23;0,02)$	
u_4	y_{552}	9	7			$N\,(2;0,93)$	
u_4	y_{584}	5	12			$N\,(3,3;0,25)$	$N\,(36;0,1)$
u_4	y_{1096}	12	16			$N\,(36,7;3,1)$	
u_4	y_{1160}	16	3			$N\,(15;1,9)$	
u_4	y_{1160}	3	0			$N\,(1;0,6)$	
u_2	y_{1160}	0	0			$N\,(23;2,82)$	
u_1	y_{1160}	1	3			$N(3;0,13)$	
u_1	y_{1096}	3	1	$N(4;0,2)$		$N(5;0,55)$	
u_1	y_{1096}	2	2	$N(3,9;0,1)$	$N(17,3;4,1)$		
u_4	y_{1096}	2	2		$N(44;5,5)$		
u_2	y_{1160}	1	3		$N(63;2,5)$		
u_2	y_{1064}	2	17		$N(23,8;2,03)$		
u_4	y_{552}	17	18		$N(52;1)$		
u_4	y_{548}	18	6		$N(36;2,15)$		
u_4	y_{292}	6	19		$N(17;3,01)$		
u_4	y_{548}	19	20			$N(24,8;2,05)$	
u_4	y_{546}	20	8			$N(36;4,25)$	
u_4	y_{290}	15	21			$N\,(25;1,25)$	
u_4	y_{274}	21	22			$N(33;5,01)$	
u_4	y_{530}	22	23			$N(14,9;2,2)$	
u_4	y_{546}	23	15		$N(10;1,14)$		
u_4	y_{648}	12	24			$N(52,2;2)$	
u_4	y_{1160}	24	25			$N(5,65;0;97)$	
u_2	y_{1160}	25	0		$N\,(75,8;4,12)$		

7.1.3. Proceso de Detección y Localización de Fallos

En aras de explicar el comportamiento de fallo del sistema, a continuación se representa la composición del símbolo de salida.

$$y_j = \Big\{\overbrace{rango_1 rango_2 rango_3 rango_4}^{Señal\_Potencia}\overbrace{rango_1 rango_2 rango_3 rango_4}^{Señal\_Corriente}\overbrace{rango_1 rango_2 rango_3 rango_4}^{Sensor\_Temperatura}\Big\}.$$

A partir de la monitorización del sistema fotovoltaico se han detectado algunos fallos que se relacionan en la Tabla 7.1.4.

7.1.4. Identificación de los Fallos

Al analizar los resultados del proceso de la monitorización del sistema fotovoltaico, se observa que los fallos encontrados no corresponden a daños en los dispositivos, son eventos con un comportamiento no deseado.

En este ejemplo, los fallos f_1 y f_4, se deben a las lecturas de potencia fuera del rango; esto significa que la señal de potencia supera a 2500 vatios. Esto es físicamente imposible debido a que el inversor está limitado a 2,5 kW; por lo que se puede deducir que es un problema de comunicación.

El fallo f_2, es un fallo que muestra un cambio en la tensión de referencia sin que previamente existiese un cambio de potencia; luego f_2 refleja una orden de control errónea creada por el inversor.

En f_3 y f_8, los fallos están relacionados con una señal de corriente menor a la esperada y con señales de voltaje y potencia normales; esto se puede explicar por un problema de comunicación o por un retraso en la actualización de los registros de memoria del inversor.

El fallo f_5, muestra señales de potencia y corriente fuera de rango; dos posibles situaciones pueden explicarlo: la primera es un error de

Tabla 7.1.4: Caracterización de fallos en el PVS

Fallo	τ_i	Lugar	Señal de fallo	Señales esperadas	Señal observada	o_{om}	Señal fallida
f_1	153	4	Salidas	[001000101000]	[000000101000]	o_2	*Potencia*
f_2	304	7	Entrada	[00100]	[01000]	o_4	
f_3	307	7	Salidas	[001000100100] o [001001001000]	[001010001000]	o_4	*Corriente*
f_4	586	12	Salidas	[001000101000] o [010001001000] o [001001001000]	[000001001000]	o_2	*Potencia*
f_5	645	24	Salidas	[010010001000]	[001001001000]	o_1	*Potencia* y *Corriente*
f_6	650	24	Entradas	[00100]	[01000]	o_2	
f_7	652	24	Entradas	[00100]	[01000]	o_2	
f_8	712	24	Salidas	[010010001000]	[010001001000]	o_1	*Corriente*

comunicación en la adquisición de datos, la segunda es la presencia de un sombreado parcial que reduce la radiación solar recibida por el generador fotovoltaico, pero no en el sensor de radiación o al revés, el resultado es en una potencia de salida superior o inferior a la potencia esperada.

Los fallos f_6 y f_7 están relacionados con la tensión, esto ocurren cuando hay cambios repentinos en la radiación solar y el algoritmo de seguimiento del punto de máxima potencia del inversor (Maximun Point Powew Tracker – MPPT) genera un voltaje de consigna que oscila entre máximos y mínimos para buscar el punto óptimo de trabajo y obtener el máximo rendimiento del generador. f_6 y f_7 no son fallos reales, son acciones para recuperar el funcionamiento normal del sistema y que no fueron detectadas en el proceso de identificación.

7.2. Punzonadora

La Figura 7.2.1 muestra una máquina que simula el proceso de punzonado, instalada en el laboratorio de automatización de la UPV.

Su funcionamiento comienza cuando se pulsa el botón de marcha (M) y se activan las fotocélulas. Tras un segundo se puede iniciar el proceso si no hay pieza al inicio o final de la cinta y la máquina se encuentra situada arriba, de lo contrario se ejecutará una rutina de recuperación. En el funcionamiento normal, una vez se detecte una pieza al inicio de la cinta (E1) se selecciona el movimiento de la cinta (M2) hacia la máquina, a los 0.5 segundos se moverá la cinta (M2-ON) llevando la pieza a la máquina. Una vez se detecte que la pieza está debajo de la máquina (E2) la cinta se detiene y se genera la orden de mover la máquina hacia abajo (M1). A los 0.5 segundos baja la máquina (M1-ON) hasta que está llegue a

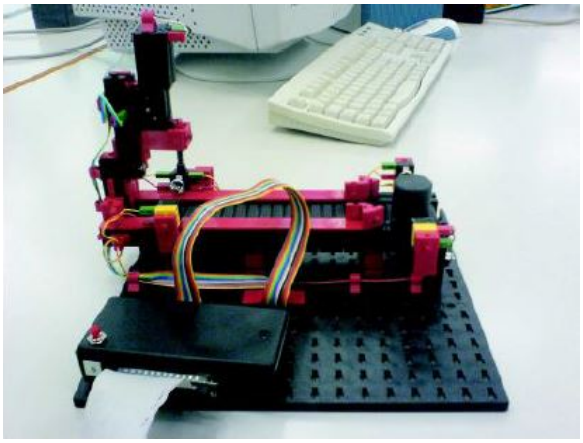


Figura 7.2.1: Punzadora

su final de carrera (E4). A continuación se esperarán 2 segundos con la máquina parada, seleccionando movimiento ascendente de la máquina. Luego se activa el ascenso de la máquina y el movimiento de extracción de la cinta (con la cinta parada). Una vez la máquina llegue a su posición superior se extraerá la pieza de la cinta hasta que ésta alcance el sensor inicial de la cinta (E1).

Cuando el operario retire la pieza el sistema se detendrá, esperando una nueva pulsación de marcha para iniciar el proceso. La rutina de recuperación se ejecutará, si tras pulsar el botón de marcha la máquina no está en posición o hay una pieza en alguna de las posiciones de los sensores. Primero se verifica que la máquina está en su posición, de lo contrario se selecciona el movimiento arriba y a los 0.5 segundos, la máquina se mueve hasta alcanzar su posición superior.

Después se verifica si hay pieza debajo de la máquina, en ese caso se selecciona el movimiento de extracción de la cinta y a los 0.5 segundos se extrae la pieza por la cinta hasta llegar a la posición inicial. En ese momento se espera que el operario retire la pieza. Por

último se verificará si hay pieza al inicio de la cinta, en ese caso se esperará que el operario retire la pieza. Cuando se cumplan todas las condiciones, se puede volver a la situación inicial, de modo que se inicie el sistema tras pulsar el botón de marcha.

Las direcciones de entradas y salidas se muestran en la Tabla 7.2.1.

Tabla 7.2.1: Señales de E/S de la punzonadora

(a) Lecturas sensoriales

Variable	Dirección	Descripción
E0	0.0	Pulsador de marcha
E1	0.1	Pieza al principio de la cinta
E2	0.2	Pieza al final de la cinta
E3	0.3	Máquina arriba
E4	0.4	Máquina abajo
EM	0.5	Interruptor de emergencia

(b) Órdenes de Control

Variable	Dirección	Descripción
M3	100.4	Activar células
M2-ON	100.1	Activar cinta
M2	100.0	Dirección cinta (0=atrás, 1=adelante)
M1-ON	100.3	Activar máquina
M1	100.2	Dirección máquina (0=atrás, 1=adelante)

7.2.1. Comportamiento Normal Identificado

Como señales de entrada externas se consideran el pulsador de marcha y el interruptor de emergencia, estas señales determinan el modo de funcionamiento del proceso.

Las entradas del sistema son las órdenes de control: Activar células, activar cinta, dirección de cinta, activar máquina punzonadora y dirección de máquina.

Las salidas son las lecturas de los sensores E1-E4, mostrados en la Tabla 7.2.1a.

La st-IPN que representa el comportamiento normal se muestra en la Figura 7.2.2.

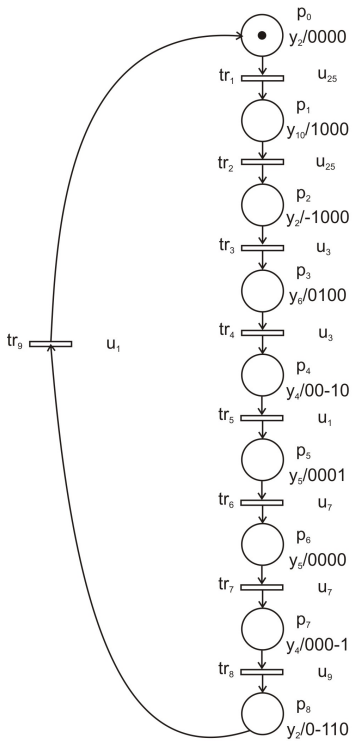


Figura 7.2.2: st-IPN Punzonadora

El comportamiento temporizado de las transiciones, para un nivel de confiabilidad del 95 %, se puede observar en la Tabla 7.2.2. Solo

se tiene un modo de funcionamiento: cuando el pulsador está en marcha (o_1).

Tabla 7.2.2: Información temporizada del modelo de la punzonadora

Transición	Función de densidad	Intervalo de Confianza
tr_1	$N(4,33;1,527)$	$[7,327-3,303]$
tr_2	$N(1,33;0,577)$	$[2,464-0,943]$
tr_3	$N(15,5;0,707)$	$[12,023-13,881]$
tr_4	1	$[1-1]$
tr_5	$N(2;1,412)$	$[1,046-4,771]$
tr_6	$N(2,5;0,707)$	$[2,023-3,885]$
tr_7	1	$[1-1]$
tr_8	$N(6;1,53)$	$[4,968-8,998]$
tr_9	$N(14;1,414)$	$[13,046-16,771]$

7.2.2. Detección y Aislamiento de Fallos

Induciendo fallos en el proceso y monitorizando el proceso on-line el diagnosticador resultante se muestra en la Figura 7.2.3 y la caracterización de los fallos en la Tabla 7.2.3

Tabla 7.2.3: Localización de fallos en la punzonadora

Fallo	τ_i	Lugar	Señal de fallo	Sr esperadas	Sr observadas	Sensor fallido
f_1	35	7	Salidas	[0010]	[0110]	s_2
f_2	49	8	Salidas	[0010]	[1010]	s_1
f_3	92	8	Entradas	[01010]	[01000]	
f_4	96	8	Entradas	[01010]	[01011]	

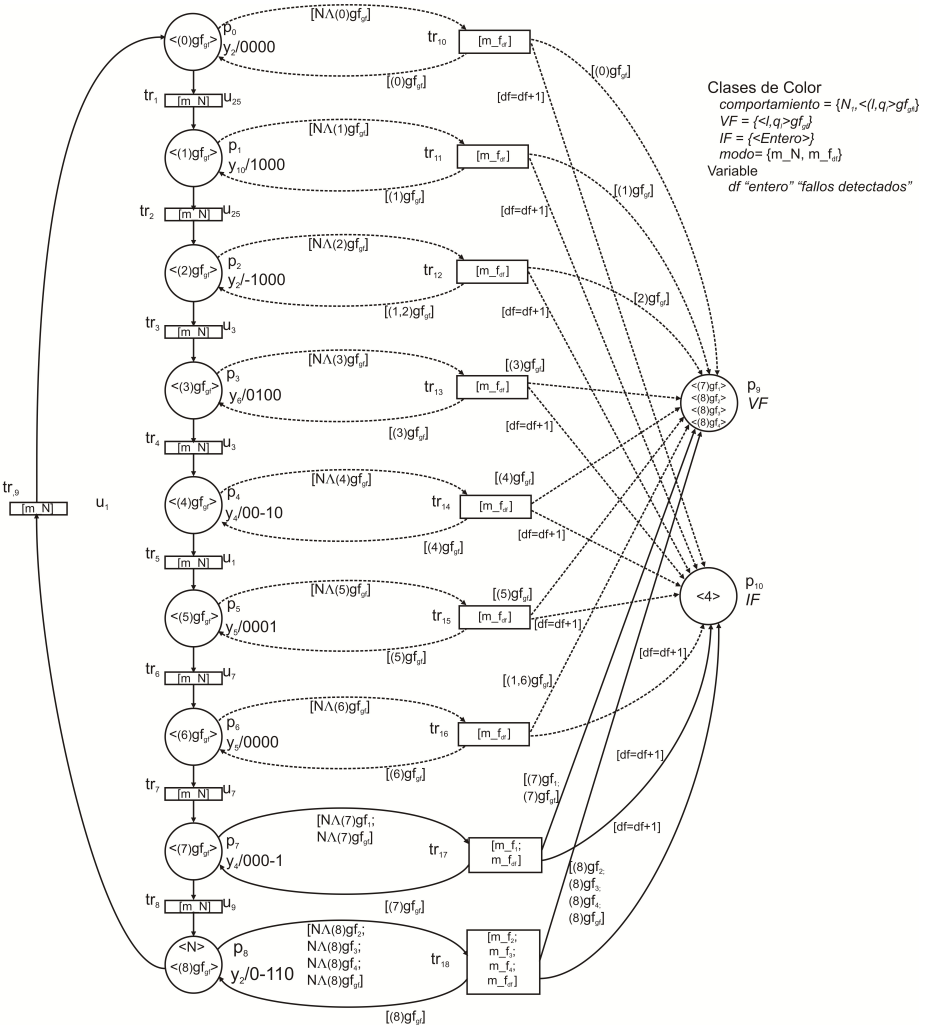


Figura 7.2.3: st-DICPN para punzonadora

7.2.3. Identificación de los Fallos

Estando en el lugar 7 se da la orden de dirigir la cinta hacia atrás; pero el sensor 2 detecta pieza al final y ocurre el fallo 1 ($(1, 7)\, f_1$); después de un tiempo se mantiene la misma orden y la pieza es retirada; la st-DICPN se mantiene en el lugar 7 con una marca normal y de fallo; por lo tanto puede evolucionar al lugar 8 cuando la pieza es retirada, estando en el lugar 8 el sistema debe regresar a su estado inicial; pero ya existe detección de pieza al inicio y el controlador sigue emitiendo órdenes asumiendo lecturas normales, por lo tanto el sistema no regresa a un estado normal hasta que el fallo en el sensor haya sido reparado.

7.3. Proceso Neumático

Se implementó un sistema en el laboratorio, el cual dispone de una placa con 6 cilindros neumáticos como la que se muestra en la Figura 7.3.1.

El esquema de los dispositivos accesibles desde el PLC se muestra en la Figura 7.3.2.

La placa dispone de dos cilindros monoestables (CA y CF), así como de cuatro biestables (CB, CC, CD y CE). Para esta práctica se utilizan tres biestables. Además dispone de un panel de mando que incluye dos botones (botón verde (BV) y botón negro o rojo (BN)), un selector de dos posiciones (S2), un selector de tres posiciones (S3), dos luces (luz verde (LV) y luz roja (LR)) y una seta de emergencia. Las direcciones de las variables en el PLC se muestran en la Tabla 7.3.1.

El proceso implementado consta de tres operaciones: A, B y C; utilizando los cilindros CB, CD y CE. Las operaciones consisten en:

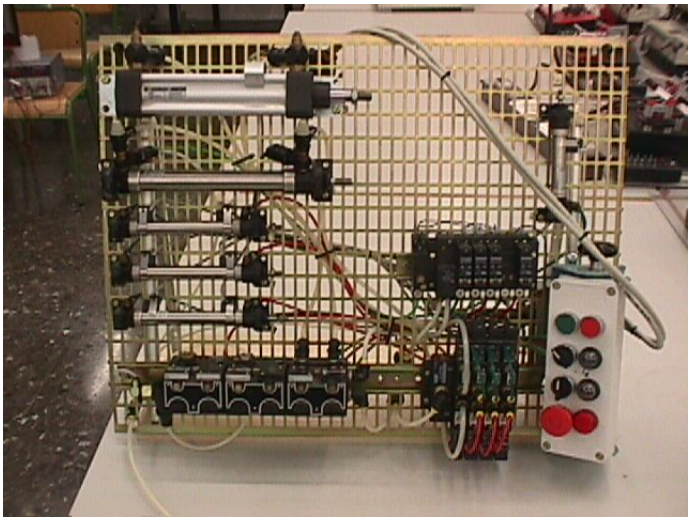


Figura 7.3.1: Proceso neumático

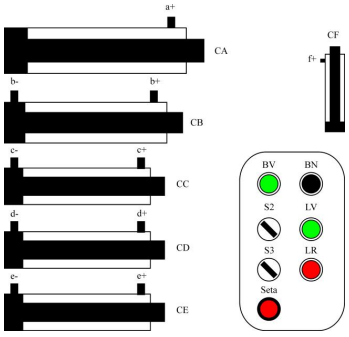


Figura 7.3.2: Esquema dispositivos del proceso neumático

Operación A (Botón S3 a la izquierda): Se extrae CB, cuando llegue al final extraer CD y a su vez cuando CD llegue al final, extraer CE y luego replegarlos en orden inverso: primero CE, luego CD y al final CB. Esta secuencia se repite tres veces seguidas. Una vez finalizado, se espera 1 segundo para completar la operación.

Operación B (Botón S3 al centro): Se extraen simultáneamente los tres cilindros, se espera a sincronizarlos todos en sus finales de carrera y luego retraerlos y volver a sincronizarlos en sus finales de carrera. Esta secuencia se debe repetir cuatro veces seguidas. Una vez finalizado, se esperan 2 segundos para completar la operación.

Operación C (Botón S3 a la derecha): Se extraen CB y CE, una vez hayan alcanzado ambos sus finales de carrera se extrae CD, cuando éste llegue a su final de carrera, se retraerá CD hasta el final de carrera y volverá a extraerse. Una vez llegue al final se retraerán los tres cilindros. Una vez finalizado, se espera 1 segundo para completar la operación.

La máquina funciona en dos modos de operación que dependen de la posición del botón S2.

Modo manual (S2 a la derecha): Se selecciona la operación a realizar y se pulsa la marcha.

Modo automático (S2 a la izquierda): Al pulsar la marcha se efectúa la operación A, luego la B y a continuación la C. La máquina sigue repitiendo el ciclo hasta que S2 cambie de posición, en ese momento, terminará la operación que esté realizando y volverá al estado de reposo. La máquina no entrará en ninguno de los modos hasta que no se pulse marcha.

7.3.1. Comportamiento Identificado

El proceso se ha implementado de la siguiente manera: En primer lugar, se eligió el modo manual y la máquina trabajó en cada

Tabla 7.3.1: Señales de E/S del proceso neumático

(a) Lecturas sensoriales

Variable	Dirección	Descripción
BV	0.0	1 indica botón pulsado
BR	0.1	0 indica botón pulsado
S2D	0.2	1 indica selector a la derecha
S3I	0.3	1 indica selector a la izquierda
S3D	0.4	1 indica selector a la derecha
Seta	0.5	Seta accionada, corta flujo de aire comprimido
a+	0.6	CA extraído completamente
b+	0.7	CB extraído completamente
b-	0.8	CB retraído completamente
c+	0.9	CC extraído completamente
c-	0.10	CC retraído completamente
d+	0.11	CD extraído completamente
d-	0.12	CD retraído completamente
e+	0.13	CE extraído completamente
e-	0.14	CE retraído completamente
f+	0.15	CF extraído completamente

(b) Órdenes de control

Variable	Dirección	Descripción
CA+	100.0	Extraer CA
CB+	100.1	Extraer CB
CB-	100.2	Retraer CB
CC+	100.3	Extraer CC
CC-	100.4	Retraer CC
CD+	100.5	Extraer CD
CD-	100.6	Retraer CD
CE+	100.7	Extraer CE
CE-	100.8	Retraer CE
CF+	100.9	Extraer CF
LV	100.10	Encender luz
LR	100.11	Encender luz

una de las operaciones hasta que cada una de ellas fue totalmente identificada.

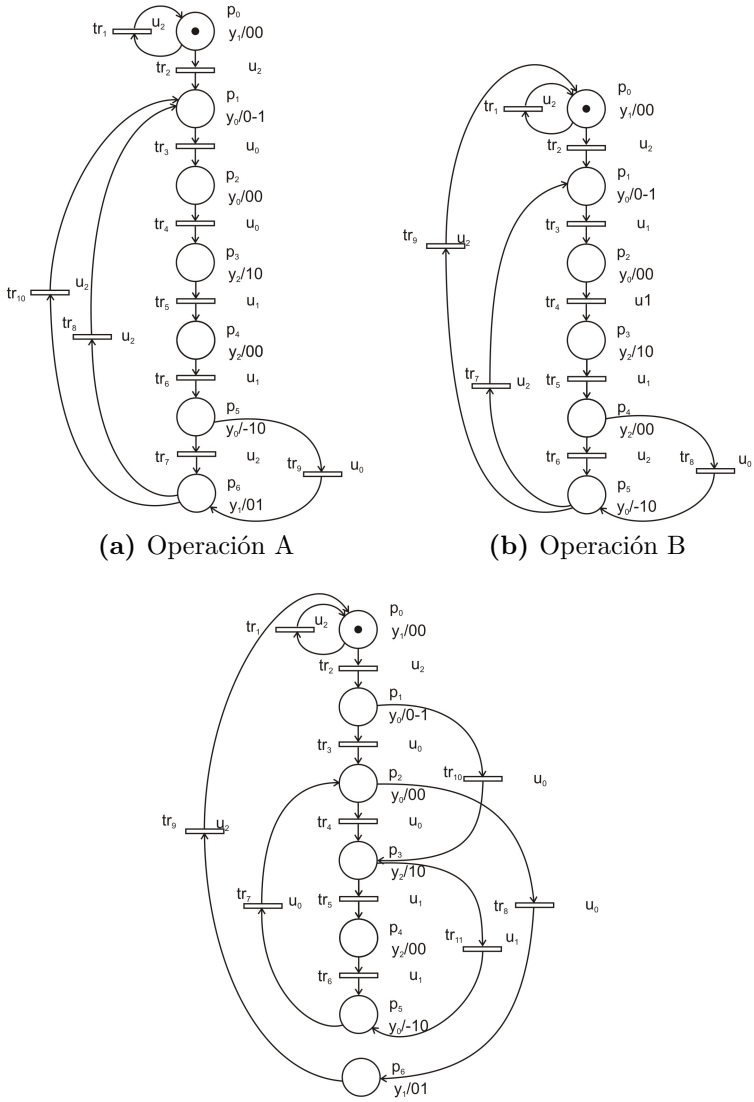
Para aplicar el algoritmo de identificación, el sistema se dividió en subsistemas, siendo cada cilindro un subsistema. Los modelos identificados de cada uno de los cilindros en cada operación se muestran a continuación.

7.3.1.1. Cilindro B

La información temporizada para el cilindro CB se muestra en la Tabla 7.3.2.

Tabla 7.3.2: Información temporizada cilindro CB

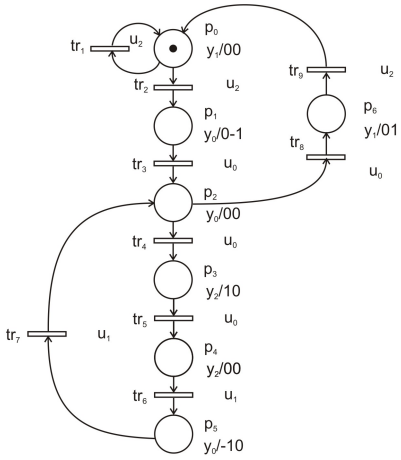
Transición	Intervalo de Confianza. Operación A.	Intervalo de Confianza. Operación B.	Intervalo de Confianza. Operación C.
tr_1	$[69 - 69]$	$[19 - 19]$	$[18 - 18]$
tr_2	$[1 - 1]$	$[1 - 1]$	$[1 - 1]$
tr_3	$[5,67 - 7,64]$	$[6,27 - 7,42]$	$[6 - 6]$
tr_4	$[1,07 - 2,24]$	$[1 - 1]$	$[1 - 1]$
tr_5	$[28,15 - 29,34]$	$[1 - 1]$	$[19,21 - 20,45]$
tr_6	$[1,45 - 2,47]$	$[15,90 - 16,73]$	$[1 - 1]$
tr_7	$[16,64 - 17,43]$	$[1 - 1]$	$[6,05 - 7,24]$
tr_8	$[1,36 - 2,47]$	$[15,88 - 16,76]$	$[11,68 - 12,77]$
tr_9	$[16,27 - 17,47]$	$[16,66 - 21,83]$	$[14,79 - 20,83]$
tr_{10}	$[16,23 - 32,63]$		$[7 - 7]$
tr_{11}			$[19,27 - 20,61]$



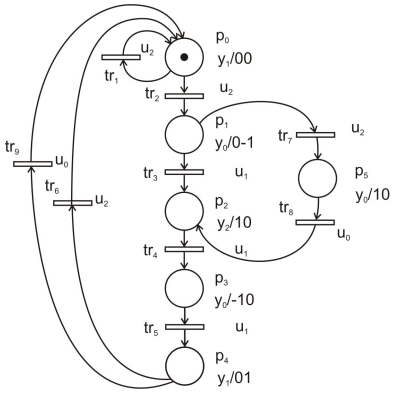
(a) Operación A **(b)** Operación B

(c) Operación C

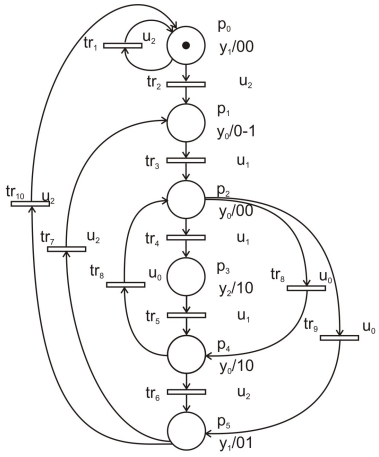
Figura 7.3.3: st-IPN del Cilindro CB



(a) Operación A



(b) Operación B



(c) Operación C

Figura 7.3.4: st-IPN del cilindro CD

7.3.1.2. Cilindro D

La información temporizada para el cilindro CD se muestra en la Tabla 7.3.3.

Tabla 7.3.3: Información temporizada cilindro CD

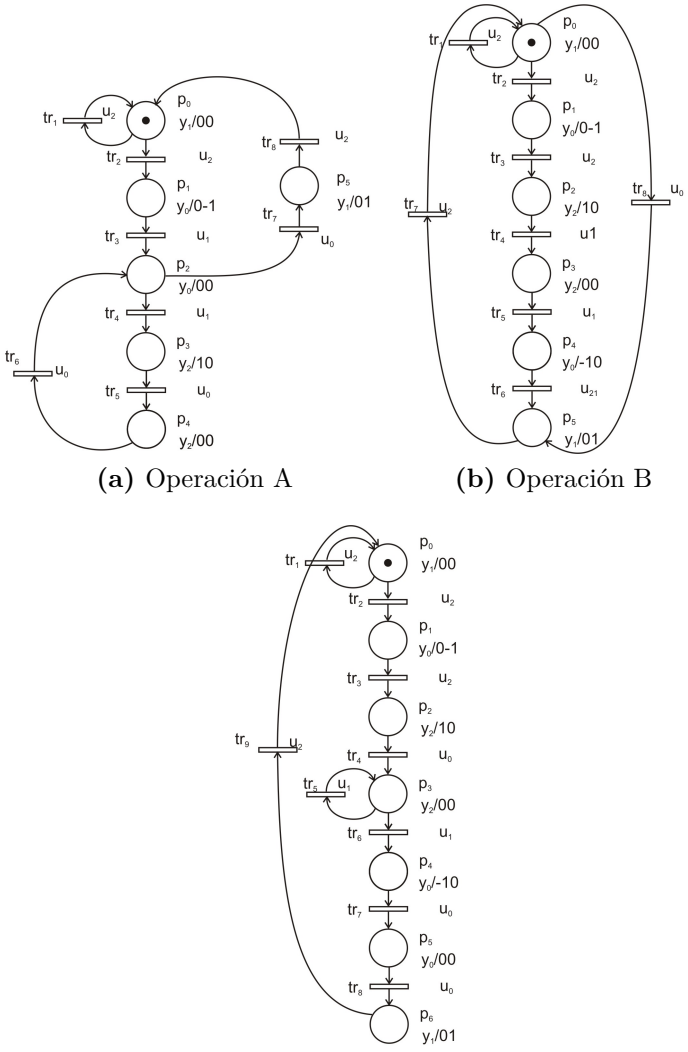
Transición	Intervalo de Confianza. Operación A.	Intervalo de Confianza. Operación B.	Intervalo de Confianza. Operación C.
tr_1	$[77 - 77]$	$[16,89 - 21,67]$	$[25 - 25]$
tr_2	$[4,07 - 5,24]$	$[4,19 - 5,37]$	$[3,32 - 4,48]$
tr_3	$[1,89 - 2,68]$	$[2,92 - 3,94]$	$[2,88 - 3,58]$
tr_4	$[1,11 - 2,29]$	$[4,35 - 5,59]$	$[1 - 1]$
tr_5	$[14,08 - 15,12]$	$[3,90 - 4,72]$	$[3,05 - 4,21]$
tr_6	$[4,95 - 5,97]$	$[7,50 - 10,03]$	$[3,48 - 4,50]$
tr_7	$[2,07 - 3,24]$	$[2,38 - 5,57]$	$[2,32 - 3,48]$
tr_8	$[1,15; 2,34]$	$[1 - 1]$	$[2,66 - 3,43]$
tr_9	$[23,70 - 2,34]$	$[8,60 - 9,47]$	$[1,66 - 2,82]$
tr_{10}			$[31,95 - 38,56]$
tr_{11}			$[3,06 - 4,30]$

7.3.1.3. Cilindro E

La información temporizada para el cilindro CE se muestra en la Tabla 7.3.4.

7.3.2. Detección y Aislamiento de Fallos

Induciendo fallos en el proceso y monitorizando el proceso on-line, el diagnosticador resultante se muestra en la Tabla 7.3.5.



(a) Operación A **(b)** Operación B

(c) Operación C

Figura 7.3.5: st-IPN del Cilindro CE

Tabla 7.3.4: Información temporizada cilindro CE

Transición	Intervalo de Confianza. Operación A.	Intervalo de Confianza. Operación B.	Intervalo de Confianza. Operación C.
tr_1	$[83 - 83]$	$[16,89 - 21,67]$	$[18 - 18]$
tr_2	$[2,57 - 3,45]$	$[2,68 - 3,41]$	$[1,66 - 2,43]$
tr_3	$[0,92 - 1,89]$	$[2,55 - 3,46]$	$[2,32 - 3,48]$
tr_4	$[1,15 - 2,34]$	$[1,68 - 2,41]$	$[2,32 - 3,48]$
tr_5	$[2,34 - 3,45]$	$[4,83 - 5,27]$	$[19,99 - 21,55]$
tr_6	$[7,24 - 8,41]$	$[8,90 - 9,72]$	$[4,32 - 5,48]$
tr_7	$[1,07 - 2,24]$	$[3,78 - 4,67]$	$[2,08 - 3,31]$
tr_8	$[38,17 - 62,85]$	$[4 - 4]$	$[6,88 - 7,65]$
tr_9			$[19,75 - 26,18]$

Tabla 7.3.5: Localización de fallos en el proceso neumático

Subsistema	Fallo	τ_i	Modo	Lugar	Señal de fallo	Sr esperadas	Sr observadas	Sensor fallido
3	f_1	53	A	5	Salidas	[01]	[10]	$s_{3,1}, s_{3,2}$
1	f_1	69	B	3	Salidas	[10]	[01]	$s_{1,1}, s_{1,2}$
2	f_1	38	A	4	Salidas	[00]	[01]	$s_{2,2}$

7.4. Conclusiones del capítulo

En este capítulo se ha probado la eficacia del método de diagnóstico propuesto en sistema reales, tanto en procesos sencillos como en procesos con más dispositivos.

Como recomendación en la aplicación a sistemas reales cuyos procesos sean muy rápidos, es el ajuste del tiempo de adquisición de datos para evitar la no lectura de señales.

8 Conclusiones y Trabajos Futuros

En esta tesis se ha tratado el tema de diagnóstico de fallos en Sistemas de Eventos Discretos de naturaleza estocástica, sin modelo previo. El sistema a considerar, puede ser descrito a partir de la interrelación de señales tanto internas como externas, las cuáles se convierten en eventos; por lo tanto, el comportamiento del sistema puede ser modelado con base en su lenguaje regular. Además se tiene en cuenta en el proceso de modelado, las señales externas que afectan al controlador, convirtiéndolas en eventos externos que imponen modos de funcionamiento del sistema. La tarea de diagnóstico sin modelo previo del sistema, se lleva a cabo comparando el lenguaje observado con el lenguaje previamente identificado en comportamiento normal del sistema.

Se ha propuesto e implementado un generador determinístico de lenguaje que represente el lenguaje observado del sistema, basado en redes de Petri temporizadas, con unas características particulares en cuanto a que: La st-IPN es determinística, ya que la forma en que se definen sus estados evita que el sistema pueda evolucionar a dos estados diferentes a partir de un mismo evento. La st-IPN representa solo el lenguaje observado; es decir, la red no genera cadenas de eventos que no hayan sido observadas, característica importante ya que en el proceso de diagnóstico las cadenas no observadas; pero representadas en la red pueden ser cadenas de evento de fallo que ya no pueden ser detectadas, puesto que harían

parte del lenguaje normal. La aplicabilidad del proceso de identificación es a cualquier tipo de sistema en el cuál se puedan medir las señales on-line u off-line, independiente del tamaño ya que además contempla la variabilidad del tiempo de los procesos.

El proceso de identificación, termina en el momento en que se cumplan las condiciones de representatividad estadística del tamaño de los datos observados; esta condición permite establecer los intervalos de confianza de las transiciones involucradas.

Una vez el comportamiento normal del sistema ha sido identificado, la estructura del generador es transformada para la tarea de diagnóstico, de tal manera que sea variable y pueda crecer y aprender en la medida que diagnostica fallos. Para alcanzar este objetivo, primero se define una red de Petri coloreada que modele sistemas a partir de las señales de entrada /salida y que a su vez permita adicionar etiquetas en las transiciones y en los lugares, con base en esta red de Petri temporizada, coloreada e interpretada, st-ICPN se hace una propuesta de diagnosticador (st-DICPN), en el cual las entradas a las matrices de incidencias crecen a medida que se requiere.

Una característica importante del diagnosticador es su estructura variable y su habilidad para aprender lenguajes de fallo. Esto lo realiza cada vez que se actualiza con un nuevo fallo. La actualización del diagnosticador consiste en adicionar en su estructura las marcas de fallo generadas, así como las transiciones y arcos disparados en el proceso de detección del fallo; de esta manera el generador aprende la secuencia de eventos que generó el fallo.

El método de diagnóstico propuesto detecta fallos que generan comportamientos no deseados en las lecturas posteriores de las señales, así como en la alteración del tiempo de los eventos. Una vez un fallo ha sido detectado, se verifica si el fallo se debe a un fallo de propagación, si es de propagación el fallo es eliminado de lo

contrario el fallo es aislado, este proceso lo realiza el algoritmo de filtrado propuesto. La división del sistema en subsistemas permite la localización y caracterización del fallo, brindando herramientas para que los expertos en el sistema puedan determinar el nivel de criticidad del fallo y tomen las medidas de mantenimiento preventivo y correctivo del sistema, completando de esta manera las etapas de detección, aislamiento e identificación del fallo, para lograr el objetivo final que es el diagnóstico.

Una forma de medir la eficiencia del método de identificación y del método de diagnóstico es a partir diferentes test. Se presentan y verifican las condiciones: suficiente y necesaria de identificabilidad y de detectabilidad basada en la teoría de lenguajes, a partir de una propuesta de operación de proyección de lenguajes temporizados compuestos. En particular se prueba que el lenguaje de una st-IPN es determinísticamente identificable y que el diagnosticador, st-DICPN, permite detectar eventos inobservables, bajo ciertas condiciones de las señales.

Se puede concluir que el método de diagnóstico propuesto en esta tesis detecta y aísla cualquier tipo de fallo que genere comportamientos no deseados, en sistemas de eventos discretos que pueden ser modelados a partir de lenguajes cuyos símbolos son la interrelación de señales. El diagnosticador que representa el lenguaje es un generador determinista, con estructura variable, con capacidad de aprender lenguajes de fallo y puede ser implementado en sistemas reales; basado solo en señales observadas del sistema.

El método de diagnóstico fue implementado en procesos de laboratorio y se encontraron modelos coherentes con los sistemas, además se logró detectar y aislar fallos inducidos por interrupción de señales o alteraciones del tiempo, comprobando la eficacia del método. También se comprobó la eficacia del método en procesos más complejos como lo es el sistema fotovoltaico.

Trabajos futuros

Los resultados alcanzados en la presente investigación abren enfoques de investigación en el diagnóstico de fallos sin modelo previo en SED estocásticos.

El presente método verificó su aplicabilidad en diferentes tipos de sistemas, manteniendo un enfoque centralizado; un trabajo futuro está relacionado con la extensión del proceso en entornos distribuidos para la identificación y diagnóstico de fallos; así como en un entorno descentralizado cuando los eventos llegan al sistema de manera asíncrona pero sin un orden definido en una línea de tiempo.

Otra línea de investigación está relacionada con la aplicación del método de identificación propuesto, para analizar el comportamiento del sistema cuando se altera el número de sensores, con el objeto de establecer cuál podría ser el número óptimo de sensores que debe tener un sistema para que sea identificado correctamente y determinar qué tipo de red de Petri se identifica; si es igual o es diferente, qué tanto diferente? a la st-IPN propuesta en esta investigación.

Aplicabilidad del modelado de sistemas, bajo st-IPN, con el objeto de verificar características de funcionamiento o propiedades estructurales de la red modelada del sistema.

Respecto al comportamiento estadístico del modelo se puede analizar el tiempo de disparo de transiciones, cuando existe traslape en los rangos del tiempo con el fin de evitar que transiciones habilitadas se disparen en un orden no adecuado.

Con base en el algoritmo de filtrado y en sistemas que posean sensores de flujo compartido, diseñar un modelo degradado, es decir un modelo que aísle los sensores que generan fallos de propagación y se pueda continuar con el proceso de diagnóstico; posiblemente, fusionando lugares.

Diseño de un dispositivo embebido a partir de las señales de entrada - salida, consideradas en el algoritmo de diagnóstico propuesto en el presente trabajo; que permita detectar fallos on-line en un SED.

Bibliografía

[1] M. P. Fanti, A. M. Mangini, and W. Ukovich, "Fault detection by labeled petri nets in centralized and distributed approaches," *Automation Science and Engineering, IEEE Transactions on*, vol. PP, no. 99, p. 1, 2012.

[2] S. Klein, L. Litz, and J.-J. Lesage, "Fault detection of discrete event systems using an identification approach," in *Proc. 16th IFAC World Congr*, 2005.

[3] M. Roth, J.-J. Lesage, and L. Litz, "The concept of residuals for fault localization in discrete event systems," *Control Engineering Practice*, vol. 19, no. 9, pp. 978 – 988, 2011, special Section: DCDS'09 - The 2nd IFAC Workshop on Dependable Control of Discrete Systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0967066111000396

[4] J. Zaytoon and S. Lafortune, "Overview of fault diagnosis methods for discrete event systems," *Annual Reviews in Control*, vol. 37, no. 2, pp. 308 – 320, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1367578813000552

[5] Papadopoulos and McDermid, "Automated safety monitoring: A review and classification of methods," *International Journal of Condition Monitoring and Diagnostic Engineering Management, COMADEM International-UK*, vol. 4, no. 4, pp. 14–32, 2001.

[6] D. Pandalai and L. Holloway, "Template languages for fault monitoring of timed discrete event processes," *Automatic Control, IEEE Transactions on*, vol. 45, no. 5, pp. 868–882, May 2000.

[7] M. Sayed-Mouchaweh, "Decentralized fault free model approach for fault detection and isolation of discrete event systems," *European Journal of Control*, vol. 18, no. 1, pp. 82 – 93, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0947358012705290

[8] M. Roth, "Identification and fault diagnosis of industrial closed-loop discrete event system," Ph.D. dissertation, DE L'ÉCOLE NORMALE SUPÉRIEURE DE CACHAN (France) et DE TECHNISCHE UNIVERSITÄT KAISERSLAUTERN (Allemagne), 2010.

[9] M. A. F. J. J. C. J. Guasch Petit, Antonio. Piera, *Modelado y Simulación: Aplicación a procesos logísticos de fabricación y servicios*, 2nd ed., U. P. de Catalunya. Iniciativa Digital Politecnica., Ed., 2003.

[10] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, S. S. Media, Ed., 2008.

[11] P. Ramadge and W. Wonham, "The control of discrete event systems," vol. 77, no. 1, jan 1989, pp. 81 –98.

[12] A. Giua, "Supervisory control of petri nets with language specifications," in *Control of Discrete-Event Systems*, ser. Lecture Notes in Control and Information Sciences, C. Seatzu, M. Silva, and J. H. van Schuppen, Eds., vol. 433. Springer London, 2013, pp. 235–255. [Online]. Available: http://dx.doi.org/10.1007/978-1-4471-4276-8_12

[13] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, "Failure diagnosis using discrete-event models," vol. 4, no. 2, mar 1996, pp. 105 –124.

[14] A. Correcher Salvador, "Diagnóstico de fallos intermitentes en procesos industriales basado en modelos de eventos discretos," Ph.D. dissertation, Universidad Politécnica de Valencia, 2005.

[15] F. Cassez and S. Tripakis, "Fault diagnosis with dynamic observers," in *Discrete Event Systems, 2008. WODES 2008. 9th International Workshop on*, may 2008, pp. 212 –217.

[16] V. Sicco, M. d. W. Mathijs, and W. Cees, "Efficiently learning simple timed automata," in *Induction of Process Models*. University of Antwerp, 2008, pp. 61–68.

[17] C. Girault and R. Valk, *Petri nets for systems engineering - a guide to modeling, verification, and applications*. Springer, 2003.

[18] M. Silva and L. Recalde, "Redes de petri continuas: Expresividad, análisis y control de una clase de sistemas lineales conmutados," *Revista Iberoamericana de Automática e Informática Industrial*, vol. 04, no. 03, pp. 5–33, 2007.

[19] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541 –580, apr 1989.

[20] M. Dotoli, M. Fanti, and A. M. Mangini, "Real time identification of discrete event systems using petri nets," vol. 44. Tarrytown, NY, USA: Pergamon Press, Inc., May 2008, pp. 1209–1219.

[21] A. Ramirez-Treviño, E. Ruiz-Beltran, J. Aramburo-Lizarraga, and E. Lopez-Mellado, "Structural diagnosability of des and design of reduced petri net diagnosers," vol. 42, no. 2, march 2012, pp. 416 –429.

[22] K. Hernández and M. Meda-Campana, "Fault diagnosis using petri nets. a case study," *Proceedings of the 10th Latin Ame-*

rican and Caribbean Conference for Engineering and Technology, July 2012.

[23] S. Gaubert and A. Giua, "Deterministic weak-and-marked petri net languages are regular," *Automatic Control, IEEE Transactions on*, vol. 41, no. 12, pp. 1802–1803, Dec 1996.

[24] A. Ramirez-Treviño, E. Ruiz-Beltran, I. Rivera-Rangel, and E. Lopez-Mellado, "Online fault diagnosis of discrete event systems. a petri net-based approach," vol. 4, no. 1, jan. 2007, pp. 31 –39.

[25] C. Girault and R. Valk, *Petri Nets for System Engineering: A Guide to Modeling, Verification, and Applications*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2001.

[26] J. E. Bartlett, J. W. Kotrlik, and C. C. Higgins, "Organizational research: Determining appropriate sample size in survey research," *Information Technology, Learning and Performance*, vol. 19, no. 1, pp. 43–50, 2001.

[27] H. Toutenburg, *Statistical Methods for Rates and Proportions*. WILEY-VCH Verlag, 1974, vol. 16, no. 8. [Online]. Available: http://dx.doi.org/10.1002/bimj.19740160814

[28] E. H. Livingston and L. Cassidy, "Statistical power and estimation of the number of required subjects for a study based on the t-test: A surgeon's primer," *Journal of Surgical Research*, vol. 126, no. 2, pp. 149 – 159, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0022480404007279

[29] W. G. Cochran, *Sampling Techniques*, 2nd ed. New York, NY: John Wiley and Sons, 1977.

[30] J. Lunze, "Qualitative modelling of dynamical systems: Motivation, methods, and prospective applications," *Mathematics*

and Computers in Simulation, vol. 46, no. 6, pp. 465 – 483, 1998.

[31] A. Ichikawa and K. Hiraishi, "Analysis and control of discrete event systems represented by petri nets," in *Discrete Event Systems: Models and Applications*, ser. Lecture Notes in Control and Information Sciences, P. Varaiya and A. Kurzhanski, Eds. Springer Berlin Heidelberg, 1988, vol. 103, pp. 115–134. [Online]. Available: http://dx.doi.org/10.1007/BFb0042308

[32] A. Fanni and A. Giua, "Discrete event representation of qualitative models using petri nets," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 28, no. 6, pp. 770–780, Dec 1998.

[33] H. Hu, M. C. Zhou, Z. Li, and Y. Tang, "An optimization approach to improved petri net controller design for automated manufacturing systems," *Automation Science and Engineering, IEEE Transactions on*, vol. PP, no. 99, p. 1, 2012.

[34] J. Ashley and L. Holloway, "Qualitative diagnosis of condition systems," *Discrete Event Dynamic Systems*, vol. 14, no. 4, pp. 395–412, 2004.

[35] L. E. Holloway, X. Guan, R. Sundaravadivelu, and J. Ashley, Jr., "Automated synthesis and composition of taskblocks for control of manufacturing systems," *Trans. Sys. Man Cyber. Part B*, vol. 30, no. 5, pp. 696–712, Oct. 2000.

[36] R. Sreenivas and B. Krogh, "Petri net based models for condition/event systems," in *American Control Conference, 1991*, 1991, pp. 2899–2904.

[37] S. Patil, V. Vyatkin, and M. Sorouri, "Formal verification of intelligent mechatronic systems with decentralized control logic," in *Emerging Technologies Factory Automation (ETFA), 2012 IEEE 17th Conference on*, 2012, pp. 1–7.

[38] J. Ashley, "Diagnosis of condition systems," Ph.D. dissertation, University of Kentucky, 2004.

[39] P. Merlin and D. J. Farber, "Recoverability of communication protocols -implications of a theoretical study," *Communications, IEEE Transactions on*, vol. 24, no. 9, pp. 1036–1043, Sep 1976.

[40] B. Berthomieu and M. Diaz, "Modeling and verification of time dependent systems using time petri nets," *Software Engineering, IEEE Transactions on*, vol. 17, no. 3, pp. 259–273, Mar 1991.

[41] B. Berthomieu, P.-O. Ribet, and F. Vernadat, "The tool tina - construction of abstract state spaces for petri nets and time petri nets," *International Journal of Production Research*, vol. 42, no. 14, pp. 2741–2756, 2004. [Online]. Available: $http://www.tandfonline.com/doi/abs/10.1080/00207540412331312688$

[42] G. Gardey, D. Lime, M. Magnin, and O. (h. Roux, "Roméo: A tool for analyzing time petri nets," in *In Proc. CAV'05, vol. 3576 of LNCS*. Springer, 2005, pp. 418–423.

[43] B. Berthomieu, F. Peres, and F. Vernadat, "Bridging the gap between timed automata and bounded time petri nets," in *Formal Modeling and Analysis of Timed Systems*, ser. Lecture Notes in Computer Science, E. Asarin and P. Bouyer, Eds. Springer Berlin Heidelberg, 2006, vol. 4202, pp. 82–97. [Online]. Available: $http://dx.doi.org/10.1007/11867340_7$

[44] F. Peres, B. Berthomieu, and F. Vernadat, "On the composition of time petri nets," *Discrete Event Dynamic Systems*, vol. 21, no. 3, pp. 395–424, Sep. 2011.

[45] H. Boucheneb and R. Hadjidj, "Model checking for time petri nets," *Theoretical Computer Science*, vol. 353, no. 13,

pp. 208 – 227, 2006. [Online]. Available: $http://www.sciencedirect.com/science/article/pii/s0304397505007875$

[46] L. Salum, "Petri nets and time modelling," *The International Journal of Advanced Manufacturing Technology*, vol. 38, no. 3-4, pp. 377–382, 2008. [Online]. Available: $http://dx.doi.org/10.1007/s00170-007-1098-5$

[47] R. Kumar and L. Holloway, "Supervisory control of deterministic petri nets with regular specification languages," *Automatic Control, IEEE Transactions on*, vol. 41, no. 2, pp. 245–249, Feb 1996.

[48] R. Valk and G. Vidal-Naquet, "Petri nets and regular languages," *Journal of Computer and System Sciences*, vol. 23, no. 3, pp. 299 – 325, 1981.

[49] D. Nakamura, Y. Takeda, H. Murakoshi, N. Funakubo, and Y. Dohi, "A modeling language for petri net based factory automation systems," in *Industrial Electronics Society, 1998. IECON '98. Proceedings of the 24th Annual Conference of the IEEE*, vol. 1, Aug 1998, pp. 120–125 vol.1.

[50] R. Sreenivas, "Deterministic lambda;-free petri net languages and their application to the supervisory control of discrete event dynamic systems," in *Circuits and Systems, 1993., Proceedings of the 36th Midwest Symposium on*, aug 1993, pp. 340 –343 vol.1.

[51] ——, "On minimal representations of petri net languages," *Automatic Control, IEEE Transactions on*, vol. 51, no. 5, pp. 799–804, May 2006.

[52] J. Desel, "On cyclic behaviour of unbounded petri nets," in *Application of Concurrency to System Design (ACSD), 2013 13th International Conference on*, July 2013, pp. 110–119.

[53] M. Cabasino, A. Giua, M. Pocci, and C. Seatzu, "Discrete

event diagnosis using labeled petri nets. an application to manufacturing systems," *Control Engineering Practice*, vol. 19, no. 9, pp. 989 – 1001, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0967066111000025

[54] M. Dotoli, M. Fanti, A. Mangini, and W. Ukovich, "On-line identification of petri nets with unobservable transitions," in *Discrete Event Systems, 2008. WODES 2008. 9th International Workshop on*, may 2008, pp. 449 –454.

[55] A.-P. Estrada-Vargas, J.-J. Lesage, and E. López-Mellado, "Identification of industrial automation systems: Building compact and expressive petri net models from observable behavior," in *2012 American Control Conference (ACC'12)*, Canada, Jun. 2012, pp. 6095 – 6101. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00716235

[56] F. Basile, P. Chiacchio, J. Coppola, and G. De Tommasi, "Identification of petri nets using timing information," in *Dependable Control of Discrete Systems (DCDS), 2011 3rd International Workshop on*, june 2011, pp. 154 –161.

[57] A.-P. Estrada-Vargas, E. López-Mellado, and J.-J. Lesage, "A comparative analysis of recent identification approaches for discrete-event systems," *Mathematical Problems in Engineering*, vol. 2010, p. 10.1155/2010/453254, May 2010, 21 pages.

[58] H. Hu, M. Zhou, and Z. Li, "Supervisor optimization for deadlock resolution in automated manufacturing systems with petri nets," *Automation Science and Engineering, IEEE Transactions on*, vol. 8, no. 4, pp. 794 –804, oct. 2011.

[59] M. Roth, J.-J. Lesage, and L. Litz, "An fdi method for manufacturing systems based on an identified model," in *Information Control Problems in Manufacturing,(INCOM 2009)*, vol. 13, 15. Oct 2009.

[60] ——, "Black-box identification of discrete event systems with optimal partitioning of concurrent subsystems," in *American Control Conference (ACC), 2010*, 30 2010-july 2 2010, pp. 2601 –2606.

[61] D. E. Jarvis, "An identification technique for timed event systems," *10th International Workshop on Discrete Event Systems*, vol. 10, 2010.

[62] M. Roth, L. Litz, and J.-J. Lesage, "Identification of discrete event systems - implementation issues and model completeness." in *ICINCO (3)*, J. Filipe, J. Andrade-Cetto, and J.-L. Ferrier, Eds. INSTICC Press, 2010, pp. 73–80. [Online]. Available: http://dblp.uni-trier.de/db/conf/icinco/icinco2010-3.html

[63] M. Meda-Campana and E. Lopez-Mellado, "Required event sequences for identification of discrete event systems," in *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, vol. 4, dec. 2003, pp. 3778 – 3783.

[64] ——, "Identification of concurrent discrete event system using petri nets." in *Proceedings of the 17th IMACS World Congress on Computational and Applied Mathematics*, July 2005, pp. 11–15.

[65] L. Li and C. Hadjicostis, "Least-cost transition firing sequence estimation in labeled petri nets with unobservable transitions," *Automation Science and Engineering, IEEE Transactions on*, vol. 8, no. 2, pp. 394 –403, april 2011.

[66] A. Giua and C. Seatzu, "Identification of free-labeled petri nets via integer programming," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, Dec 2005, pp. 7639–7644.

[67] M. Cabasino, A. Giua, and C. Seatzu, "Identification of de-

terministic petri nets," in *Discrete Event Systems, 2006 8th International Workshop on*, july 2006, pp. 325 –331.

[68] ——, "Identification of petri nets from knowledge of their language," vol. 17, no. 4. Hingham, MA, USA: Kluwer Academic Publishers, Dec. 2007, pp. 447–474. [Online]. Available: http://dx.doi.org/10.1007/s10626-007-0025-0

[69] M. Dotoli, M. Fanti, and A. Mangini, "An optimization approach for identification of petri nets," in *Discrete Event Systems, 2006 8th International Workshop on*, july 2006, pp. 332 –337.

[70] ——, "On line identification of discrete event systems via petri nets: an application to monitor specification," in *Automation Science and Engineering, 2007. CASE 2007. IEEE International Conference on*, sept. 2007, pp. 893 –898.

[71] A. P. Estrada-Vargas, "Black-box identification of automated discrete event systems." Ph.D. dissertation, NORMALE SUPERIEURE DE CACHAN, 2013.

[72] M. Gomez-Lopez, R. Gasca, C. Valle, and S. Pozo, "Distributed model-based diagnosis using object-relational constraint databases," in *Advanced Information Networking and Applications, 2006. AINA 2006. 20th International Conference on*, vol. 2, April 2006, pp. 5 pp.–.

[73] S. Jiang and R. Kumar, "Failure diagnosis of discrete-event systems with linear-time temporal logic specifications," *Automatic Control, IEEE Transactions on*, vol. 49, no. 6, pp. 934–945, 2004.

[74] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, "Diagnosability of discrete-event systems," *Automatic Control, IEEE Transactions on*, vol. 40, no. 9, pp. 1555–1575, Sep 1995.

[75] P. Supavatanakul, J. Lunze, V. Puig, and J. Quevedo, "Diagnosis of timed automata: Theory and application to the damadics actuator benchmark problem," 2004.

[76] F. Basile, "Overview of fault diagnosis methods based on petri net models," in *Control Conference (ECC), 2014 European*, June 2014, pp. 2636–2642.

[77] J. Prock, "A new technique for fault detection using petri nets," *Automatica*, vol. 27, no. 2, pp. 239 – 245, 1991. [Online]. Available: http://www.sciencedirect.com/science/article/pii/000510989190074C

[78] A. Aghasaryan, E. Fabre, A. Benveniste, R. Boubour, and C. Jard, "Fault detection and diagnosis in distributed systems: An approach by partially stochastic petri nets," *Discrete Event Dynamic Systems*, vol. 8, no. 2, pp. 203–231, 1998. [Online]. Available: http://dx.doi.org/10.1023/A%3A1008241818642

[79] C. N. Hadjicostis and G. C. Verghese, "Monitoring discrete event systems using petri net embeddings," in *in Application and Theory of Petri Nets 1999, no. 1639 in Lecture Notes in Computer Science*. Springer-Verlag, 1999, pp. 188–208.

[80] A. Benveniste, E. Fabre, S. Haar, and C. Jard, "Diagnosis of asynchronous discrete-event systems: a net unfolding approach," *Automatic Control, IEEE Transactions on*, vol. 48, no. 5, pp. 714–727, May 2003.

[81] S. Genc and S. Lafortune, "Distributed diagnosis of discrete-event systems using petri nets," in *Applications and Theory of Petri Nets 2003*, ser. Lecture Notes in Computer Science, W. Aalst and E. Best, Eds. Springer Berlin Heidelberg, 2003, vol. 2679, pp. 316–336. [Online]. Available: http://dx.doi.org/10.1007/3-540-44919-1_21

[82] ——, "Distributed diagnosis of place-bordered petri nets," *Automation Science and Engineering, IEEE Transactions on*, vol. 4, no. 2, pp. 206–219, 2007.

[83] A. Giua and C. Seatzu, "Fault detection for discrete event systems using petri nets with unobservable transitions," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, Dec 2005, pp. 6323–6328.

[84] M. P. Cabasino, A. Giua, and C. Seatzu, "Fault detection for discrete event systems using petri nets with unobservable transitions," *Automatica*, vol. 46, no. 9, pp. 1531 – 1539, 2010.

[85] A. Ramirez-Trevino, E. Ruiz-Beltran, I. Rivera-Rangel, and E. López-Mellado, "Online fault diagnosis of discrete event systems. a petri net-based approach," *Automation Science and Engineering, IEEE Transactions on*, vol. 4, no. 1, pp. 31–39, Jan 2007.

[86] M. Dotoli, M. P. Fanti, A. M. Mangini, and W. Ukovich, "On-line fault detection in discrete event systems by petri nets and integer linear programming," *Automatica*, vol. 45, no. 11, pp. 2665 – 2672, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109809003720

[87] F. Basile, P. Chiacchio, and G. De Tommasi, "An efficient approach for online diagnosis of discrete event systems," *Automatic Control, IEEE Transactions on*, vol. 54, no. 4, pp. 748–759, 2009.

[88] C.-H. Kuo and H.-P. Huang, "Failure modeling and process monitoring for flexible manufacturing systems using colored timed petri nets," *Robotics and Automation, IEEE Transactions on*, vol. 16, no. 3, pp. 301–312, Jun 2000.

[89] Y. Li, T. Melliti, and P. Dague, "A colored petri nets model for the diagnosis of semantic faults of bpel services," in

Proc. of International Workshop on Petri Nets and Software Engineering (PNSE), 2009.

[90] A. S. Staines, "A compact cpn representation for embedded and control systems fault diagnosis and recovery," in *Proceedings of the 8th WSEAS International Conference on Software Engineering, Parallel and Distributed Systems*, ser. SEPADS'09. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2009, pp. 78–83. [Online]. Available: http://dl.acm.org/citation.cfm?id=1553890.1553905

[91] E. Garcia, L. Rodriguez, F. Morant, A. Correcher, and E. Quiles, "Latent nestling method: A new fault diagnosis methodology for complex systems," in *Industrial Electronics, 2008. IECON 2008. 34th Annual Conference of IEEE*, 2008, pp. 253–258.

[92] A. Ramirez-Trevino, E. Ruiz-Beltran, I. Rivera-Rangel, and E. Lopez-Mellado, "Diagnosability of discrete event systems: a petri net based approach," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 1, April 2004, pp. 541–546 Vol.1.

[93] M. Cabasino, A. Giua, and C. Seatzu, "Diagnosability of discrete-event systems using labeled petri nets," *Automation Science and Engineering, IEEE Transactions on*, vol. 11, no. 1, pp. 144–153, Jan 2014.

[94] M. Cabasino, A. Giua, L. Marcias, and C. Seatzu, "A comparison among tools for the diagnosability of discrete event systems," in *Automation Science and Engineering (CASE), 2012 IEEE International Conference on*, Aug 2012, pp. 218–223.

[95] J. Lunze, "Diagnosis of quantized systems based on a timed discrete-event model," *Systems, Man and Cybernetics, Part*

A: Systems and Humans, IEEE Transactions on, vol. 30, no. 3, pp. 322 –335, may 2000.

[96] M. Piera and G. Music, "Coloured petri net scheduling models: Timed state space exploration shortages," *Mathematics and Computers in Simulation*, vol. 82, no. 3, pp. 428 – 441, 2011, 6th Vienna International Conference on Mathematical Modelling. [Online]. Available: $http://www.sciencedirect.com/science/article/pii/S0378475410003216$

[97] K. Jensen, L. M. Kristensen, and L. Wells, "Coloured petri nets and cpn tools for modelling and validation of concurrent systems," in *INTERNATIONAL JOURNAL ON SOFTWARE TOOLS FOR TECHNOLOGY TRANSFER*, 2007, p. 2007.

[98] D. Ruppert, *Statistics and Data Analysis for Financial Engineering (Springer Texts in Statistics)*, 1st ed. Springer, Berlin, 2010. [Online]. Available: http://www.amazon.de/Statistics-Analysis-Financial-Engineering-Springer/dp/1441977864

[99] W. Conover, *Practical nonparametric statistics*, 3rd ed., ser. Wiley series in probability and statistics. New York, NY [u.a.]: Wiley, 1999. [Online]. Available: http://gso.gbv.de/DB=2.1/CMD?ACT=SRCHA&SRT=YOP&IKT=1016&TRM=ppn+24551600X&sourceid=fbw_bibsonomy

[100] E. Garcia, F. Morant, R. Blasco-Gimenez, A. Correcher, and E. Quiles, "Centralized modular diagnosis and the phenomenon of coupling," in *Discrete Event Systems, 2002. Proceedings. Sixth International Workshop on*, 2002, pp. 161–168.

[101] S. Abdelwahed, G. Karsai, N. Mahadevan, and S. Ofsthun, "Practical implementation of diagnosis systems using timed failure propagation graph models," *Instrumentation*

and Measurement, IEEE Transactions on, vol. 58, no. 2, pp. 240–247, Feb 2009.

[102] M. Cabasino, A. Giua, S. Lafortune, and C. Seatzu, "A new approach for diagnosability analysis of petri nets using verifier nets," *Automatic Control, IEEE Transactions on*, vol. 57, no. 12, pp. 3104–3117, Dec 2012.

[103] M. Danancher, M. Roth, J. Lesage, and L. Litz, "A comparative study of three model-based fdi approaches for discrete event systems," in *Dependable Control of Discrete Systems (DCDS), 2011 3rd International Workshop on*, June 2011, pp. 29–34.

[104] M. A. Eltawil and Z. Zhao, "Grid-connected photovoltaic power systems: Technical and potential problems - a review," *Renewable and Sustainable Energy Reviews*, vol. 14, no. 1, pp. 112 – 129, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1364032109001749

[105] A. Houssein, N. Heraud, I. Souleiman, and G. Pellet, "Monitoring and fault diagnosis of photovoltaic panels," in *Energy Conference and Exhibition (EnergyCon), 2010 IEEE International*, Dec 2010, pp. 389–394.

[106] K.-H. Chao, S.-H. Ho, and M.-H. Wang, "Modeling and fault diagnosis of a photovoltaic system," *Electric Power Systems Research*, vol. 78, no. 1, pp. 97 – 105, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378779607000041

Nomenclatura

SED	Sistema de eventos discretos.
E/S	Señales de entrada / salida del sistema.
o_{om}	Modo de operación.
c	Número de subsistemas.
Cc	Órdenes de control. $Cc = GCc \cup LCc$
GCc	Órdenes de control globales $GCc_g = \left\{gcc_1, \cdots, gcc_{n_{gc}}\right\}$, n_{gc} es el número de órdenes de control globales.
LCc	Órdenes de control locales $LCc = \cup LCc_l$, $l = 1 \cdots c$, c es el número de subsistemas $LCc_l = \left\{cc_{l,1}, \cdots cc_{l,n_{l,cc}}\right\}$ y $n_{l,cc}$ es el número de órdenes de control en el subsistema l.
m_t	Número total de entradas. $m_t = m_g + m_L$; $m_L = \sum_{l=1}^{c} m_l$; $m_l = m_{l_{uo}} + m_{l_o}$.
Sr	Lecturas sensoriales. $Salidas = \cup Sr_l$.
Sr_l	Lecturas sensoriales locales $Sr_l = \cup sr_{l,n_{l,sr}}$; donde $n_{l,sr}$ es el número de sensores en el subsistema l y $l = 1 \cdots c$.
n_t	Número total de salidas. $n_t = \sum_{l=1}^{c} n_l$.
u_s	Símbolo de entrada. $u_s = \left[euop_1 \cdots eup_{n_{euop}} eop_1 \cdots ep_{n_{eop}} cc_1 \cdots cc_{n_{cc}}\right]$, s representación decimal del vector binario de entrada.
U	Alfabeto de entrada. $U = \left\{u_0, u_1, \cdots, u_{\left\|2^{(m_t)}\right\|-1}\right\}$

y_j	Símbolo de salida $y_j = [sr_1...sr_n]$, j representación decimal del vector binario de salida.
Y	Alfabeto de salida. $Y = \left\{y_0, y_1, \cdots, y_{\lvert 2^{n_t}\rvert -1}\right\}$
dy_j	Diferencial de símbolo de salida. $dy_j(y_j \times y_{j-1}) \rightarrow \{-1, 0, 1\}$
ω^i	Símbolo de E/S, al instante τ_i Hay k instantes de tiempo $\tau_1 \leq \tau_2 \leq \cdots \leq \tau_k$
ω_l^i	Símbolo de E/S en el subsistema l al instante τ_i. $\omega_l^i = (u_{ls} y_{lj})\, \tau_i$, $u_{ls} \in U_l$; $U_l = \left\{u_0, u_1, \cdots, u_{\left\lvert 2^{(m_{gc}+m_l)}\right\rvert -1}\right\}$ y $y_{lj} \in Y_l$; $Y_l = \left\{y_0, y_1, \cdots, y_{\lvert 2^{n_l}\rvert -1}\right\}$.
PN	Red de Petri. $N = (P, TR, Pre, Post, M_0)$
IPN	Red de Petri Interpretada. $Q = (N, U, Y, \lambda, \varphi)$
st-IPN	Red de Petri Interpretada, temporizada, estocástica. $stQ = (Q, \Omega, \delta)$.
P	Conjunto de lugares. $\lvert P\rvert = np$. (p_q lugar q)
P_l	Conjunto de lugares en el subsistema l. (p_{l,q_l} lugar q_l en el subsistema l)
TR	Conjunto de transiciones. $\lvert TR\rvert = ntr$. (tr_r transición r)
TR_l	Conjunto de transiciones en el subsistema l. (tr_{l,r_l} transición r_l en el subsistema l)
σ	Secuencia de disparo. $\sigma = tr_1 \cdots tr_k$.
$R(N, M_0)$	Conjunto de alcanzabilidad de una PN
$\mathcal{L}(F)$	Lenguaje de Disparo. $\mathcal{L}(F) = \left\{\sigma \lvert \sigma = tr_1 tr_2 \cdots tr_k \cdots \wedge M_0 \stackrel{tr_1}{\rightarrow} M_1 \stackrel{tr_2}{\rightarrow} \cdots M_w \stackrel{tr_k}{\rightarrow} \cdots \mid M_0, M_1, M_w \in R(N)\right\}$

$\mathcal{L}(Q)$	Lenguaje global del sistema. $\mathcal{L}(Q) = \omega^1\omega^2 \cdots \omega^k$ a instantes, $\tau_1 \leq \tau_2 \leq \cdots \leq \tau_k$
$\mathcal{L}(Q_l)$	Lenguaje del subsistema l. $\mathcal{L}(Q_l) = \omega_l^1\omega_l^2 \cdots \omega_l^k$ a instantes, $\tau_1 \leq \tau_2 \leq \cdots \leq \tau_k$
$F_n(x)$	Función escalonada y no decreciente.
$F_0(x)$	Función de distribución continua.
D	Estadístico que mide la discrepancia máxima
$(p - valor)$	entre $F_n(x)$ y $F_0(x)$.
$p - valor_\alpha$	Estadístico para test de bondad de ajuste con nivel de significancia α.
n	Tamaño de muestra.
$st_{\alpha/2}$	Estadístico para test de contraste con un nivel de significancia $\alpha/2$
S^2	Cuasi-varianza muestral.
d	Nivel de precisión del estudio $d = \left(st_{\alpha/2}\right) \times S/\sqrt{n}$.
RMSE	raíz cuadrada del error medio cuadrado, $RMSE = \frac{S}{\sqrt{n}}$.
CPN	Red de Petri Coloreada. $CN = (P, TR, Pre, Post, C, cd, M_0)$.
C	Conjunto de clases de color.
$Bag(A)$	Conjunto de todos los *bgs* (multiconjuntos) sobre A. $(bg : A \to \mathbb{N})$.
β	Conjunto de asignaciones de color de la forma $f : cd(tr) \to Bag(cd(p))$.
$ICPN$	Red de Petri Interpretada Coloreada. $CQ = (CN, U, Y, \lambda, \varphi)$.
nc	Número de clases de color.
U_h	Alfabeto de entrada para la clase de color h. $U_h = \left\{u_{0,h}, u_{1,h}, \cdots, u_{\lvert 2^{m_h}\rvert -1,h}\right\}$
u_s^h	Símbolo de entrada para la clase de color h.

Y_h	Alfabeto de salida para la clase de color h. $Y_h = \left\{y_{0,h}, y_{1,h}, \cdots, y_{\lvert 2^{n_h}\rvert -1,h}\right\}$.
y_j^h	Símbolo de salida para la clase de color h.
st-ICPN	Red de Petri Coloreada Interpretada Temporizada Estocástica. $stCQ = (CQ, \Omega, \delta)$.