ANEXO 1

CONTROL DE SISTEMAS A EVENTOS DISCRETOS

1.1 Modelado De Sistemas A Eventos Discretos

Es clásico mencionar que un modelo representa un objeto artificial construido para representar de forma simplificada un aspecto problemático de la realidad dentro de un fenómeno o sistema real. Son muchos los sistemas o campos de aplicación en los que se desconocen las consecuencias de la ocurrencia de un cierto evento (como podría ser una perturbación), así como la influencia sobre el rendimiento global del proceso de una variación en la secuencia de eventos que se producirían como consecuencia de la aparición de un cierto evento situado en algún instante anterior en el tiempo. Esta situación en el modelamiento de Sistemas de Producción se torna compleja por la dificultad de formalizar las secuencias de actividades que puede desencadenar un determinado evento.

En un sistema de manufactura es habitual que existan fuertes y complejas interacciones entre sus variables y que éstas respondan a eventos discretos, algunos de ellos como la existencia de una falla o la llegada imprevista de una orden sin una causa o una temporalidad conocida, lo que dificulta el conocer cómo evolucionará el sistema en su conjunto. (Adaptación de ())

El modelamiento de sistemas a eventos discretos está generalmente asociado con algunos formalismos populares como Autómatas de Estado Finito, Redes de Petri, Grafos de Eventos (Event Graphs), Statecharts, entre otros. De acuerdo a (Chen and Lin, 2000) y (Huang et al, 2004) la teoría de control para sistemas a eventos discretos modelada como máquinas de estado finito, se ha desarrollado en relación de diversas situaciones fundamentales de control. Sin embargo, el modelado de máquinas de estados finito tiene la debilidad de la explosión de estados que hace inadecuado para muchas aplicaciones prácticas. En esta línea y según lo presentado en (Zapata, 2008) otro de los métodos ampliamente difundidos para modelar y controlar DES, son los basados en las denominadas Redes de Petri (PN) son una generalización de la teoría de autómatas. Las PN tienen una representación matemática sencilla empleando álgebra lineal y teoría de conjuntos, lo que las hace particularmente útiles para modelar y analizar sistemas a eventos discretos.

Las principales características de los sistemas a eventos discretos que deben ser consideradas por un formalismo que represente el control de un sistema a eventos discretos son:

- **Son asíncronos.** Porque algunos eventos pueden ocurrir en cualquier momento, sin ningún tipo de periodicidad ni de continuidad.
- **Están dirigidos por eventos.** Cuando ocurre un suceso cambia el estado del sistema.
- **Son secuenciales.** Porque puede haber eventos que guarden una cierta secuencia, tal que para que ocurra uno, antes debe de haber ocurrido el anterior.

- ➲ **Presentan sincronización**. Es el problema de retrasar la ejecución de un proceso hasta que se cumpla una determinada condición.

- ➲ **Presentan concurrencia.** Porque varios eventos pueden ocurrir al mismo tiempo.
  Aquél DES que contiene uno o más procesos que trabajan de forma conjunta para realizar una determinada tarea.

- ➲ **Pueden representar conflictos o exclusión mutua.** El conflicto se presenta cuando un recurso es compartido por varias entidades y se resuelve haciendo que no se puedan presentar al mismo tiempo dos solicitudes del recurso.

- ➲ **Pueden presentar parada por interbloqueo o deadlock.** Por ejemplo, el robot ha cogido una pieza de la máquina 1 y la máquina 2 requiere la pieza y no la recibe.

Antes de seleccionar un formalismo para la representación de control de DES, es necesario presentar los requerimientos impuestos desde la teoría de control para el modelamiento de DES. Conjuntamente se presentan las exigencias definidas hacia el modelamiento de control desde la complejidad de los sistemas representados. Por tanto, es necesario cumplir con los siguientes requerimientos para lograr una representación de un sistema dinámico:

- ➲ ***Reactividad:***

  La reactividad consiste en mantener la operación productiva y/o en contribuir a reducir el tiempo de inactividad cuando se detecta un problema.

- ➲ ***Tratamiento de bloqueos.***

  Cuando dos procesos y quieren tener acceso simultáneamente a dos recursos r0 y r1, es posible que se produzca un bloqueo de ambos procesos (Formella, 2006).

  1. los procesos tienen que compartir recursos con exclusión mutua

  2. los recursos no permiten ser usados por más de un proceso al mismo tiempo

     Ante las situaciones de bloqueo es posible uno de los siguientes comportamientos:

     - ⇨ Detectar y actuar
     - ⇨ Evitar
     - ⇨ Prevenir

- ➲ ***Resolución de conflictos***

  Los sistemas de control inteligentes deben combinar mecanismos de resolución de conflictos y estrategias de control para detectar, identificar y resolver conflictos lógicos causados por la posibilidad de eventos simultáneos, que es propio de sistemas no determinísticos como los sistemas de manufactura. En (Wong et al, 1998) se definen prioridades independientes de la dinámica del sistema como esquema de resolución de conflictos entre controladores concurrentes.

➲ ***Vivacidad del sistema de control (Sistema libre de bloqueos)***

Una propiedad de viveza (liveness property) es aquella que garantiza que el programa entrará eventualmente en un estado valido (Rus Mansilla, 2005), (Formella, 2006).

Ejemplos de propiedades de vivacidad:

- ⇨ Ningún proceso se muere por inanición, es decir, el sistema resuelve justamente los conflictos. Estos es, si un proceso pide un recurso, lo consigue en algún momento.
- ⇨ Los procesos no se bloquean mutuamente
- ⇨ No se termina un proceso desde fuera sin razón
- ⇨ Un proceso no queda dormido o suspendido
- ⇨ la conexión entre procesos es fiable

➲ ***El control es acotado y estable (Seguro):***

- ⇨ Una propiedad de seguridad (safety property) es aquella que garantiza que el programa nunca entra en un estado no válido. Indica que no está pasando nada malo en el programa, es decir, el programa no ejecuta instrucciones que no debe hacer.
- ⇨ Un estado no es válido cuando algunas variables de estado tienen un valor incorrecto

➲ ***Alcanzabilidad***

Cuando un sistema cuenta con esta característica, entonces mediante un controlador se puede llevar este sistema desde un estado inicial hasta otro estado cualquiera, en un tiempo finito

➲ ***Controlabilidad***

Un sistema es controlable si se puede llevar desde cualquier punto al origen en tiempo finito. De acuerdo a (Formella, 2006) la controlabilidad no implica alcanzabilidad

Estas propiedades permiten validar el comportamiento del procedimiento de control de un sistema.

1.2 Formalismos de Modelamiento de Sistemas a Eventos Discretos

Mediante paradigma o formalismo se referencia a un conjunto de conceptos, leyes y medios que sirven para definir un conjunto de modelos. Hay varios paradigmas para especificar formalmente sistemas a eventos discretos y la mayoría de ellos tienen existencia conceptual independiente de los lenguajes de simulación que pueden usarse para llevar a cabo las simulaciones.

En el estudio de los sistemas a eventos discretos (DES) se han desarrollado soluciones teóricas acompañadas de métodos y algoritmos que implementadas a través de soluciones informáticas, han permitido el control, automatización e integración de estos sistemas.

Los formalismos mejoran la comprensión de los sistemas, permiten identificar parámetros clave e influencias, conducen a razonamiento más eficiente, ayudan en la aplicación, entre otros. Por otra parte, algunos métodos formales facilitan el diálogo entre las distintas personas implicadas en el diseño y operación, especialmente cuando se proporcionan representaciones gráficas e intuitivas. Adicionalmente, el uso de herramientas formales de modelamiento es una necesidad del medio industrial, debido a las propiedades que ofrecen, tales como el uso del mismo modelo para el análisis de las propiedades de comportamiento (propiedades de alcanzabilidad, seguridad y vivacidad) y la evaluación de desempeño mediante herramientas matemáticas (Zapata et al, 2008).

Algunos de los formalismos que han sido utilizados para representar el control de un sistema mediante sistemas a eventos discretos son: Las Redes de Petri, El GRAFCET, El formalismo DEVS y las máquinas de estado finito. De acuerdo a (Guasch et al, 2002) deben respetar las siguientes propiedades:

- El formalismo debe ser independiente de los constructores y herramientas que ofrecen los entornos de simulación.
- El modelo formalizado debe poder ser analizado para determinar relaciones entre componentes y evaluar alternativas que permitan la simplificación del modelo.
- El formalismo debe permitir una fácil transformación a las representaciones soportadas por los entornos de simulación.
- Adecuación para tratar con sistemas reales.
- Facilidad de uso e implementación.

Por tanto, los sistemas a eventos discretos definen requerimientos hacia los formalismos, estos provienen desde la necesidad definida por la dinámica natural y los requerimientos de representación de control de los DES:

- Concurrencia de operaciones.

- Sincronización y operaciones
- Secuencilidad
- Recursos limitados
- Representación de conflictos
- Seguridad
- Vivacidad
- Fairness (Imparcialidad)

  Cuando los procesos compiten por acceso a recursos compartidos, este se puede definir por los conceptos de Imparcialidad. Normalmente se quiere que todos los procesos manifiesten algún progreso en su trabajo. Sin embargo, eso no es necesario en programas concurrentes; se puede vivir bien con algunos procesos "muertos", mientras no involucre otros problemas para el controlador. Siempre existe la posibilidad que el trabajo asignado a un proceso está hecho por otro proceso dejando el primero en espera infinita.

  La Imparcialidad ("fairness") es la garantía de que todo proceso tiene la posibilidad de evolucionar, independientemente de lo que hagan los restantes procesos.

  En programas concurrentes es posible que un proceso nunca llegue a hacer nada si el planificador o el control de los recursos compartidos respectivamente no permiten que el proceso pueda cumplir con sus pedidos. Es decir, el proceso está sometido a una espera infinita, o en otras palabras, sufre una inanición. Para superar esto en el sistema debe definirse una regla de prioridad que le permita a los procesos acceder con Imparcialidad al recurso compartido.

  "el comportamiento es "equitativo".

- ***Verificación***

  La finalidad de la verificación es comprobar que no se han cometido errores al traducir el modelo, bien usando un entorno de modelado o mediante un lenguaje de simulación o de programación.

  Entre otros, pueden usarse los siguientes procedimientos para verificar el modelo:

  - Verificación manual de la lógica. Consiste en ejecutar la simulación durante un periodo de tiempo corto y comprobar manualmente los resultados obtenidos.
  - Árbol de alcanzabilidad
  -

- ***Validación***

  La validación consiste en comprobar que el modelo supone una aproximación adecuada de la realidad para los objetivos particulares del estudio de simulación.

  Puede considerarse que la validación del modelo tiene tres vertientes diferentes.

  Consiste en determinar:

  - Si el modelo representa adecuadamente al sistema real (comprobación de la estructura del modelo).
  - Si los datos generados de la simulación del modelo reproducen de forma adecuada el comportamiento del sistema real (comprobación del comportamiento del modelo).

1.3 Redes de Petri

Las Redes de Petri (PN) son una herramienta matemática y gráfica que permite modelar, simular y controlar DES (Silva, 1985), (Guasch et al., 2002). Las PN representan una herramienta de modelado independiente de cualquier tecnología, clara, fácil de utilizar y no ambigua, que comprende los conceptos básicos de receptividad y sensibilidad, por los cuales es posible obtener descripciones de los sistemas a modelar con un mínimo de información conocida y suficiente para sintetizarlos. Además las PN son ideales como metodología de modelado para capturar las relaciones causales y de precedencia entre eventos y situaciones, es decir, facilitan la representación de evoluciones simultáneas, las cuales son algo clave a la hora de modelar y simular el comportamiento de un sistema de manufactura. Como se muestra en (Zapata, 2008), La Figura 78 muestra un DES que evoluciona entre los estados x_1, x_2, x_3 y x_4 por la ocurrencia de los eventos α, β, λ y σ. Esta figura representa la relación de la trayectoria definida por un sistema a eventos discreto y el modelo PN que constituye esta trayectoria.

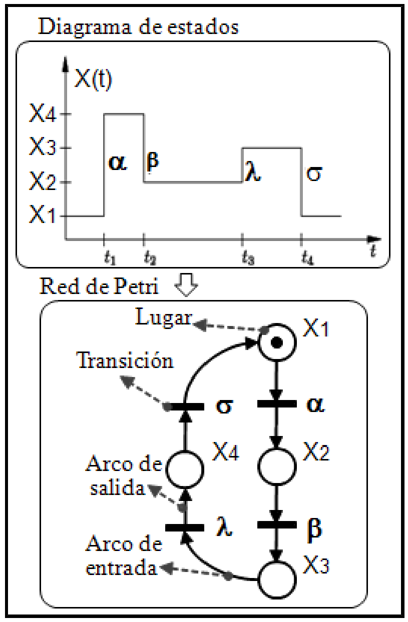

Figura 78. Representación de un DES mediante PN

La especificación de un modelo orientado a eventos discretos usando el formalismo de las PN permite obtener información del sistema, tanto si se analiza el comportamiento de la red como si se estudia su estructura.

Específicamente en sistemas de manufactura, la formalización con PN permite representar de forma muy natural clientes o peticiones, recursos y procesos como marcas situadas en los distintos nodos lugar. Adicionalmente (Silva et al, 1998) expresan que las PN son un formalismo que provee un marco o paradigma de trabajo apropiado para el diseño y la representación de la operación de sistema de producción.

En resumen, las PN obtienen su utilidad fundamentalmente porque permiten:

- Representan de forma explícita los estados y eventos del modelo.
- Los fenómenos de concurrencia, sincronismo y dependencia causal se representan de forma natural.
- El conjunto de recursos restringidos se representa de forma explícita en el modelo.
- Hay muy pocas reglas, lo que facilita su aprendizaje.
- Su representación gráfica es muy intuitiva.
- Su semántica es precisa y sin ambigüedades.
- Es independiente de la herramienta de simulación que se emplee.

- Capacidad para representar de forma natural concurrencia, la causalidad, la sincronización, recursos compartidos, los conflictos, comportamientos no deseados del sistema (Situaciones de bloqueo), tal como se ilustra en la Figura 79.

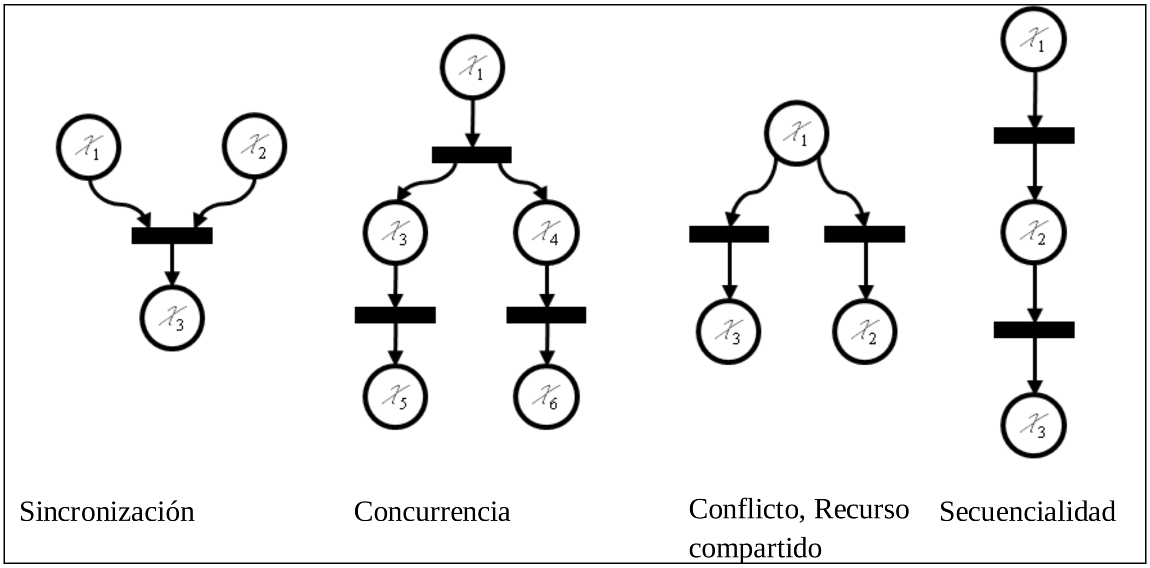

Figura 79. Capacidad de representación de las PN

- Permiten modularidad y reusabilidad. Esto permite una representación compacta debido a la representación de estados de forma distribuida, ventaja en comparación a una representación secuencial.

- La representación gráfica facilita la documentación y control del sistema.

Para modelar los sistemas a eventos discretos las PN se basa en el concepto de que las relaciones entre los componentes de un sistema pueden ser representadas por una red que permite el análisis de manera formal, obtener información del comportamiento dinámico del sistema modelado, capturar las relaciones de precedencia e interacción de eventos concurrentes y asíncronos. Son modelos lógicos con un fundamento matemático que permite el análisis cuantitativo y cualitativo del sistema (Vásquez, 2007). En esta línea, la robustez matemática del método de modelamiento brinda técnicas formales para analizar las propiedades y garantizar el desempeño del sistema. Típicamente se consideran las propiedades: vivacidad, seguridad y reversibilidad.

Para modelar la ejecución de una secuencia de eventos sobre el sistema modelado, es necesario asociar a las transiciones de una PN una etiqueta que relacione un operador a una transición. Además, las PN utilizadas para este trabajo tienen la capacidad de manejar variables con valores enteros, caracteres y booleanos en los estados o lugares definidos en el modelo. Bajo esta consideración los arcos deben tener la capacidad de reconocer inscripciones con expresiones que contengan constantes, variables y funciones *(f())*. Conjuntamente se considera la restricción de la dinámica del modelo mediante la asignación a una transición de una función de guarda.

En consecuencia, para representar estos conceptos de control y flujo de información, se realiza una extensión de las PN, definida como Redes de Petri Etiquetadas de Alto Nivel (High-Level labeled Petri Net-(HLLPN)).

Formalmente una HLLPN es una 9-tupla N=$\langle P,T,F,M_I,l,\vartheta,C,G,E\rangle$, donde P y T son un conjunto finito de lugares y transiciones respectivamente, A$\subseteq$(PxT)U(TxP) es un conjunto finito de arcos representando el flujo de relaciones (flow relation), l es una etiqueta que asigna a cada transición un evento $l:T\rightarrow 2^{\Sigma}\cup\{\varepsilon\}$ donde $\Sigma=\Sigma_{nc}\cup\Sigma_c$ y ε es un evento silencioso (Zapata, 2008), ϑ es un conjunto finito no vacío de conjuntos numéricos-NC (R, Z, R^+, B), C es una función que mapea a cada lugar P un conjunto numérico, tal que C:P$\rightarrow\vartheta$. G relaciona una función de guarda que mapea a cada transición un predicado que representa un valor booleano de la forma:$\forall$t$\in$T: [NC(G(t))=B $\wedge$ NC(Var(G(t)))$\subseteq\vartheta$], E es una función que mapea a cada arco una expresión de arco del tipo C((p(a)) tal que: $\forall$a$\in$A:[NC(E(a))=C(p)$\wedge$NC (Var(E(a)))$\subseteq\vartheta$], M_I es la función de inicialización, que es definida desde P mediante una expresión cerrada tal que: $\forall$p$\in$P:[NC(I(p))=C(p)], (Jensen,1997).

La red R(N, M_i) es una red marcada con marcaje inicial M_I. La notación $M_I[\sigma\rangle M$ se utiliza para expresar que el disparo de σ en el marcaje M_i lleva a M (Zapata, 2008), donde σ representa una secuencia de disparos $\sigma = t_1\ t_2\ t_3 \ldots t_n$

Los tipos de dato asignados a las marcas pueden ser inspeccionados mediante las expresiones de arco de llegada a las transiciones, lo que permite activarlas no sólo en función del número de marcas en los nodos lugar de entrada a las transiciones sino en función del valor del tipo de dato de las marcas disponibles en dichos nodos lugar. Al mismo tiempo estas expresiones modelan también los efectos de salida de la transición modificando los valores de los tipos de dato de las marcas de los nodos lugar de salida.

Las PN Etiquetadas de Alto Nivel permitan especificar el flujo de información que suele reflejarse mediante datos asignados a entidades cuyos valores cambian en función de los eventos que aparecen. Estas entidades en las PN Etiquetadas de Alto Nivel se representan como marcas de distintos tipos de dato en los correspondientes nodos lugar de la red. Con estos atributos, las PN Etiquetadas de Alto Nivel permiten construir modelos más compactos y paramétricos, lo que facilita considerablemente su mantenimiento e implementación. Estos modelos requerirían de estructuras con un número elevado de componentes si fueran desarrollados con el formalismo de las PN (Guasch et al., 2002).

Concretamente, al formalismo que represente el control de los DES se le exige del cumplimiento de los siguientes requerimientos:

- **Entidades**

  Las entidades son objetos dinámicos en la simulación, que son creados y se mueven por el sistema, cambiando el valor de sus atributos, afectados por otras entidades y por el estado del sistema. Concretamente, las entidades comprenden los objetos, las personas, los conceptos, entre otros, cuya existencia es reconocida por algún sistema de ontología, que define las relaciones de un conjunto de entidades dentro un dominio definido como sistema de estudio (Honderich, 2005).

➲ Atributos

Los atributos son características o propiedades de las entidades, permiten individualizar cada instanciación de una determinada clase de entidad sin más que asignar valores a sus atributos. Por ejemplo, algunos atributos que podrían definirse para el tipo de entidad "cliente" son: la prioridad con que debe ser atendido o determinados datos personales, como son el nombre y los apellidos, la edad, la nacionalidad, etc. En general, el valor de los atributos diferirá de un cliente a otro y es lo que permite diferenciarlos.

➲ Variables

Las variables representan características del sistema que son independientes de los tipos de entidades o del número de realizaciones existentes en determinado instante. Por tanto, las variables no están asociadas a entidades en concreto, sino que pertenecen al conjunto del sistema.

➲ Recursos

Los recursos pueden ser el personal ("operario"), las máquinas (por ejemplo, si las entidades son piezas que deben ser procesadas), el espacio (por ejemplo, en un almacén), etc. Una entidad captura un recurso cuando éste está disponible, a fin de obtener un servicio de él, y lo libera una vez ha terminado.

El recurso puede ser individual o estar compuesto por un grupo de elementos individuales, cada uno de los cuales se llama una unidad del recurso.

➲ Actividades

Las actividades son las tareas o acciones que tienen lugar en el sistema. Toda actividad está siempre delimitada por dos eventos, el de comienzo y el de fin de la actividad, por tanto tiene una duración temporal y, normalmente, precisa del uso de recursos.

Ejemplos de actividades son: la reparación de una máquina, el procesado de una pieza, el transporte de un cliente, etc.

➲ Flujo de Información

La necesidad de representación de flujos de conocimiento se deriva de la realidad, las especificaciones de las tareas de control (como criterios, restricciones, diferentes circunstancias concernientes con las influencias del entorno, entre otras) estas se expresan como reglas, estilo sistemas expertos. Entonces, un dominio orientado a una base de conocimiento (KB) expresando el conocimiento sobre las especificaciones de las tareas de control es requerido sea definido.

1.3.1 Propiedades de las Redes de Petri

La fortaleza del modelado de las PN radica en sus propiedades, que se dividen en dos grandes áreas, las dependientes del marcado inicial llamadas propiedades *dinámicas* o *del comportamiento* y las propiedades independientes del marcado, llamadas *estructurales* o *estáticas* (Soto, 2008).

Propiedades de Comportamiento

Las principales propiedades de comportamiento son la alcanzabilidad, la acotabilidad, la vivacidad, la reversibilidad, la cobertura y la persistencia.

- Marcado Alcanzable: Un marcado M_n se dice alcanzable desde un marcado si existe una secuencia de disparos que transforma M_i en M_n .
- Alcanzabilidad: El problema de la alcanzabilidad para las redes de Petri será el problema de encontrar si M pertenece a R(M_i) en una red dada R(N, M_i).
- Red de Petri Acotada: Una red de Petri R(N, M_i) se dice k-acotada o acotada si el número de tokens en cada lugar no es superior a un número finito k para cualquier marcado alcanzable desde M_i, es decir, M(p)≤ k para todo lugar p y todo marcado M que pertenece a R(M_i).
- Red de Petri Segura: Una red de Petri R(N, M_i) se dice segura si está 1-acotada.
- Interbloqueo: Se dice que en una red de Petri ocurre un interbloqueo cuando se alcanza un marcado desde el que no se puede disparar ninguna transición.
- Red de Petri Viva: Una red de Petri R(N, M_i) se dice que está viva (o equivalentemente se dice que es un marcado vivo para N) si, sea cual sea el marcado que se alcance desde , existe una secuencia disparable que permite disparar cualquier transición de la red. La vivacidad garantiza, por tanto, la ausencia de interbloqueos. Esto implica que cualquier transición es eventualmente disparable en alguna secuencia de disparo.

Propiedades Estructurales

Las propiedades estructurales son aquellas que dependen de la estructura topológica de las redes de Petri. Son independientes del marcado inicial en el sentido de que dichas propiedades se cumplen para cualquier marcado inicial. Son propiedades estructurales la acotabilidad estructural, la vivacidad estructural, la controlabilidad, la conservatividad, la repetitividad y la consistencia (Soto, 2008).

Red de Petri Acotada Estructuralmente: Una red de Petri está acotada estructuralmente si está acotada para cualquier marcado inicial finito.

Lugar No Acotado Estructuralmente: Un lugar p en una red de Petri se dice no acotado estructuralmente si existe un marcado M y una secuencia de disparo σ desde M tal que p no esté acotado.

Red de Petri Estructuralmente Viva: Una red de Petri está estructuralmente viva si existe algún marcado inicial para el que está viva.

Red de Petri Controlable: Una red de Petri se dice completamente controlable si cualquier marcado es alcanzable desde cualquier otro marcado.

1.3.2 Métodos de análisis de propiedades

A partir de una Red de Petri R(N, M_i), se puede obtener tantos nuevos marcados como transiciones habilitadas disparadas. Este proceso resulta en un árbol de marcados infinito para una PN no acotada. Para redes acotadas, el árbol de cobertura es llamado árbol de alcanzabilidad.

Formalmente un árbol de alcanzabilidad es una enumeración del conjunto de marcajes alcanzables con un marcaje M_i para una red de Petri R(N, M_i) (Qinghua, 2008).

El conjunto de marcajes alcanzables desde el marcaje inicial M_i con diferentes secuencias de disparo de transiciones se representa como un grafo dirigido, donde los nodos corresponden a los marcajes generados y los arcos corresponden a las transiciones disparadas entre marcajes de la red. Un árbol de alcanzabilidad completo representa todas las posibles trayectorias en el espacio de estados discretos que describe el comportamiento de una red Petri. A través del árbol de alcanzabilidad es posible realizar la búsqueda de una ruta óptima entre dos marcajes en el grafo de estados resultante a partir de la evolución de una red de Petri. Esta capacidad es útil para programación de actividades en un sistema a eventos discretos, dado que es posible a analizar todas las posibles secuencias de disparo, que en el grafo de estado corresponden a programas factibles como solución al problema de asignación de actividades en un DES.

Cada nodo en el árbol de alcanzabilidad que representa un estado de la red de Petri, está definido en su interior por un número de nodo y en la parte inferior se referencia la cantidad de nodos predecesores y la cantidad de nodos sucesores, tal como se ilustra en la Figura 80.

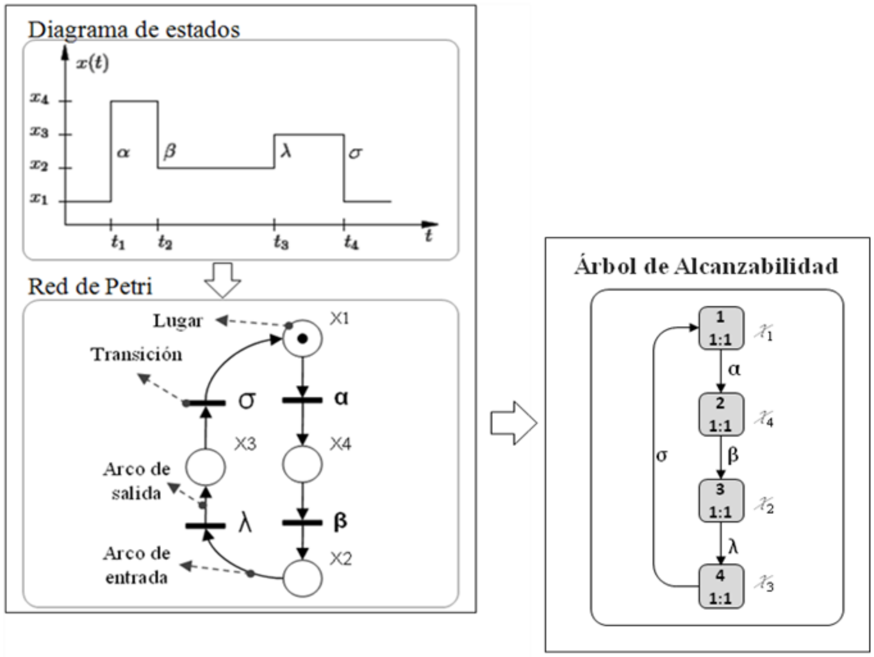

Figura 80. Árbol de alcanzabilidad

ANEXO 2

PROPIEDADES DINÁMICAS DE LAS REDES DE PETRI

Con el fin de evitar que en el proceso de elaboración de un sistema de control se pase a la fase de implementación con un modelo erróneo, se estudiarán algunas técnicas para llevar a cabo la verificación y validación funcional de los modelos hechos.

Para ello se introducirán las propiedades dinámicas definidas para las redes de Petri, las cuales están íntimamente ligadas a las propiedades de los grafos, estas propiedades son: Alcanzabilidad, Acotación, Reversibilidad, Vivacidad e Imparcialidad.

2.1 Definiciones Formales de Propiedades

En el presente trabajo los análisis serán hechos con base en simulaciones y Grafo de Alcanzabilidad, por lo tanto, las propiedades dinámicas serán dadas en términos de teoría de grafos simplificada y extendida al caso particular de las redes de Petri Binarias autónomas no jerárquicas, en el Capítulo 3 se extenderán las definiciones a las redes Jerárquicas.

2.1.1. Grafo de Alcanzabilidad

El Grafo de Alcanzabilidad de una PN [4] es un grafo dirigido: $\mathbf{OG} = (\mathbf{V}, \mathbf{A}, \mathbf{N})$ que satisface las siguientes condiciones:

(i) **V** es un conjunto de **Nodos** tal que**:** $\mathbf{V} = [\mathbf{M_0}\rangle$

(ii) **A** es un conjunto de **Arcos** tales que: $\mathbf{V} \cap \mathbf{A} = \phi$

(iii) **N** es una **Función de Nodo** la cual está definida como:

$$\forall a \in A : N(a) \in [A \to V \times V]$$
$$N(a) = (M_1, M_2)$$

El Grafo de Alcanzabilidad tiene un nodo para cada marcaje alcanzable y un arco para cada paso que ocurre, el nodo fuente del arco es el marcaje inicial para el paso mientras que el nodo de destino es el marcaje final.

Proposición:

El Grafo de Alcanzabilidad satisface las siguientes propiedades:

(i) Cada secuencia de ocurrencia finita:

$$\mathbf{M_1 \xrightarrow{t_1} M_2 \xrightarrow{t_2} M_3 \cdots M_n \xrightarrow{t_n} M_{n+1}}$$

Donde $\mathbf{M_1 \in [M_0\rangle}$ y $\mathbf{t_i \in T}$ para todo $\mathbf{i = 1,2,\ldots,n}$

Con $\mathbf{[M_0\rangle}$ el conjunto de los marcajes alcanzables a partir de $\mathbf{M_0}$ y **T** el conjunto de todas las transiciones.

Tiene asignado un camino dirigido entre cada par de nodos:

$$\mathbf{M_1(M_1,t_1,M_2)M_2(M_2,t_2,M_3)M_3 \cdots M_n(M_n,t_n,M_{n+1})M_{n+1}}$$

Y cada camino dirigido finito tiene asignada una secuencia de ocurrencia.

2.1.2. Componentes Fuertemente Conectados

Un conjunto de nodos $\mathbf{V^* \subseteq V}$ está fuertemente conectado [16] si y solo si:

(i) $\forall \mathbf{V_1, V_2 \in V^* \times V^* : DPF(V_1, V_2) \neq \phi}$

DPF: es el conjunto de todos los caminos dirigidos finitos $\mathbf{DPF(V_1, V_2)}$ que comienzan en $\mathbf{V_1}$ y terminan en $\mathbf{V_2}$. En un Grafo de Alcanzabilidad, el **DPF** es un conjunto de arcos.

Un componente fuertemente conectado es el Subgrafo generado por un conjunto no vacío de nodos $\mathbf{V^* \subseteq V}$, donde:

(ii) $\mathbf{V^*}$ está fuertemente conectado.

(iii) $\forall \mathbf{V' \subseteq V : (V' \text{ está fuertemente conectado} \wedge V^* \subseteq V') \Rightarrow V^* = V'}$

El conjunto de todos los componentes fuertemente conectados (Strongly Connected Componentes) se denota por **SCC** por su siglas en inglés y se denomina Grafo de Componentes Fuertemente Conectados.

Explicación:

Un **SCC** es un subgrafo del Grafo de Alcanzabilidad, en el cual se especifican los grupos de nodos para los cuales es posible hallar un camino desde uno cualquiera de ellos hacia otro y viceversa.

2.1.3. Propiedades de Alcanzabilidad

Las siguientes propiedades prueban la Alcanzabilidad [4] y son válidas para todo $\mathbf{M1, M2} \in [\mathbf{M_0}\rangle$:

(i) $\mathbf{V} = [\mathbf{M_0}\rangle$

(ii) $\mathbf{M_2} \in [\mathbf{M1}\rangle \Leftrightarrow \mathbf{DPF(M_1, M_2)} \neq \phi$

(iii) $\mathbf{M_2} \in [\mathbf{M1}\rangle \Leftarrow |\mathbf{SCC}| = \mathbf{1}$

Explicación:

La propiedad (i) define si un nodo es o no alcanzable a partir de un marcaje inicial $\mathbf{M_0}$, (ii) establece que un marcaje $\mathbf{M_2}$ es alcanzable a partir de un marcaje $\mathbf{M_1}$ sí y sólo si el Grafo de Alcanzabilidad tiene algún camino dirigido que permita ir de $\mathbf{M_1}$ a $\mathbf{M_2}$, la propiedad (iii) precisa el caso en el cual el Grafo de Alcanzabilidad tiene un único componente fuertemente conectado, en esta situación, todos los marcajes del modelo son alcanzables y mas aún, todos son alcanzables desde cualquier otro marcaje.

2.1.4. Propiedades de Acotación

Sea $\mathbf{X} \subseteq \mathbf{T}$ una transición de una PN y $\mathbf{n} \in \mathbb{N}$ un número entero positivo. Se tienen las siguientes reglas de prueba para la acotación:

(i) **n** es una Cota Superior de marcas para X sí y sólo si: $\forall \mathrm{M} \in [\mathrm{M_0}\rangle : |(\mathrm{M}|\mathrm{X})| \leq \mathrm{n}$

(ii) **n** es una Cota Inferior de marcas para X sí y sólo si: $\forall \mathrm{M} \in [\mathrm{M_0}\rangle : |(\mathrm{M}|\mathrm{X})| \geq \mathrm{n}$

En consecuencia una PN es acotada si el número de marcas en todos los lugares posee una cota superior.

De las definiciones (i) y (ii) se siguen las siguientes reglas de prueba, válidas para todo $\mathbf{X} \subseteq \mathbf{T}$ y todo $\mathbf{p} \in \mathbf{P}$:

(iii) Máxima Cota Superior(**X**): $\mathrm{MCS}(\mathrm{p}) = \mathbf{max}_{\mathrm{M} \in \mathrm{V}} |\mathrm{M}(\mathrm{p})|$

(iv) Mínima Cota Inferior(**X**): $\mathrm{MCI}(\mathrm{p}) = \mathbf{min}_{\mathrm{M} \in \mathrm{V}} |\mathrm{M}(\mathrm{p})|$

Explicación:

La definición anterior establece que una PN es acotada si el número de marcas en cada uno de los lugares es finito, esto es, la red tiene un número máximo de marcas $\mathbf{MCS(p)}$ para cada uno de los lugares, garantizándose así que el Grafo de Alcanzabilidad sea finito.

Si en una PN $\mathrm{MCS}(\mathrm{p}) = \mathbf{max}_{\mathrm{M} \in \mathrm{V}} |\mathrm{M}(\mathrm{p})| = 1$ la red se dice que es 1-acotada y por lo tanto es Segura [17].

Verificando que la red es acotada o segura, se puede garantizar que en el modelo no hay saturación en los lugares y por lo tanto los procesos se realizan de manera efectiva.

La red de Petri de la Figura 81 es acotada ya que el número máximo de marcas en cada uno de los lugares $\mathbf{MCS(p) = 1}$ con ello puede concluirse que la red es 1-acotada.

2.1.5. Propiedades de Reversibilidad

Sea $\mathrm{M} \in \mathbb{M}$ un marcaje y $\mathrm{X} \subseteq \mathbb{M}$ un conjunto de marcajes:

(i) **M** es un *Marcaje Reversible* sí y sólo si: $\forall \mathrm{M}' \in [\mathrm{M}_0\rangle : \mathrm{M} \in [\mathrm{M}'\rangle$

(ii) **X** es un *Espacio Reversible* sí y sólo si: $\forall \mathrm{M}' \in [\mathrm{M}_0\rangle : \mathrm{X} \cap [\mathrm{M}'\rangle \neq \phi$

Explicación:

La propiedad (i) indica que si un marcaje **M** es reversible, entonces siempre es posible ir del marcaje **M’** al marcaje **M**, siendo **M’** un marcaje alcanzable desde el marcaje inicial $\mathbf{M_0}$. Por su parte (ii) indica que para que un conjunto **X** sea un Espacio Reversible este debe tener por lo menos un elemento **M** (pues de lo contrario será un conjunto vacío).

En la figura 3 se encuentra representado un sistema en el cual es posible siempre regresar a cualquier marcaje, todos los marcajes son alcanzables a partir del marcaje inicial $\mathbf{M_0}$ y este a su vez es siempre alcanzable desde cualquier otro, de esta manera no habrán bloqueos en el modelo, todos los marcajes son reversibles y la red es cíclica.

2.1.6. Propiedades de Vivacidad

Sea $\mathbf{M} \in \mathbb{M}$ un marcaje y $\mathbf{X} \subseteq \mathbf{T}$ un conjunto de transiciones:

(i) **M** es un marcaje *Muerto* sí y sólo si ningún elemento se habilita en dicho marcaje, es decir:

$$\forall \mathrm{x} \in \mathrm{T} : \neg \mathrm{M}[\mathrm{x}\rangle$$

(ii) **X** es un elemento *Muerto* en **M** sí y sólo si ningún elemento de **X** se puede habilitar en **M**, es decir:

$$\forall M' \in [M\rangle\ \forall x \in X : \neg M'[x\rangle$$

(iii) X es un elemento vivo sí y sólo si no hay marcajes alcanzables en los cuales X sea muerto, es decir:

$$\forall M' \in [M_0\rangle\ \exists M''[M'\rangle\ \exists x \in X : M''[x\rangle$$

Explicación:

Como consecuencia de la propiedad (i) se dice que un marcaje es muerto, si el nodo correspondiente en el Grafo de Alcanzabilidad es terminal, es decir, no tiene arcos de salida. Las propiedades (ii) y (iii) indican las condiciones en las cuales las transiciones de un modelo resultan ser muertas o vivas, respectivamente.

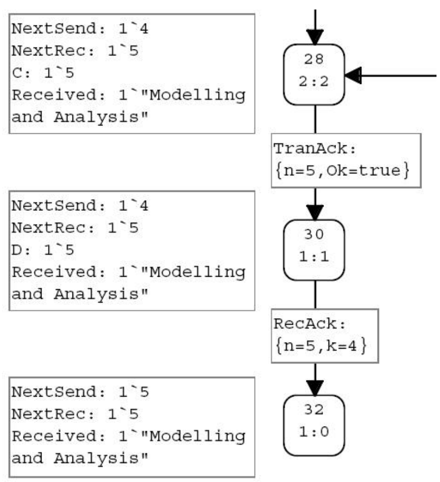

Figura 81. Fragmento de un Grafo de Alcanzabilidad

El subgrafo presentado en la Figura 81 es una porción de un Grafo de Alcanzabilidad. El nodo 32 es un estado de bloqueo ya que no posee arcos de salida, este ilustra claramente lo que es un nodo terminal.

2.1.7. Propiedades de Imparcialidad

Sea $\mathbf{X} \subseteq \mathbf{T}$ un conjunto de transiciones y σ una secuencia infinita de ocurrencias:

(i) **X** es *Imparcial* para σ sí y sólo si **X** tiene infinitamente muchas ocurrencias, es decir:

$$OCC_X(\sigma) = \infty$$

(ii) **X** es *Objetivo* para σ sí y sólo si un número infinito de habilitaciones implica un número infinito de ocurrencias, es decir:

$$ENB_X(\sigma)=\infty \quad \Rightarrow \quad OCC_X(\sigma)=\infty$$

(iii) X es *Justo* para σ sí y sólo si una habilitación de manera persistente genera al menos una ocurrencia, es decir:

$$\forall \mathbf{i} \geq \mathbf{1} : \left(\mathbf{ENB}_{\mathbf{X,i}}(\sigma) \neq 0 \Rightarrow \exists \mathbf{k} \geq \mathbf{i} : [\mathbf{ENB}_{\mathbf{X,k}}(\sigma) = 0 \ \vee \ OCC_{\mathbf{X,k}}(\sigma) \neq 0]\right)$$

Se utilizó la notación $OCC_X(\sigma)$ para denotar el número total de ocurrencias de los elementos de X en σ mientras que el símbolo $ENB_X(\sigma)$ es usado para indicar el número total de habilitaciones de los elementos de X, de igual manera, $\mathbf{OCC}_{\mathbf{X,k}}(\sigma)$ expresa el número de elementos de X que ocurren en el k-ésimo paso y $\mathbf{ENB}_{\mathbf{X,i}}(\sigma)$ expresa el número de elementos de X que se habilitan en el i-ésimo paso.

Las siguientes son consecuencias directas de la anterior definición:

- Como consecuencia de la propiedad (i), una transición es *Imparcial* para una secuencia $\boldsymbol{\sigma}$ de ocurrencia infinita si dicha transición ocurre infinitamente. Esto significa que la transición siempre se habilitará y ocurrirá cuando la red esté evolucionando, así, una transición imparcial sólo dejará de habilitarse y ocurrir cuando el sistema esté detenido.

Las transiciones **T1** y **T6** de la Figura 82 tienen la propiedad de que siempre que la marca presente en **P1** evolucione, estas se habilitarán y ocurrirán. Claramente las dos transiciones están obligadas a ocurrir siempre, en consecuencia, son *Imparciales*.

- De acuerdo con la propiedad (ii), una transición es *Objetiva* si cada vez que esta se habilita, ocurre. Lo anterior no implica que la transición siempre se habilite en una secuencia de ocurrencia determinada.

T3 y **T5** en la Figura 82 son mutuamente excluyentes, de manera que cuando ocurre **T3** no puede ocurrir **T5** y viceversa, sin embargo, cada vez que una de ellas se habilita, ocurre. **T3** y **T5** son entonces Objetivas.

- Según la propiedad (iii), una transición es *Justa* si ante múltiples habilitaciones esta ocurre finitas veces, en consecuencia, no siempre que la transición se habilita, ocurre, pero debe ocurrir al menos una vez en la evolución de la red. Esta propiedad es característica de las transiciones de salida de los lugares donde hay conflicto.

En la PN de la figura 3, **T2** y **T4** se habilitan simultáneamente pero solo una de las dos ocurrirá, de esta manera, es posible por ejemplo que en 100 ciclos completos, **T4** ocurra 99 veces mientras que **T2** lo haga solo una vez. Lo mismo puede suceder en sentido inverso.

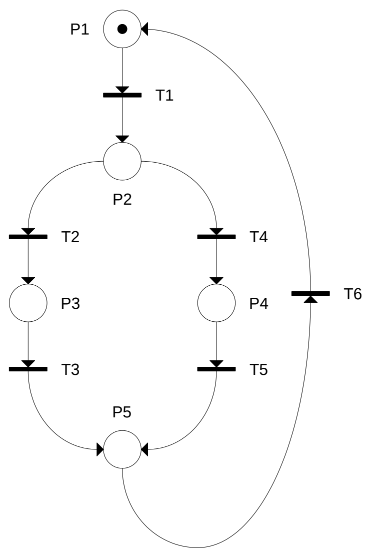

Figura 82. Red de Petri ilustrativa

ANEXO 3

REPORTES DE VALIDACIÓN Y VERIFICACIÓN DE MODELOS

3.1 Estructura del reporte

Estadísticas (Statistics): Muestra información general sobre el número de elementos del Grafo de Alcanzabilidad (Occ Graph) y el grafo de Componentes Fuertemente Conectados (Scc Graph).

Propiedades de Acotación (Boundedness Properties): Detalla la máxima y mínima distribución de marcas en cada uno de los lugares.

Integer Bounds: Indica el número máximo y mínimo de marcas que puede tener cada uno de los lugares de la red.

Multiset Bounds: Indica los valores que pueden tener las marcas en cada lugar (colores asignados a ellas). **Upper** es el máximo multiconjunto de marcas que pueden tener los lugares, mientras que **Lower** es el mínimo.

Propiedades de Reversibilidad (Home Properties): Especifican la capacidad que tiene el modelo para regresar a un estado determinado, siguiendo una secuencia de ocurrencia.

Marcaje Reversible (Home Marking): Muestra a cuáles estados es posible siempre regresar.

Propiedades de Vivacidad (Liveness Properties): Especifica el comportamiento dinámico de los marcajes y las transiciones del modelo.

Marcajes Muertos (Dead Markings): Indica qué marcajes no tienen transiciones habilitadas. En Design/CPN un Marcaje Muerto es, por sus características, un Marcaje Reversible.

Instancias de Transiciones Muertas (Dead Transitions Instances): Detalla cuáles son las transiciones del modelo, que nunca se habilitan para todo marcaje alcanzable. Si una transición se habilita por lo menos una vez para algún marcaje, ésta no es una transición muerta.

Instancias de Transiciones Vivas (Live Transitions Instances): Muestra las transiciones que siempre se pueden habilitar una vez más. Si una transición no es muerta pero se dispara finitas veces, se dirá que es una *transición parcialmente viva*. En este caso el reporte será:

Dead Transitions Instances: None

Live Transitions Instances: None

Propiedades de Imparcialidad (Fairness Properties): Brindan información acerca de la habilitación y ocurrencia de cada una de las transiciones de acuerdo a la semántica del modelo.

En una misma instancia, cada transición de manera inherente posee una de las características de Imparcialidad (Impartial), Objetividad (Fair) o Justo (Just), se explican en más detalle en el Holón Recurso

```
CPN Tools state space report for:
D:\Universidad\Maestria\Ejecución de tesis\Escribir
Tesis\Validación y Verificación\Holon_Recurso.cpn


 Statistics
------------------------------------------------------------------
-----

  State Space
     Nodes:  10
     Arcs:   23
     Secs:   0
     Status: Full

  Scc Graph
     Nodes:  1
     Arcs:   0
     Secs:   0


 Boundedness Properties
------------------------------------------------------------------
-----

  Best Integer Bounds
                             Upper      Lower
     Holon_Recurso'Arranque 1
                             1          0
     Holon_Recurso'Degradado 1
                             1          0
     Holon_Recurso'Disponibilidad_deagradada 1
                             1          0
     Holon_Recurso'Disponible 1
                             1          0
     Holon_Recurso'Falla 1   1          0
     Holon_Recurso'Mtto_Programado 1
                             1          0
     Holon_Recurso'Mtto_preventivo 1
                             1          0
     Holon_Recurso'Normal 1  1          0
     Holon_Recurso'Parada 1  1          0
     Holon_Recurso'Reservado 1
                             1          0
```

```
 Best Upper Multi-set Bounds
    Holon_Recurso'Arranque 1
                        1`m
    Holon_Recurso'Degradado 1
                        1`m
    Holon_Recurso'Disponibilidad_deagradada 1
                        1`m
    Holon_Recurso'Disponible 1
                        1`m
    Holon_Recurso'Falla 1
                        1`m
    Holon_Recurso'Mtto_Programado 1
                        1`m
    Holon_Recurso'Mtto_preventivo 1
                        1`m
    Holon_Recurso'Normal 1
                        1`m
    Holon_Recurso'Parada 1
                        1`m
    Holon_Recurso'Reservado 1
                        1`m

 Best Lower Multi-set Bounds
    Holon_Recurso'Arranque 1
                        empty
    Holon_Recurso'Degradado 1
                        empty
    Holon_Recurso'Disponibilidad_deagradada 1
                        empty
    Holon_Recurso'Disponible 1
                        empty
    Holon_Recurso'Falla 1
                        empty
    Holon_Recurso'Mtto_Programado 1
                        empty
    Holon_Recurso'Mtto_preventivo 1
                        empty
    Holon_Recurso'Normal 1
                        empty
    Holon_Recurso'Parada 1
                        empty
    Holon_Recurso'Reservado 1
                        empty
```

Home Properties

--

```
 Home Markings
    All

Liveness Properties
------------------------------------------------------------------
-----

 Dead Markings
    None

 Dead Transition Instances
    None

 Live Transition Instances
    All


Fairness Properties
------------------------------------------------------------------
-----
      Holon_Recurso'Fin_Mtto_Preventivo 1
                        Fair
      Holon_Recurso'Iniciar 1
                        Just
      Holon_Recurso'Iniciar_1 1
                        Just
      Holon_Recurso'Iniciar_2 1
                        Just
      Holon_Recurso'Mtto 1   Just
      Holon_Recurso'Mtto_1 1 Just
      Holon_Recurso'Mtto_2 1 Fair
      Holon_Recurso'Mtto_Preventivo 1
                        Just
      Holon_Recurso'Poner_Disponibilidad_Degradada 1
                        Just
      Holon_Recurso'Poner_Disponible 1
                        Just
      Holon_Recurso'Poner_Disponible_1 1
                        Just
      Holon_Recurso'Poner_en_Degradado 1
                        Just
      Holon_Recurso'Poner_en_Falla 1
                        Just
      Holon_Recurso'Poner_en_Falla_1 1
                        Just
      Holon_Recurso'Poner_en_Falla_2 1
                        Just
```

```
      Holon_Recurso'Poner_en_Falla_3 1
                        Just
      Holon_Recurso'Poner_en_Falla_4 1
                        Just
      Holon_Recurso'Poner_en_Falla_5 1
                        Just
      Holon_Recurso'Reservar 1
                        Just
      Holon_Recurso'Secuencia_Arranque 1
                        Just
      Holon_Recurso'Secuencia_Parada 1
                        Just
      Holon_Recurso'Sin_Falla 1
                        Fair
      Holon_Recurso'sp_set 1 Just
```

3.2 Misión del recurso

```
CPN Tools state space report for:
D:\Universidad\Maestria\Ejecución de tesis\Escribir
Tesis\Validación y Verificación\Modulo_Mision.cpn
Report generated: Sat Aug 08 21:08:26 2009


 Statistics
------------------------------------------------------------------
-----

  State Space
     Nodes:  8
     Arcs:   18
     Secs:   0
     Status: Full

  Scc Graph
     Nodes:  3
     Arcs:   2
     Secs:   0


 Boundedness Properties
------------------------------------------------------------------
-----

  Best Integer Bounds
                             Upper      Lower
     Mision'Abortado 1       1          0
```

```
     Mision'Avance_Normal 1  1           0
     Mision'Desviada 1       1           0
     Mision'Incumplimiento 1 1           0
     Mision'Incumplimiento_autoriszado 1
                             1           0
     Mision'Riesgo 1         1           0
     Mision'Sin_Compromiso 1 1           0
     Mision'Terminada 1      1           0

  Best Upper Multi-set Bounds
     Mision'Abortado 1   1`m
     Mision'Avance_Normal 1
                         1`m
     Mision'Desviada 1   1`m
     Mision'Incumplimiento 1
                         1`m
     Mision'Incumplimiento_autoriszado 1
                         1`m
     Mision'Riesgo 1     1`m
     Mision'Sin_Compromiso 1
                         1`m
     Mision'Terminada 1  1`m

  Best Lower Multi-set Bounds
     Mision'Abortado 1   empty
     Mision'Avance_Normal 1
                         empty
     Mision'Desviada 1   empty
     Mision'Incumplimiento 1
                         empty
     Mision'Incumplimiento_autoriszado 1
                         empty
     Mision'Riesgo 1     empty
     Mision'Sin_Compromiso 1
                         empty
     Mision'Terminada 1  empty
```

Home Properties

```
------------------------------------------------------------------
-----

  Home Markings
     [3]
```

Liveness Properties

```
------------------------------------------------------------------
-----
```

```
  Dead Markings
     [3]
  Dead Transition Instances
     None
  Live Transition Instances
     None
```

Fairness Properties

```
-------------------------------------------------------------------
-----
       Mision'Autorizacion 1  Just
       Mision'Autorizacion_1 1
                        Just
       Mision'preguntar 1     Just
       Mision's0 1            Fair
       Mision'sa 1            Just
       Mision'sa_1 1          Just
       Mision'sb 1            Just
       Mision'sd 1            Just
       Mision'sd_1 1          Just
       Mision'sd_2 1          Just
       Mision'sd_3 1          Just
       Mision'sn 1            Just
       Mision'sn_1 1          Just
       Mision'sn_2 1          Fair
       Mision'sn_3 1          Just
       Mision'sp_1 1          Just
       Mision'spd 1           Just
       Mision'sr 1            Just
```