REVISION BIBLIOGRAFICA DE LAS TECNICAS QUE RELACIONAN LA LOGICA DE PROGRAMACION DE RED DE PETRI CON LA LOGICA DE PROGRAMACION DE ESCALERA PARA LA IMPLEMENTACION DE SISTEMAS AUTOMATIZADOS DE CONTROL

CARLOS ADOLFO ROBLES SILVA

DIANA CAROLINA JAIMES BALCUCHO

UNIVERSIDAD PONTIFICIA BOLIVARIANA
ESCUELA DE INGENIERIA Y ADMINISTRACION
FACULTAD DE INGENIERIA ELECTRONICA
ESPECIALIZACION EN CONTROL E INSTRUMENTACION INDUSTRIAL
BUCARAMANGA
2011

REVISION BIBLIOGRAFICO DE LAS TECNICAS QUE RELACIONAN LA LOGICA DE PROGRAMACION DE RED DE PETRI CON LA LOGICA DE PROGRAMACION DE ESCALERA PARA LA IMPLEMENTACION DE SISTEMAS AUTOMATIZADOS DE CONTROL

CARLOS ADOLFO ROBLES SILVA

DIANA CAROLINA JAIMES BALCUCHO

MONOGRAFIA DE GRADO

MC.S JUAN CARLOS VILLAMIZAR
DIRECTOR

UNIVERSIDAD PONTIFICIA BOLIVARIANA
ESCUELA DE INGENIERIA Y ADMINISTRACION
FACULTAD DE INGENIERIA ELECTRONICA
ESPECIALIZACION EN CONTROL E INSTRUMENTACION INDUSTRIAL
BUCARAMANGA
2011

Nota de Aceptación

_____
_____
_____
_____
_____
_____

_____
Firma de Presidente del Jurado

_____
Firma del Jurado

_____
Firma del Jurado

Bucaramanga, 19 de Febrero de 2011

*A **Dios***, por darme la vida.

*A mi **novia***, por darme el apoyo.

Padres y Hermana, ejemplos de vida.

Carlos Adolfo

*A **Dios***, por regalarme la vida.

*A mis **Abuelos, Padres y Hermanos***, ejemplos de vida.

*A mi **novio***, la razón de mi vida

Diana Carolina

AGRADECIMIENTOS

Al Ingeniero Juan Carlos Villamizar, docente de la facultad de ingeniería electrónica, por su valiosa orientación, colaboración y respaldo para la realización y desarrollo del presente proyecto.

A todos los docentes de la especialización, por sus enseñanzas, orientaciones y sabios consejos durante el desarrollo de la misma.

A todos los compañeros de la cuarta cohorte de la especialización, por su amistad, colaboración y compañía durante el transcurso de esta especialización.

A todas las demás personas que de una u otra forma ayudaron a la realización y feliz término de la especialización y presente proyecto de grado.

TABLA DE CONTENIDO

Pág.

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE ANEXOS

Pág.

GLOSARIO

- **DECS.** *(Discrete Event Control Systems).* Sistemas de Control de Eventos Discretos.
- **PN.** *(Petri Net). Red de Petri.*
- **PNC.** (*Petri Net Controller*). Controlador de red de Petri.
- **TPL.** (Token Passing Logic). Logica de Paso de Muestra.
- **PLC'S.** *(Programmable Logic Controller).* Controladores Lógicos Programables
- **VLSI.** *(Very Large Scale Intration)* Integración en escala muy grande.
- **CPN.** *(Colours Petri Nets)* Redes de Petri de Color.
- **IEEE.** *(Institute of Electrical and Electronics Engineers).* Instituto de Ingenierias Electrica y Electronica.
- **SFE.** *(Secuencial Functional Chart).* Grafico Funcional Secuencial.
- **DEDS.** (*Discrete Event Dinamic Systems*). Sistemas Dinamicos de Eventos Discretos.
- **CtlPN.** *(Controlled Petri net).* Redes de Petri controladas.
- **LPN.** *(Label Petri Net).* Redes de Petri etiquetadas.
- **FMC.** *(Flexible Manufacturing Cells) Celdas de Manufactura Flexible.*
- **LLD.** *(Ladder Logic Diagram)* Diagrama de Lógica de Escalera.
- **IEC.** *(International Electrotechnical Commission)* Comisión Electrotécnica Internacional.

OBJETIVOS

OBJETIVO GENERAL

Realizar una revisión bibliográfica acerca de la forma de cómo relacionar las redes de Petri con la programación conocida como lógica de escalera, además el avance que actualmente se lleva a cabo en la implementación de la topología de Petri, y la forma de implementar las redes de Petri en el diseño de la lógica en redes industriales.

OBJETIVOS ESPECIFICOS

- Establecer un marco de referencia del estado del arte de la topología de Petri.
- Dar a conocer las técnicas que actualmente se utilizan para llevar los modelos de la redes de Petri hacia el lenguaje de lógica de escalera.
- Concentrar los resultados del estudio de los métodos actuales de la topología de Petri, para mejorar el diseño de la lógica industrial.

RESUMEN GENERAL DE TRABAJO DE GRADO

TITULO: **REVISION BIBLIOGRAFICO DE LAS TECNICAS QUE RELACIONAN LA LOGICA DE PROGRAMACION DE RED DE PETRI CON LA LOGICA DE PROGRAMACION DE ESCALERA PARA LA IMPLEMENTACION DE SISTEMAS AUTOMATIZADOS DE CONTROL**

AUTORES: **CARLOS ADOLFO ROBLES SILVA**
DIANA CAROLINA JAIMES BALCUCHO

FACULTAD: **ESP. EN CONTROL E INSTRUMENTACION INDUSTRIAL**

DIRECTOR: **JUAN CARLOS VILLAMIZAR**

RESUMEN

Las redes de Petri representan el más efectivo método para el diseño y la implementación de sistemas de control de tiempo discreto (DEC'S), ya que cuando se trata del diseño de sistemas de control de alto nivel que exigen la realización de tareas paralelas que interactúan periódicamente entre sí, son este tipo de redes la más efectiva técnica para el diseño de este clase de sistemas. Pero debido a la estandarización y gran popularidad de sistemas de control automático implementados bajo la estructura de los PLC's, los cuales manejan su programación basados en el lenguaje de lógica de escalera (LADDER), el cual a pesar de tener gran simplicidad no ofrece las herramientas necesarias en el desarrollo de sistemas de control complejos, surge la necesidad de encontrar una relación entre el método de diseño mediante redes de Petri y el lenguaje de programación de lógica de escalera. La conversión de tales redes de Petri en aplicaciones en tiempo real recientemente se ha simplificado con la aparición de la metodología de lógica con paso de testigo (TPL). En esta monografía se expondrán los conceptos de las redes de Petri, además se extenderán hacia lo que son los controladores de redes de Petri. Siguiendo, con la metodología TPL, la conversión de un controlador de red de Petri en los diagramas de lógica de escalera y las actualizaciones en las técnicas basadas en las redes de Petri para el control supervisorío de sistemas de eventos discretos.

PALABRAS CLAVE: Redes de Petri, Controladores de redes de Petri, metodología TPL, Lenguaje de Lógica de escalera.

V° B° THESIS DIRECTOR

ABSTRACT OF THESIS PROJECT

TITLE: **TECHNICAL REVIEW OF THE LITERATURE OF LINKING THE LOGIC PETRI NETWORK PROGRAMMING WITH LADDER LOGIC PROGRAMMING FOR THE IMPLEMENTATION OF AUTOMATED CONTROL SYSTEMS.**

AUTHORS: **CARLOS ADOLFO ROBLES SILVA**
DIANA CAROLINA JAIMES BALCUCHO

DEPARTMENT: **SP. CONTROL AND INDUSTRIAL INSTRUMENTATION**

DIRECTOR: **JUAN CARLOS VILLAMIZAR**

ABSTRACT

Petri nets represent the most effective method for the design and implementation of control systems for discrete time (DEC's), because when it comes to the design of high-level control systems that require the completion of parallel tasks that interact regularly each other, such networks are the most effective technique to the design of this class of systems. But due to the standardization and popularity of automatic control systems implemented under the framework of the PLC’s, which handle their programming based on the language ladder, which despite their simplicity does not offer the tools necessary in the development of complex control systems, the need to find a relationship between the design method using Petri nets and programming language ladder. The conversion of such Petri nets in real-time applications has recently been simplified with the advent of methodology token logic (TPL). In this paper outlines the basic concepts of Petri nets also extend to what are the drivers of Petri nets. Continuing with the methodology TPL, the conversion of a Petri net controller in ladder logic diagrams and technical updates in the "Summary" based on Petri nets to supervisory control of discrete event systems.

KEYWORDS: Petri nets, Petri nets controller, TPL methodology, Ladder Logic Language.

V° B° THESIS DIRECTOR

INTRODUCCION

En la industria actual, los PLC's se han convertido en el pilar fundamental en la ejecución de tareas de automatización, esta elección para el control de tareas se debe a su bajo costo, robustez y facilidad de programación. De hecho, la mayoría de los PLC's pueden ser programados en un lenguaje grafico simbólico llamado lógica de escalera (LADDER). La simplicidad de los programas en lógica de escalera los hace muy transparentes pero al mismo tiempo es su mayor debilidad, esto es porque cuando se desarrollan sistemas de control complejos estos implican tareas paralelas que interactúan periódicamente y este lenguaje de programación ofrece pocos caminos para la construcción estructural de este tipo de controladores. Surge así la necesidad de una herramienta de diseño efectivo para producir sistemas de control de tiempo discreto (DEC'S) de altos niveles. Las redes de Petri han aparecido como la herramienta más prometedora para facilitar este trabajo de diseño. Sin embargo, los métodos de las redes de Petri no han sido utilizados para el diseño directo de los programas de PLC's, esto se debe que hasta la llegada de la lógica TPL, no existía ninguna técnica para convertir las redes de Petri en un formato adecuado para la aplicación de un PLC. Esta técnica es de gran alcance y sin embargo fácil de comprender y aplicar. Por otra parte, la técnica también se ha ampliado para hacer frente a las redes de Petri de P-timed, redes de Petri de T-timed y redes de Petri de colores.

En la primera sección de este documento se expone un marco conceptual acerca de la teoría básica de las redes de Petri, definición, características y propiedades de diseño, así como también de la teoría básica de lo que tiene que ver con el diagrama de Lógica en escalera. En la segunda sección se realiza una revisión cronológica de distintos trabajos que han contribuido al desarrollo de los diagramas de programación en redes de Petri y la relación que estos pueden tener con la metodología de diagrama en escalera para la implementación de sistemas automatizados. Y por último en la tercera sección se explica un conjunto de características adicionales para las redes de Petri ordinarias que facilitan la transformación de este tipo de programación en la lógica de programación de diagramas de escalera básicamente trabajando con la herramienta de controladores de redes de Petri y la metodología TPL (Token Passing Logic).

En resumen el propósito de este trabajo es presentar una serie de teorías y técnicas adicionales para las redes de Petri que faciliten la transformación de este tipo de lógica de programación, en la lógica de programación de diagrama de escalera para la implementación de sistemas de control avanzados y además exponer las actualizaciones en la parte teórica y práctica de la automatización basada en las redes de Petri.

1. INGENIERIA CONCEPTUAL

1.1. FUNDAMENTACION TEORICA DEL PROYECTO.

1.1.1. Redes de Petri.

a. Definición.

Las redes de Petri fueron definidas en los años 1960 por Carl Adam Petri. Son una generalización de la teoría de autómatas que permite expresar eventos concurrentes. Y se definen como uno de los varios lenguajes de modelación matemática de sistemas distribuidos. Esta red es un grafo bilateral dirigido, formado por lugares, transiciones y arcos dirigidos, así como por muestras que ocupan posiciones, como se observa en la Figura 1. Las transiciones (es decir, acontecimientos que pueden ocurrir, son representados por las barras rectangulares), los lugares (es decir, las condiciones), son representados por los círculos y los arcos dirigidos describen los lugares que son pre- y/o post-condiciones para cada una de las transiciones (estos se simbolizan con las flechas). [1]

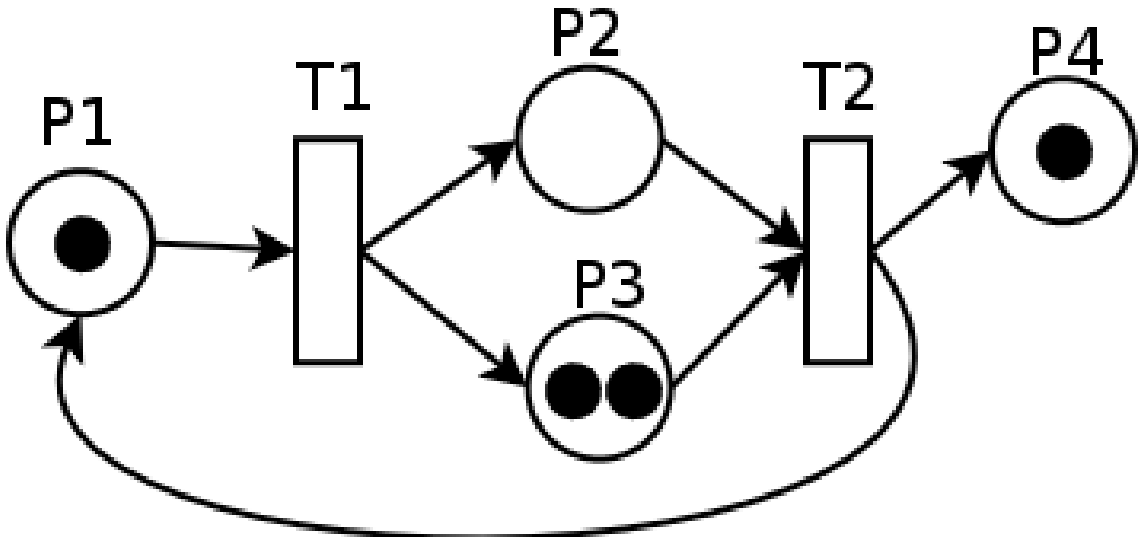

Figura 1. Diagrama de una red de Petri [1]

Los arcos conectan un lugar a una transición o una transición a un lugar. No puede haber arcos entre lugares ni entre transiciones. Los lugares contienen un número cualquiera de muestras. Los lugares en los cuales un arco se ejecuta a una transición son llamados los lugares de entrada de la transición. Las transiciones se disparan, es decir consumen muestras de una posición de inicio y

producen muestras en una posición de llegada. Una transición está habilitada si tiene muestras en todas sus posiciones de entrada. [1]

Formalmente, una Red de Petri se define como una quíntupla:

$$PN = (P, T, F, W, M_0) \quad (1.1)$$

Donde:

- P = {p1,p2,…,pm} es un conjunto finito de lugares.
- T= { t1,t2,…,tn } es un conjunto finito de transiciones.
- F $\subseteq$ (P ×T) **U** (T ×P) es un conjunto de arcos dirigidos.
- W : F→ {1, 2, 3, …} es una función de pesos de los arcos.
- M_0 : P→ {1, 2, 3, …} es el marcado inicial de la red.
- P ∩ T= ∅ y P **U** T≠ ∅.

El marcado inicial de una red de Petri son las muestras que posee cada lugar de la red en su inicio. Una Red de Petri con la estructura (P, T, F, W) sin especificar su marcado inicial es denotada por N. Por otro lado, una red de Petri con un marcado inicial dado es denotado por PN=(N, M0). El comportamiento de los sistemas puede ser descrito en términos de sus estados y sus cambios. En las Redes de Petri, el estado del sistema, o mejor dicho, el marcado cambia de acuerdo con las siguientes reglas de disparo o transición:

- Se dice que una transición es habilitada si cada lugar de entrada *p* de *t* es marcada con al menos *w(p,t)* muestras, donde *w(p,t)* es el peso del arco de *p* a *t*.
- Una transición habilitada puede o no ser disparada (esto depende solamente del carácter no determinista del evento).
- El disparo de una transición *t* habilitada remueve *w(p,t)* muestras de cada lugar de entrada *p* de *t* y agrega *w(t,p)* muestras a cada lugar de salida *p* de *t*, donde *w(t,p)* es el peso de los arcos de *t* a *p*.

En su forma más básica, las muestras que circulan en una red de Petri son todas idénticas. Se puede definir una variante de las redes de Petri en las cuales las

muestras pueden tener un color (una información que las distingue), un tiempo de activación y una jerarquía en la red.

Podemos decir entonces que mediante una red de Petri puede modelarse un sistema de evolución en paralelo compuesto de varios procesos que cooperan para la realización de un objetivo común. [32]

b. Propiedades de las Redes de Petri

Una de las principales fortalezas que tienen las redes de Petri tiene que ver con sus propiedades, las cuales se dividen en dos tipos, las que son dependientes de una marcación inicial y son llamadas propiedades dinámicas y las que no dependen de ningún marcado en su inicio y son las propiedades estructurales o de estática. A continuación se nombrarán y expondrán estos dos tipos de propiedades:

- **Propiedades dinámicas**

- *Alcanzabilidad*: esta es la principal propiedad dinámica y consiste en que cada disparo de una transición habilitada modifica la distribución de las muestras dentro de la red, de acuerdo con las reglas de disparo. Una secuencia de disparos generará una secuencia de marcados. Se dice que un muestra Mn es alcanzable desde la M_0 si y sólo si existe una secuencia de disparos que transforme M_0 en Mn. La secuencia de disparos se denota por sigma:

$$\sigma = M_0\ t_1\ M_1\ t_2\ M_2\ \ldots t_n\ M_n \qquad (1.2)$$

En este caso, M_n es alcanzable desde M_0 por σ, lo cual se escribe de la siguiente forma: $M_0[\sigma> M_n$. El conjunto de todos los marcados posibles a partir de M_0 es denotado por $R(N,M_0)$ y el conjunto de todos los posibles disparos desde M_0 es denotado como $L(N,M_0)$. El problema de alcanzabilidad consiste en encontrar un $M_n \in R(N,M_0)$. [32]

- *Limitable o Acotada*: una red $PN=(N,M_0)$ se dice limitada si el número de marcas de la red en cada lugar no excede un número finito *k* para cualquier marcado alcanzable desde M_0 y existirá dentro de todos los posibles marcados de la red, $M(p) \leq k$, y $M(p) \in R(N,M_0)$. Se dice que una red es segura si es acotada a uno, esto es si todos los marcados posibles de los lugares poseerán a lo más una marca. [32]

- *Vivacidad*: Se dice que una red es viva si no importa cuál marcado haya sido alcanzado, siempre será posible una nueva secuencia σ de disparos y alcanzar un nuevo marcado. Esta propiedad lo que indica es que una red viva garantiza una operación libre de bloqueos. Esta propiedad es deseable en la ejecución de programas. [32]

- *Reversibilidad y estado inicial*: De una Red de Petri (N,M_0) se dice que es reversible si para cada marcado M_n existente dentro de $R(N,M_0)$, M_0 es alcanzable desde M_n. De esta forma una red de Petri reversible es aquella donde siempre es posible alcanzar nuevamente el marcado inicial o estado inicial del sistema. [32]

- *Cobertura*: Un marcado M dentro de una Red de Petri (N,M_0) en un conjunto de marcados cubiertos o contenido, si existe un marcado M' dentro de $R(N,M_0)$ tal que $M'(p) \geq M(p)$ para cada lugar *p* dentro de la Red. [32]

- *Persistencia*: Una Red de Petri es persistente si para cualquiera de dos transiciones habilitadas, el disparo de una transición no deshabilitará a la otra transición. [32]

- *Distancia Sincrónica*: Es una métrica asociada al grado de dependencia mutua entre dos eventos en un sistema condición/evento. La distancia sincrónica de las transiciones t1 y t2 en la red de Petri $PN=(N,M_0)$ está dada por:

$$d_{12} = \max \left| \sigma(t1) - \sigma(t2) \right| \qquad (1.3)$$

Donde σ es una secuencia de disparos iniciando en cualquier marcado M perteneciente a R(N, M0) y $\sigma(t_i)$ es el número de veces que una transición t_i es disparada en σ. [32]

- **Propiedades Estáticas:**

Estas propiedades estáticas de las redes de Petri sólo aplican a redes de Petri ordinarias y puras:

- *Vivacidad Estructural:* Una Red de Petri es estructuralmente viva si tiene un marcado inicial para N. [32]

- *Controlabilidad*: De una red de Petri, se dice que es completamente controlable si cualquier marcado es alcanzable desde cualquier otro marcado. Para una red

completamente controlable se cumple que, Rango (A) = m, donde *m* es la cantidad de lugares de la red. [32]

- *Limitación o acotado estructural*: Una red de Petri es limitada estructuralmente si es limitada para cualquier conjunto finito de marcados iniciales M_0. [32]

- *Conservabilidad*: Una red de Petri es conservativa si existe un entero positivo *y(p),* para cada lugar *p* tal que la sumatoria de marcas sea constante para cada M ϵ $R(N,M_0)$. [32]

- *Repetibilidad*: Una red de Petri es repetible si existe un marcado M_0 y una secuencia de disparos σ desde M_0, tal que las transiciones se disparan infinitamente en la secuencia definida por σ. [32]

- *Consistencia*: Una red de Petri es consistente si existe un marcado M_0 y una secuencia de disparos reversible σ desde M_0 hacia M_0, tal que cada transición haya sido disparada al menos una vez en σ. [32]

c. Tipos de redes de Petri

Las redes de Petri enfocadas hacia el aspecto de modelamiento de sistemas de automatización se pueden dividir en redes de Petri de P-timed, redes de Petri de T-timed y redes de Petri de Colores. A continuación se expondrán cada uno de estos modelos.

- Redes de Petri de P-timed: este tipo de redes de Petri son aquellas en las cuales una acción o acciones son asignadas a lugares por tiempos finitos. [31]

- Redes de Petri de T-timed: son aquel tipo de redes de Petri en las cuales una acción o acciones son asignadas a una transición durante un tiempo finito. [31]

- Redes de Petri de Colores: este tipo de redes se caracterizan por incorporar marcas de diferentes colores permitiendo la circulación de diferentes tipos de muestras por la red, en donde, los colores pueden ser vistos como tipos de datos, variables o valores. Por tal motivo en este tipo de redes se debe especificar el dominio de color para cada lugar y la información asociada a cada arco de la red. [28]

d. Aspectos de Modelación con redes de Petri.

Hay dos tendencias en el modelado de Redes de Petri que se enfocan en diferentes perspectivas: condiciones en los lugares y eventos en las transiciones, o condiciones en las transiciones y eventos en los lugares. Aunque no hay un consenso de cuál usar y cuál presenta las mayores ventajas, la preferencia es la de utilizar la primera, es decir, representar los elementos pasivos como lugares y elementos activos como transiciones. [28]

Las redes de Petri han tenido gran aceptación entre la comunidad científica, debido a su permisividad y generalización que han dado pie a diversas interpretaciones cuando se hace el modelado de los sistemas; por ejemplo, si usamos el concepto de condiciones y eventos, los lugares representan condiciones y las transiciones representan eventos, entonces, una transición tiene un cierto número de lugares de entrada y salidas que representan, respectivamente, las precondiciones y poscondiciones del evento, la presencia de una marca en un lugar es interpretada como una condición verdadera asociada al lugar; otra interpretación podría ser, que el número de marcas en un lugar indican la cantidad de recursos disponibles, tal como se muestra en la Tabla 1. [28]

LUGARES DE ENTRADA	TRANSICIONES	LUGARES DE SALIDA
Precondiciones Datos de entrada	Eventos Procesamiento o Computo	Poscondiciones Datos de salida
Señales de entrada Recursos necesarios	Procesador de señales Tarea o Trabajo	Señales de salida Recursos liberados
Condiciones Buffers	Clausula lógica Procesador	Conclusiones Buffers

Tabla 1. Representación de las marcas en los lugares [28]

1.1.2. Lenguaje de Lógica de Escalera (LADDER)

a. Definición del Lenguaje de Lógica de Escalera (LADDER)

El lenguaje de lógica de escalera es un lenguaje de programación gráfico basado en los esquemas de lógica de hardware que utilizan como estructura básica los relés. Este tipo de programación tiene su mayor utilización en la programación de los bien conocidos PLC's utilizados para las operaciones de control industrial. Este

nombre de lenguaje de lógica de escalera, se basa en la observación de que los programas en este lenguaje se asemejan a las escaleras, con dos rieles verticales y una serie de peldaños horizontales entre ellos como se observa en la Figura 2. [2]

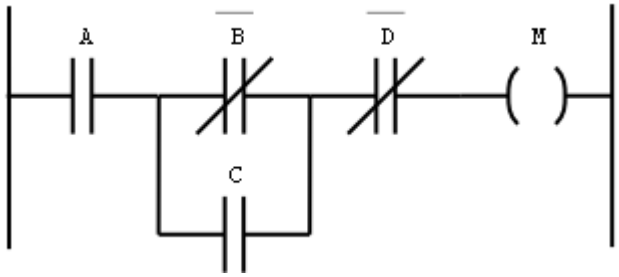

Figura 2. Ejemplo de un diagrama de Lógica de Escalera. [2]

La lógica de escalera es ampliamente utilizada para los autómatas programables, donde se requiere el control secuencial de una operación de proceso o fabricación. A menudo, el programa de lógica de escalera se utiliza en combinación con una interfaz hombre-máquina en las estaciones de trabajo. [2]

Los fabricantes de controladores lógicos programables generalmente proporcionan sistemas de programación de lógica de escalera. Normalmente los lenguajes de lógica de escalera de dos fabricantes diferentes no serán completamente compatibles. Entonces la lógica de escalera es mejor concebida como un conjunto de lenguajes de programación estrechamente relacionados en lugar que un solo lenguaje. La norma IEC 61131-3 ha contribuido a reducir las diferencias innecesarias pero la traducción de los programas entre los sistemas todavía requiere mucho trabajo. Incluso los diferentes modelos de autómatas programables dentro de una misma familia pueden tener diferente notación de programación en escalera de tal manera que los programas no pueden ser perfectamente intercambiables entre los modelos. En resumen la lógica de escalera puede ser pensada como un lenguaje basado en reglas en lugar de un lenguaje de procedimiento. [2]

b. Propiedades del Lenguaje de Lógica de Escalera (LADDER)

El lenguaje de escalera puede ser visto como un conjunto de conexiones lógicas entre conectores (contactos) y actuadores (bobinas). En la lógica de escalera los contactos abren o cierran circuitos o bobinas de control. Cada bobina o contacto corresponde al estatus de un bit simple en la memoria de un controlador programable. A diferencia de los relés electromecánicos, en la programación de

diagrama de escalera, el estado de un bit se puede referir cualquier número de veces, lo cual es equivalente a un relé con indefinido número de contactos. Si un camino puede ser trazado entre el lado izquierdo del renglón y la salida, a través de contactos acertados (abiertos o cerrados), el renglón es verdadero y el bit de memoria de la bobina de salida se afirma (1) o es verdadero. Si un camino no puede ser trazado entonces la salida es falsa (0) y la "bobina", por analogía, a los relés electromecánicos se considera "desernergizada". Cada renglón de este lenguaje generalmente tiene una bobina en el extremo derecho, aunque algunos fabricantes de PLC's pueden permitir más de una bobina de salida en un reglón de su programación. Esta bobina a la salida puede representar una salida física que opera algún dispositivo conectado al controlador programable, o puede representar un bit de memoria interna para ser usado en cualquier otro lugar en el programa. Los llamados contactos se pueden referir a las entradas físicas en los controladores programables tales como pulsadores o switches de límite en un módulo integral o externo, o pueden representar el estado de los bits de memoria interna que se pueden generar en cualquier parte del programa. [2]

Los símbolos básicos que se utilizan para realizar una programación de diagrama de escalera son los siguientes:

- --( )-- , bobina regular, energizada siempre que su renglón este cerrado.
- --(\)-- , bobina negada, energizada siempre que su renglón este abierto.
- --[ ]-- , un contacto regular, cerrado siempre que su correspondiente bobina o entrada de control esté energizada.
- --[\]-- , un contacto negado, abierto siempre que su correspondiente bobina o entrada de control esté energizada. [2]

La programación en lenguaje de escalera es la más adecuada notación para diseñar soluciones a problemas de control donde sólo son requeridas variables binarias y en donde la interconexión y secuencia de binarios es el problema principal de control. Desde que la ejecución de renglones sea secuencial dentro de un programa y pueda ser indefinida dentro de un escalón, algunas condiciones lógicas de escaleras son posibles lo cual puede producir resultados inesperados. También los escalones complejos pueden ser divididos mejor en varios pasos sencillos para evitar este problema.

El diagrama de escalera no es muy útil en lo que tiene que ver con el manejo de cantidades analógicas y operaciones aritméticas, por eso cada fabricante de controladores programables tiene su propia forma de extender la notación de escalera para darle solución a estas situaciones. Por lo general, un apoyo limitado

para las matrices y los bucles, a menudo resulta en la duplicación de código para expresar los casos en los cuales en otros lenguajes exigiría el uso de variables indexadas. [2]

1.2 EVOLUCION DE LAS REDES DE PETRI Y SU IMPACTO EN LOS PLCs.

Las Redes de Petri tienen más de 48 años de existencia desde su formulación en 1962 y durante este periodo se han realizado miles de publicaciones. En esta sección se presentan en forma cronológica distintos trabajos que contribuyen al desarrollo de programas para PLCs.

Uno de los primeros trabajos fue realizado por J. Peterson [24] y publicado en 1977, y consiste en un repaso histórico de las Redes de Petri en sus primeros quince años. En este se expone que las Redes de Petri son un mecanismo de modelado matemático donde los gráficos están constituidos por dos tipos de nodos y arcos con pesos asociados. Estas redes poseen propiedades estáticas y dinámicas, donde las propiedades estáticas se asocian a la topología de la red y las dinámicas a la ejecución y evolución de las marcas en la red. En este trabajo aparecen numerosos ejemplos sobre Red de Petri aplicados a algoritmos. Peterson expone por qué las redes de Petri se observan como una secuencia de eventos discretos cuya orden de ocurrencia puede ser una de muchas posibilidades permitidas. Esto tiene como característica ser aptas para modelar eventos no deterministicos. Por último se detallan los tipos de Redes de Petri.

En 1981, Peterson [25] formula las bases para la teoría de comunicación entre componentes asíncronos de un sistema computacional, la relación entre evento. Al mismo tiempo este autor trabaja las técnicas de análisis para las redes de Petri: el árbol de cobertura, también llamada enumerativa, consiste esencialmente en la enumeración de todos los marcados mediante la generación del grafo de alcanzabilidad para sistemas acotados o del grafo de cobertura para sistemas no acotados. Estas técnicas son aplicables a todas las clases de redes aunque en la práctica son restringidas a redes relativamente pequeñas debido a que un árbol de cobertura de gran tamaño imposibilita un cálculo efectivo; y la matriz de ecuaciones, también llamada estructurales, las cuales tienen como objetivo obtener la máxima información del modelo utilizando su estructura y marcado inicial, investigando la relación entre el comportamiento de la red y su estructura. La característica más importante de estas técnicas es que pueden ser fácilmente implementadas por medio de ordenadores, permitiendo el análisis automático de los sistemas modelados; así mismo, señala que estas técnicas proveen de mecanismos de solución a los principales problemas asociados a las propiedades.

Para el siguiente año, J. Martinez y M. Silva, [17] plantean que las redes de Petri se pueden implementar mediante las técnicas programables con PLC. En esta investigación se centralizan en las técnicas cableadas, donde las redes de Petri utilizan el lenguaje de controladores lógicos como bloques funcionales

parametrizables, por tanto estas se convierten en redes binarias. Este artículo fue precursor de las redes de Petri como alternativa de diseño para algoritmos de PLC. En este mismo año T. Murata [21], establece a través de su trabajo que los dispositivos o estructuras modelados mediante las redes de Petri, se simulan y se validan mediante una herramienta formal de análisis, como los grafos de ocurrencia.

En 1989, siendo muy importante el aporte en la investigación, por parte de las redes de Petri, se publica un tutorial hecho por Murata [22], donde las redes de Petri representan una alternativa grafica y matemática para el modelado de sistemas de información paralela, concurrentes, asíncronos, no-deterministicos, distribuidos y/o estocásticos. En este trabajo se mencionan áreas de aplicación, donde se destacan: medición de rendimiento, protocolos de comunicación, base de datos distribuidas, software paralelo, lógica programable VLSI, circuitos síncronos, estructuras asíncronas, compiladores y sistemas operativos, sistemas informáticos de oficina, lenguajes formales, programas lógicos, redes locales, redes neuronales, filtros digitales y modelos de decisión. Por otra parte menciona siete artículos sobre sistemas de control industrial y sobre modelado de sistemas de eventos discretos, dentro de uno de estos artículos, se explica que el modelado de Redes de Petri utiliza el concepto de condiciones y eventos, los lugares representan las condiciones y las transiciones los eventos. Una transición o evento tiene un número determinado de lugares de entrada y lugares de salida. Este artículo finaliza con una exposición teórica de las redes temporizadas utilizadas en la evaluación de rendimientos de sistemas, de las redes estocásticas y de las redes de alto nivel conocidas como Redes de Petri Coloreadas (CPN), estas últimas pertenecen a la familia de las redes de Petri, la diferencia viene marcada por las consideraciones en las redes de Petri de colores y de funciones lineales asociadas a sus arcos. Las marcas de color pueden representar un atributo o distintivo, si es necesario definir dos atributos entonces surge la idea de colores compuestos. Una transición en CPN está en estado ENABLED si todos sus nodos de entrada contienen un número de colores igual o mayor que los definidos por fi<c> donde fi es una función lineal asociada al nodo pi con la transición tj. Entonces además del concepto de color, estas redes manejan una función asociada para los elementos de las funciones IO de la Red de Petri. Estas redes son utilizadas para la lógica de programas. Así mismo en este mismo año, Pessen [22] publica en su libro el método cascada, el cual se basa en crear un dispositivo de mando que tenga tantas salidas como fases a desarrollar en la secuencia, entendiendo como fase un grupo de letras de la secuencia en las que no se repite ninguna. Este diseño es también llamado diagrama de escalera y tiene como aplicación el desarrollo de un lenguaje de programación para controladores lógicos programables. El método propuesto no garantiza que el circuito sea mínimo en la

cantidad de componentes que utiliza, sino que las soluciones sean fácilmente obtenidas para problemas complejos. Pesswen, así mismo desarrolla una implementación práctica en contactos y relés de una Red de Petri ordinaria, binaria, segura, acotada que es fácilmente implementada en PLC.

En 1990, Holloway [9] propone una nueva red llamada red de Petri controlable, en la cual cualquier muestra es alcanzable desde cualquier otra muestra y se utiliza para modelar el sistema dinámico de eventos discretos. Con este modelo se sintetiza el controlador utilizando un algoritmo para garantizar que el sistema no entre en estados prohibidos. En este trabajo, el objetivo es sintetizar el controlador apropiado. En conclusión Holloway [9] establece que la lógica del control se puede separar del modelo del sistema, introduciendo el concepto de lugares de control externos. Schruben [29] en 1992, mencionó que los métodos gráficos con pocos tipos de objetos son más fáciles de aprender que aquellos con muchos tipos de objetos, estos modelos no garantizan que sean más poderoso que los que tienen pocos objetos. Algunos de los métodos usados son: los gráficos marcados, las redes de procesos y las maquinas de estado. En el periodo de dos años, se realizo paralelamente a la búsqueda de los métodos de diseños formales, la búsqueda de herramientas para verificar programas de PLC. Lo anterior fue desarrollado por Moon [19] en 1994, que definió un método para verificar la operabilidad y seguridad de los PLCs, inspirándose en el método de prueba de protocolos de comunicación y de circuitos. Este método utiliza una representación booleana del programa y fue desarrollado por la carencia de métodos formales para la verificación de programas en PLC. Moon hizo un recuento de los distintos lenguajes para programar PLC, entre ellos, el diagrama escalera y los bloques de funciones; además, efectuó un repaso del funcionamiento y del ciclo de trabajo o de enfriamiento del PLC.

Cai, en 1995 [4] propuso el diseño de sistemas de control secuencial utilizando los invariantes *p,* de las Redes de Petri que son el conjunto de lugares tales que la suma ponderada de sus marcas es constante. Los autores tomaron una Red de Petri y la dividieron en módulos. Cada modulo debe cumplir la propiedad de ser invariante *p*. Los autores brindan una solución grafica a un ejemplo de llenado de tanques de agua. Sin embargo, la solución obtenida no es corroborada en forma algebraica, ante lo cual citan que la realizaran en posteriores trabajos. En ese mismo año, Pettersson en 1995 [25] propone un modelo hibrido general, al separar la planta a lazo abierto para construir un controlador hibrido. El sistema hibrido puede ser modelado por una red de Petri hibrida. El artículo desarrolla los conceptos de los sistemas continuos y de los sistemas de eventos discretos, y propone un conjunto de ecuaciones para modelar un sistema que combine los sistemas discretos como híbridos. Asimismo, definieron las propiedades de un

controlador hibrido y como sintetizarlo, a la vez que presentaron las Redes de Petri hibridas como un mecanismo para modelado de sistemas híbridos y realizaron un ejemplo de modelado de un proceso industrial.

David [8] en 1995, publicó un estudio comparativo entre las Redes de Petri y el Grafico Funcional Secuencial (SFC), como herramientas para especificar algoritmos para controladores lógicos programables. La importancia del estudio radica en que presenta la primera comparación de un formalismo matemático con un estándar europeo ampliamente aceptado, publicado en la IEEE. En el estudio se concluye que el Grafcet o SFC se puede ver como una Red de Petri del tipo binaria, acotadas donde las transiciones son disparadas en forma determinista. En el artículo se realiza una exposición del estándar SFC y del algebra de eventos, así como la solución para la automatización de unos tanques de almacenamiento de agua. La solución es obtenida con los métodos de redes de Petri, SFC, máquinas de estado y tablas de estado; al final se realiza una comparación entre las soluciones para determinar cuál método de diseño fue el menos elaborado.

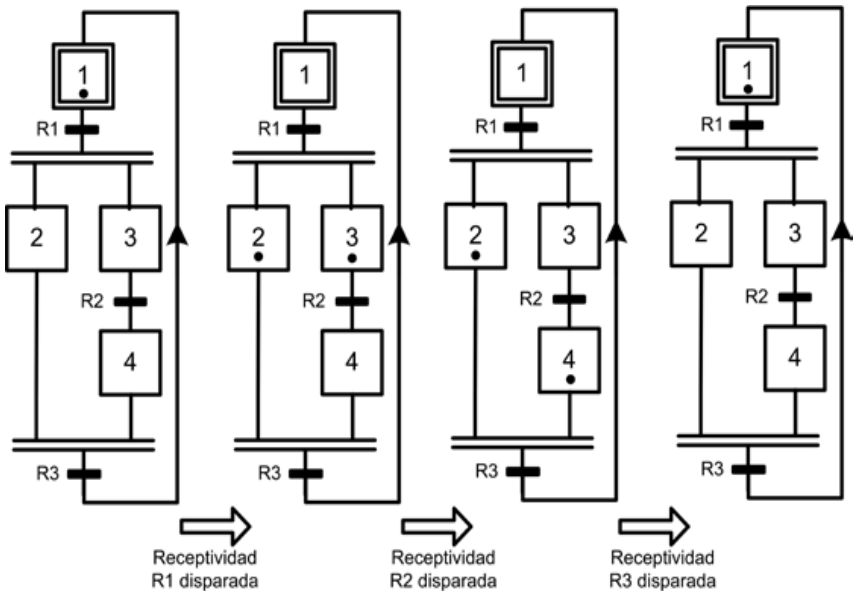

Figura 3. Cuatro SFC donde muestra la evolución de las marcas en las etapas, cuando se valida la receptividad de una transición. [32]

En 1997, vuelve Holloway [9], haciendo un estudio sobre tres importantes corrientes de investigación para controladores:

a) Aproximación de comportamiento controlado.
b) Aproximación de controlador lógico.
c) Aproximación teórica de control.

La aproximación de comportamiento controlado se refiere a modelos con Redes de Petri que describen el comportamiento del sistema a lazo cerrado, integrando tanto la planta como el controlador y cuando se obtiene el comportamiento deseado, se extrae la lógica del controlador para su implementación. La aproximación de controlador lógico consiste en diseñar en forma directa el controlador de la planta, pero sin modelar la planta. El objetivo es definir el comportamiento de las entradas–salidas del controlador para obtener un comportamiento controlado del sistema a lazo cerrado. El Grafcet, es un ejemplo de esta aproximación. La aproximación teórica de control proviene del prototipo de control automático, donde proporcionado el modelo dinámico de una planta y las especificaciones deseadas del lazo de control, se sintetiza un controlador que cumpla con esas especificaciones. El modelo con autómatas finitos no es tan compacto como los modelos con PN, y estos últimos se vuelven una alternativa para el modelado de DEDS.

Holloway, Krogh y Giua plantearon en su investigación como sintetizar los controladores para plantas modeladas por Redes de Petri. Exponen dos grandes aproximaciones: el control retroalimentado de estado o "*state feedback control*" que utiliza las Redes de Petri Controladas (CtlPN), en este, el control de entrada es una función del estado actual del sistema y el objetivo es sintetizar un supervisor y una política de realimentación lo cual garantiza que el sistema no podría entrar en un estado prohibido, y el control retroalimentado de eventos "*event feedback control*" que utilizan las llamadas Redes de Petri Etiquetadas (LPN). Finalmente, los autores desarrollan la teoría del control supervisor y el diseño del supervisor, así como las propiedades de este tipo de control con Redes de Petri. Ellos también, resaltan que hasta la fecha no existe una completa y formal comparación entre las soluciones obtenidas por métodos basados en PN y las basadas en autómatas finitos.

En 1997, Taholakian [10] presenta una metodología para diseñar, simular y codificar sistemas de control basados en PLC usando Redes de Petri, llamada PN⇔PLC. Debido a la complejidad creciente de los sistemas de manufactura, específicamente, de las celdas de manufactura flexible (FMC), el autor implementa la red en un lenguaje de programación estándar definida dentro del estándar de programación, mediante la implementación con diagramas escalera. También menciona que la automatización de una FMC o de sistemas más complejos no se podría realizar si las tareas de control no se distribuyeran entre dispositivos de control distribuido (DCS). Este método desarrollado, es un método grafico que consiste en descomponer una PN ordinaria y segura, en secciones que son mapeadas en forma casi directa a una estructura de contactos de diagrama

escalera. Al final, la unión de las distintas secciones en el diagrama escalera conforma un circuito cuyo comportamiento es equivalente al descrito en la Red de Petri.

G. Mušič y D. Matko en 1998 [11] trabajaron la línea de investigación de la aproximación teórica de control anteriormente descrita, donde se establece el control supervisor mediante una PN flexible para un proceso industrial por lotes. Este trabajo realiza un recuento histórico del control supervisor de sistemas DES que originalmente se baso en la teoría de autómatas finitos, cuya teoría de control supervisor pretende asegurar que la planta nunca alcance ciertos estados, ya sean inseguros o de bloqueo. El método que plantean se basa en el cálculo de los invariantes *p* conocidos como *S-invariant* o *P-Invariant*. La particularidad de este trabajo es que implementa el control supervisor con SFC, que es un lenguaje estandarizado de programación de PLC. G.Frey [6] en ese mismo año propone la utilización de una nueva clase de PN llamadas Redes de Petri Interpretadas (IPN) (Figura 4) organizadas y estructuradas dentro de niveles jerárquicos, para la verificación y validación de algoritmos de control.

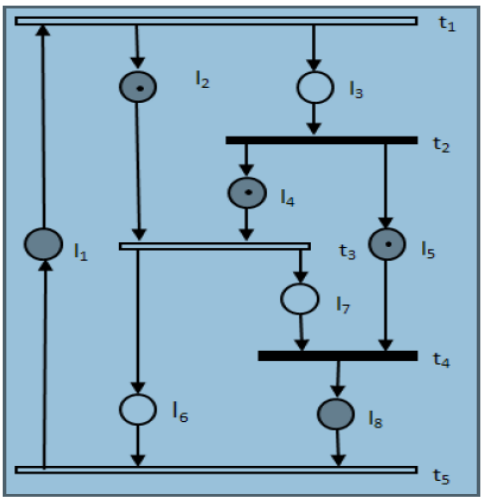

Figura 4. Red de Petri Interpretada [32]

Las IPN asocian los estados del sistema a los lugares y establecen condiciones booleanas y eventos en las transiciones. El nombre “Interpretadas por periferia” se usa para reforzar el hecho de que la influencia del ambiente en el sistema esta basada en las señales de entrada y salida y que los eventos externos se detectan en los cambios de estado de las señales binarias de entrada. En algunos trabajos, la interpretación asocia a cada transición una receptividad, la cual se define como el producto de un evento por una condición booleana. En un controlador lógico, la secuencia de acciones de control es generada por la dinámica de evolución del modelo PN. Para ilustrar mejor las IPN se muestra en la Figura 5. un módulo de

posicionamiento horizontal de un carro sobre dos tanques, incluyendo el automatismo del manejo de la grúa que manipula verticalmente un elemento sobre ellos.

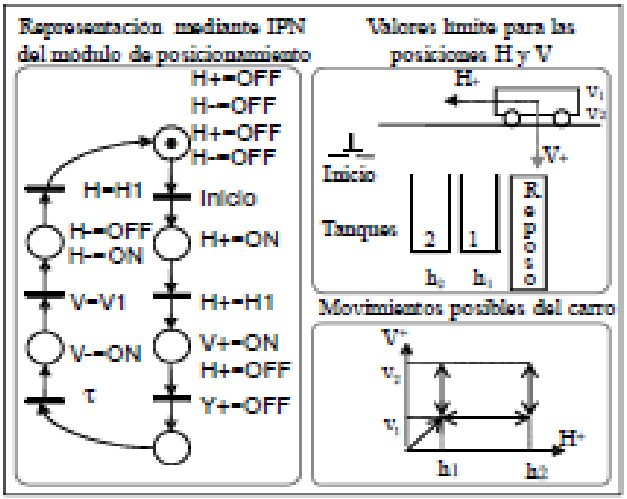

Figura 5. Modulo de posicionamiento mediante IPN [32]

Ya con los métodos estudiados en los anteriores años y los sistemas de fabricación tornándose muy complejos, la necesidad de producir una efectiva herramienta de automatización en Sistemas de control de eventos discretos se vuelve cada vez más importante. Por lo tanto en 1998, M. Uzam [13] realizo el diseño de un sistema de control de eventos discretos y su implementación a partir de una nueva clase de PN que define y llama APN (*Automation Petri Nets*). La (APN), es un nuevo formalismo para el diseño de DESs. Dado que las redes de Petri ordinarias no se ocupan de sensores y actuadores, los conceptos de la Red de Petri se amplían mediante la inclusión de acciones y lecturas de los sensores, como estructuras formales dentro de la APN, para lo cual desarrolla un método de transformación llamado TPL o "*Token pass logic*", el cual permite obtener la implementación en el lenguaje estandarizado de escalera (LLD). Estas ampliaciones consisten en involucrar a las redes de Petri para acomodarlas en señales sensoriales en las transiciones y las acciones asignadas a los lugares. Un típico sistema discreto de eventos de control es visto en la Figura 6 (a). Este consiste en un (DES), acciones de control y un controlador discreto de eventos (DEC). La lecturas de sensor son consideradas como entradas desde los DEC a los DES. La función principal del DEC es supervisar las operaciones deseadas del DES y evitar las operaciones prohibidas. Al hacer esto, el DEC procesa las lecturas del sensor y luego los impulsos de la DES se ajustan a las especificaciones deseadas a través de las acciones de control. Las redes de petri pueden ser usadas para el diseño de tales DECs. Sin embargo, Las redes de petri

ordinarias no se ocupan de actuadores o sensores. Debido a esto es necesario definir un controlador base de las redes de Petri, el cual puede abarcar ambos actuadores y sensores dentro de la extendida estructura de red de petri. Un APN se observa en la Figura 6 (b). En el APN, las lecturas de los sensores pueden ser usadas como condiciones de disparos para la transición. La presencia o falta de las lecturas del sensor pueden ser usadas conjuntamente con las pre-condiciones de la extensa red de Petri para activar las transiciones. En el APN, dos tipos de acciones pueden ser consideradas, es decir, las acciones de impulso y las acciones de nivel. Las acciones son asociadas con los lugares. Con estas características adicionales, es posible diseñar un sistema de control de eventos discretos (DECS) [13]. La Figura 6 (c) muestra como un APN puede ser usado como un DEC en un DECs.

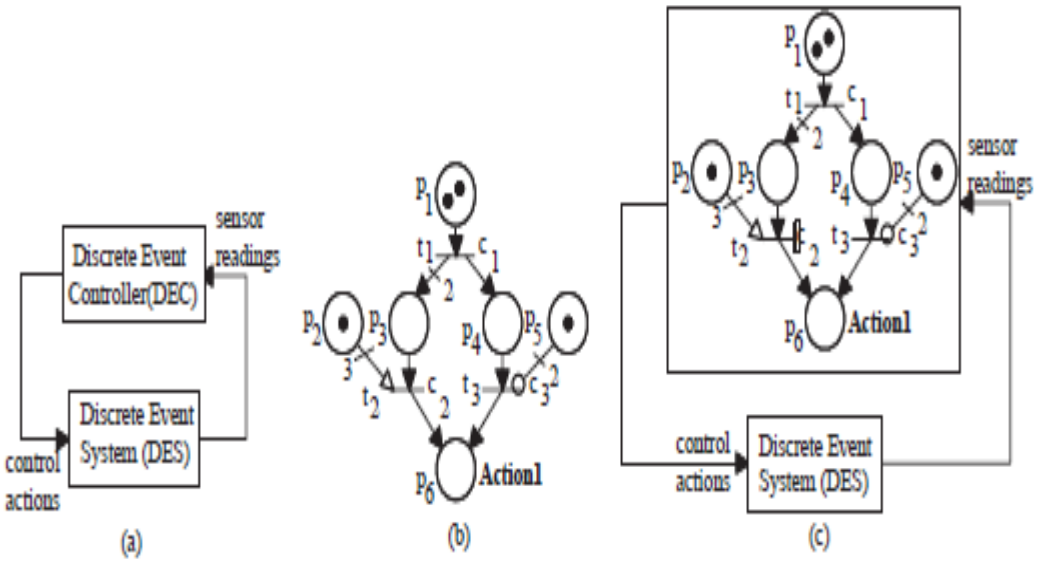

Figura 6. a) Sistema discreto de eventos. b) APN. c) APN usado como un DEC en un DECs. [32]

El movimiento de los muestras entre los lugares describe la evolución del APN y este se lleva a cabo por el disparo de la transición habilitada. Cabe señalar que el disparo de una transición permitida t no cambia la marca del lugar de entrada que son conectados a t solo por el tamaño de habilitación o los arcos inhibidores. En esto es también posible considerar el tiempo de APNs, como en las redes de Petri normales. A su vez, el autor termina realizando un recuento de los cinco lenguajes de programación estandarizados, grafico de funciones secuencial (SFC), diagrama escalera (LLD), diagrama de bloque de funciones (FBD). G. Frey en el 2000 [7], hizo una exposición de las propiedades de las IPN en 1998, pero expandió el modelo y desarrollo sus propiedades dinámicas y estáticas; además

de definir las reglas para la representación grafica del modelo. Este es un modelo sincronizado es una mejora respecto al previo y consiste en que las señales de salida son de varios tipos pudiendo activarse según los marcados de un lugar. Frey [8] explica las principales diferencias entre las IPN y el SFC definido. Por ejemplo, señala que en un SFC no existen estados transitorios, pero debido a los tiempos tan cortos del ciclo de operación de un PLC, las etapas de un SFC se comportan como etapas quasi-transitorias. En los modelos de SFC solo se permite un estado inicial y en un IPN, no. Dentro de un SFC, las etapas posteriores no son revisadas con las reglas de disparo. Además, señala que las transiciones son activadas por eventos, los cuales por definición no son simultáneos, mientras que las IPN son activadas por señales, lo que permite esta última posibilidad. Frey [14] propone en el mismo Congreso un método automático que utiliza las IPN para generar un software de diagrama escalera (LLD).

Frey vuelve a expandir el modelo en [8], para poder modelar sistemas híbridos que permitan trabajar con retardos de tiempos. El artículo explica en detalle un modelo genérico para diseñar algoritmos de control de procesos industriales, los cuales serán implementados en PLC. Adicionalmente, el artículo muestra los distintos estándares generados en el transcurso de los años para regular la programación de PLC. El proceso de diseño se debe dividir en tres actividades: La formalización que está dividida en tres actividades independientes y son realizadas según casos particulares. Las tareas que menciona son: formalización de propiedades específicas, modelado formal de procesos sin controlar modelado formal directo de algoritmos de control. La síntesis y la implementación utilizan como insumo la especificación formal obtenida en las actividades anteriores. La síntesis del algoritmo pretende obtener un diseño del algoritmo y la implementación de un algoritmo codificado de acuerdo con un lenguaje de programación de PLC estandarizado. Las actividades de verificación y validación (V&V) son actividades que se aplican a los resultados de la especificación formal y al resultado final. La V&V de especificaciones informales se divide en: basadas en el modelo, basadas en el no modelo y las basadas en las limitantes. La verificación de la especificación formal puede ser realizada de las siguientes seis formas: Redes de Petri, sistemas condición/evento, autómatas finitos e híbridos, lógica de alto nivel, lenguajes sincronizados, sistemas generales de transición y ecuaciones algebraicas.

Finalmente, la validación del resultado final se puede realizar utilizando: simulación, análisis de alcanzabilidad, chequeo del modelo y métodos de prueba de axiomas del modelo utilizando lógica matemática.

Por otra parte, P. Deussen en el 2001 [5], se dedico a verificar formalmente programas para PLC. Para esto creo una nueva subfamilia de Redes de Petri llamada RN o “Register Net”. Con este nuevo tipo de PN, los autores desarrollan un método para extraer del programa del PLC, una red que será utilizada para lo siguiente: verificar la ausencia/presencia de bloqueos, verificar la ausencia de errores de ejecución tales como rebases y divisiones entre cero, y verificar la seguridad del sistema programado. T. Mertke en el 2001, se apoyo en el trabajo de Frey para desarrollar una propuesta para la verificación de modelos de algoritmos de control para que sean utilizados por no especialistas. El define como “no especialistas” a ingenieros de control con nulo conocimiento en ciencias de la computación. Establece el proceso de verificación en seis etapas:

La primera, diseño de un controlador y generación de código escalera (LLD), la segunda, compilar el programa del PLC a un modelo de red IPN, usando un generador de Redes de Petri. La tercera, modelado del sistema del PLC, por medio de la validación, especificación informal y formal, formalización e implementación Resultado. La cuarta etapa es el modelado de la planta por PN. La quinta es combinar los tres modelos anteriores y generar un modelo general de todo el sistema en términos de PN. La sexta y última es especificar los requerimientos funcionales y de seguridad en la semántica de la lógica temporal, pues con estas reglas se evalúa el modelo general.

Inmediatamente se desarrolla un caso de estudio de una automatización a nivel industrial, donde verifican el programa para unos compresores de aire y unos tanques de almacenamiento. En el año 2001, E. Jiménez [10] presento en su tesis doctoral como utilizar las Redes de Petri para automatizar procesos industriales. A cada red le realiza su respectiva implementación en un lenguaje normado de programación de PLC. Adicionalmente, también realiza implementaciones en Matlab con el propósito de evaluar los rendimientos de los algoritmos. Finalmente, en este trabajo se realiza la automatización completa de una planta industrial.

En ese mismo año, K. John [21] publico un libro que explicaba en detalle la norma IEC 61131-3, la cual entraría a regir en el 2003. Dicha norma se ocupa de los cinco lenguajes estandarizados de programación y a su vez, es una actualización de la IEC 1131-3 de 1992.

G. Frey en el 2002 [7], estableció la necesidad de utilizar los estándares de desarrollo de software para asegurar la calidad del código generado para los PLC.

Con base en los criterios de calidad de software definidos en la ISO/IEC 9126 (funcionamiento, confiabilidad, usabilidad, eficiencia, portabilidad y mantenibilidad), el autor establece como desarrollar programas que cumplan con lo estipulado en el estándar.

Al año siguiente, Frey [13] vuelve a expandir las IPN para poder modelar sistemas híbridos mediante una 10-tupla llama hIPN. En ese mismo artículo, define y desarrolla la metodología para utilizar las IPN en diseños jerárquicos de programas para PLC. La filosofía que utiliza para el modelado es la llamada de abajo hacia arriba (*bottom-up*). Este nuevo tipo de Red de Petri es una tupla de orden once y recibe el nombre de IPNH. S.Peng y M. Zhou en el 2003 propusieron un nuevo método para modelar programas de PLC, al cual llaman SBSPN "*Sensor Based Stage Petri Nets*". Este nuevo tipo de Redes de Petri es capaz de simplificar modelos de complejos procesos. Para esto se define la Red y su representación grafica. Este nuevo tipo de PN permite:

- Que las entradas y salidas sean representadas por lugares y expresen el estado del programa de control del DECS (Controlador del Sistema de Eventos Dinámicos). Esto es diferente a los IPN, APN, etc.

- Los SBSPN permiten evaluar la lógica de control antes de la implementación y eliminar el inter-bloqueo entre elementos del programa de control.

- No existen recursos compartidos, debido a que se elimina la ocurrencia de las entradas/salidas al mismo instante.

- Para el modelado de DECS complejos, las SBSPN permiten el desarrollo e implementación de módulos.

En este trabajo se realiza un estudio de un caso práctico del modelado de una celda de manufactura flexible que utiliza un brazo robot. M. Bani y G. Frey [6] realizaron en el 2003 un estudio sobre trabajos de investigación dedicados a formalizar programas de PLC. La formalización la entiende como la definió Frey en [6,7]: proceso de transformar el código no formal existente, en uno formal, a lo cual se le conoce también como reinterpretación. Ellos clasifican la formalización de acuerdo con tres criterios:

- Formalización de módulos o partes del sistema de control.

- Formalización del programa completo (para un PLC).

- Formalización de toda la configuración de control (para varios PLC en modalidad jerárquica).

Los trabajos analizados en este estudio utilizan los modelos de autómatas finitos o Redes de Petri. En el 2003, Klein et al. desarrollaron un editor grafico donde se "dibuja" una SIPN y se genera automáticamente el código para PLC según lo estipulado en IEC 61131-3. En el artículo se presenta el editor desarrollado con la herramienta DiaGen. Luego se muestra como el editor se utiliza para verificar y validar diseños. Finalmente, se muestra el código generado por el software. En el 2004, E. Ávila et al. [1] publicaron un estudio de verificación de programas de PLC utilizando las propiedades estáticas o topología de la red. Las principales propiedades que se comprueban en el trabajo son la vivacidad y la seguridad. Para esto, se valen de un método matemático que utiliza el teorema de rango y el método grafico que utiliza las llamadas reglas de reducción para transformar una red compleja en otras simples. El uso de estas técnicas esta descrito en Murata [22].

El aporte de esta investigación radica en demostrar la utilidad práctica en problemas de automatización reales. Enmarcado en la aproximación teórica de algoritmos de control definida en [18], M. Uzam, 2004 [14], propone la síntesis de un controlador de eventos discretos a partir de Redes de Petri y la teoría de regiones. Para esto retoma el ejemplo desarrollado por Holloway en [9], que consistía en la automatización de una celda de manufactura flexible del tipo AGV. El método requiere que se defina previamente el modelo del sistema mediante una Red de Petri y se especifique el conjunto de estados prohibidos del modelo. El autor demuestra la aplicabilidad del método con un ejemplo de la teoría de control supervisor de autómatas finitos. Los ocho pasos para sintetizar el controlador son los siguientes:

- Diseñe el modelo del sistema de eventos discretos con una Red de Petri sin controlar.

- Defina los estados prohibidos de la Red.

- Defina las subredes (conjunto de lugares y transiciones) de la PN asociadas con las marcas prohibidas.

- Genere el árbol de alcanzabilidad de las subredes.

- Del árbol de alcanzabilidad, identifique los marcados buenos y los prohibidos.

- Especifique las transiciones controlables que en la evolución de la PN hacen que las marcas vayan de un marcado permitido a un marcado prohibido.

- Considere esas malas transiciones como eventos por separar del marcado bueno del árbol de alcanzabilidad. Use la teoría de regiones para generar el controlador.

- Añada el controlador generado al modelo para obtener el Sistema Controlado por Petri Nets (CPNM).

G. B. Lee, 2004 [15], propone un método de diseño para generar automáticamente diagramas escalera a partir de las Redes de Petri controladas (CPN), utilizadas para el control de sistemas de eventos discretos. Antes de desarrollar su método, se realiza un repaso de la CPN, así como su definición formal. El método consiste en convertir la red tipo CPN en un sistema de ecuaciones booleanas. Posteriormente, cada ecuación generada es trasladada a una representación en diagramas de escalera, la cual es fácilmente implementada en lenguaje de programación LLD para PLC. Si bien en la bibliografía del articulo no aparece referencia a artículos de Pessen [23], existe una similitud en la implementación final de los circuitos que automatizan procesos productivos en ambos trabajos. La deshabilitación del lugar actual es hecha por la transición posterior. Por otro lado, en el método cascada de Pessen, la deshabilitación de la bobina actual la realiza la bobina siguiente, lo cual podría traducirse como el lugar actual es deshabilitado por el lugar posterior, por lo que las soluciones en el diagrama LLD son similares. Finalmente, dicho artículo desarrolla un ejemplo práctico donde prueba el método. En este caso, la automatización de un sistema de mezclado de líquidos en una línea de producción. J. S. Lee, 2004 [24], realizo un estudio comparativo entre dos métodos de diseño de automatización, la primera obteniendo un diagrama escalera con el método cascada y la otra, diseñando con redes APN. Para esto, utiliza cinco procesos secuenciales por automatizar, donde el grado de complejidad es creciente. El autor realiza la automatización de las secuencias en diagrama escalera y en Red de Petri. Cada solución obtenida la transforma a estructuras IF-THEN y realiza un conteo de las transformaciones realizadas, así como de la cantidad de operaciones lógicas requeridas. Le demuestra que conforme aumenta la complejidad del proceso, las Redes de Petri utilizan una menor cantidad de operaciones lógicas y transformaciones IF-THEN. La siguiente figura muestra el consumo de estructuras lógicas IF-THEN para los cinco procesos. Note que los algoritmos desarrollados por PN muestran menor número de transformaciones.

S. S Peng publico en el 2004 [33] un resumen de los distintos tipos de redes de Petri para el diseño de controladores de eventos discretos. Hizo un recuento de los tipos más comunes, entre ellos, las redes de petri ordinarias, las redes de Petri temporizadas, las redes de Petri coloreadas y las redes de Petri controladas. Para las redes de Petri controladas, brinda como transformar las estructuras básicas de la red en un diagrama escalera. Se menciona que existen muchos subtipos de redes de Petri controlados y además se recalca que se debe realizar más estudios comparativos al respecto.

J.S. Lee, 2005 [16], propuso una nueva metodología para convertir las Redes de Petri controladas en diagramas escalera. La transformación o síntesis de las SPNC a diagramas escalera es realizada con el método TPL o *Token Pass Logic*, el cual es definido por Uzam en [13].

M.R. Lucas y D. M. Tilbury, 2005, propusieron un método para medir la complejidad y el tamaño de los programas para PLC que han sido diseñados bajo distintas metodologías. Los autores proponen seis índices para comparar algoritmos de control desarrollados con los métodos IPN, PN y maquinas modulares de estado finito. Los autores concluyen que las métricas presentadas son una base para futuros trabajos y los resultados que arrojan no son indicativos de cual técnica de modelado es la más apropiada para cada caso particular.

R. David, 2005, presenta una monografía de las Redes de Petri, sus clasificaciones, sus usos, etc. En cuanto a las Redes utilizadas para modelar programas para controladores industriales, presenta una exposición detallada de las redes de Petri Controladas, sus aplicaciones y ejercicios para resolver. G. Cansever, 2006, propone una nueva aproximación para el diseño de controladores de automatismos secuenciales utilizando Redes de Petri. El método propuesto es probado con el mismo problema que fue resuelto ocho años antes por Uzam con las APN.

1.3 CONVERSION DE CONTROLADORES DE REDES DE PETRI EN DIAGRAMAS DE LOGICA DE LADDER.

1.3.1. METODOLOGIA DE CONVERSION TPL

a. Controlador de Red de Petri (PNC)

Las redes ordinarias de Petri no se ocupan de actuadores o sensores. Debido a esto, es necesario definir un controlador de red de Petri (PNC), que puede abarcar tanto los actuadores y sensores en el actual marco de las redes de Petri. En un controlador de red de Petri los sensores son usados como transiciones. La presencia o ausencia de la lectura de los sensores puede ser usada junto con los requisitos normales de las redes de Petri para permitir o encender transiciones. En un controlador de red de Petri dos tipos de actuadores son considerados, los cuales son las acciones de impulso y las acciones de nivel. Los actuadores pueden ser asociados a cualquiera de los lugares o transiciones. Con estas características adicionales, es posible construir sistemas de control de eventos discretos. En este orden para facilitar la conversión de un controlador de red de Petri en un diagrama de lógica de escalera, se desarrollo la metodología TPL. [31]

La primera característica de la técnica de TPL es que facilita la conversión directa de los controladores de red de Petri a la lógica de control, esto se logra mediante la adopción del concepto de red de Petri y de la utilización de símbolos como el principal mecanismo para controlar el flujo de la lógica de control. Por lo tanto, cada lugar dentro de la red de Petri corresponde a un lugar dentro del programa del controlador de red de Petri. El movimiento simulado de marcas se logra mediante la implementación de contadores independientes en cada lugar de la lógica, cuya capacidad sea igual o superior a 1, estos contadores se incrementan y se disminuye para simular el paso de señal. Por lo tanto, cada lugar lógico dentro del programa del controlador de la red de Petri tiene un contador asociado y el valor de cuenta actual del contador en el lugar de la lógica representa el número de marcas que estaría en el lugar correspondiente dentro de la red de Petri. La asignación de un contador a un lugar de red de Petri se muestra en la Figura 7 (a), donde C representa el contador. Por último para completar la sinergia de la red de Petri, si el contador asociado con un lugar y la transición en la red de Petri asociada a ese lugar se vuelve activa, entonces el contador en el lugar se disminuye en uno, y los lugares posteriores vinculados por la transición se incrementa en uno. En el caso de lugares de capacidad única los contadores pueden ser reemplazados por banderas. El flujo de marcas se logra entonces mediante la creación y la reactivación de banderas. La asignación de una bandera

a un lugar de la red de Petri, es también mostrada en la Figura 7 (b), en donde F representa la bandera. Un temporizador de retardo puede ser fácilmente utilizado para modelar transiciones y lugares de tiempo en el método de controladores de redes de Petri. [31]

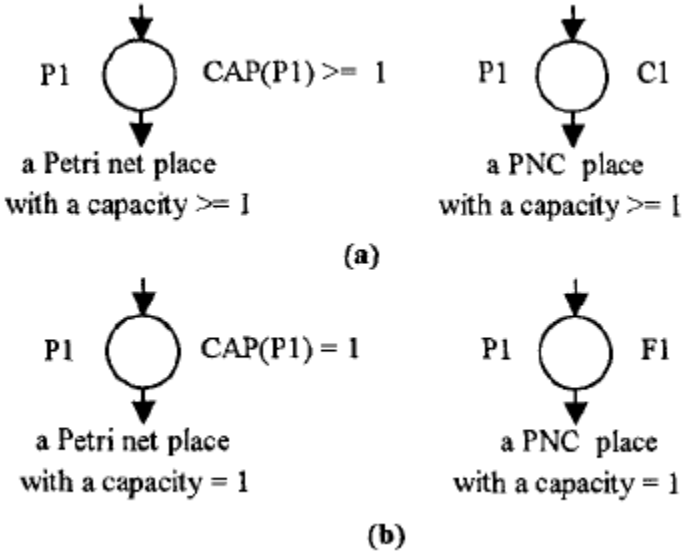

Figura 7. a) Lugares en las redes de Petri. b) Equivalente en los controladores de redes de Petri de las redes de Petri de Figura 7 (a). [31]

En esencia, los lugares en los controladores de redes de Petri son representadas por lugares lógicos, las muestras en las redes de Petri son representados por contadores independientes en cada lugar de la lógica. Por otra parte, el flujo de muestras en las redes de Petri se simula por contadores de cuenta ascendente y descendente o de manera similar mediante la creación y reactivación de las banderas adecuadas en los lugares apropiados. En los controladores de las redes de Petri, las acciones pueden ser asignadas a lugares o transiciones. Los lugares en los que se asignan las acciones se denominan lugares de control y las transiciones en las que se asignan las acciones se denominan transiciones de control. Si una acción es asignada a un lugar por un tiempo finito, esta corresponde a una característica de la red de Petri de P-timeed, sin embargo, si una acción es asignada a una transición de un tiempo finito, esto corresponde a una característica de red de Petri de T-timeed. [31]

En teoría, la metodología puede hacer frente a cualquier número de pasos y proporcionar una descripción visual del programa de control que tiene todas las ventajas de un completo análisis de red de Petri. Además, los controladores de red de Petri de color también se puede convertir en la lógica de control utilizando esta metodología, simplemente agregando más contadores a las banderas para cada

lugar. Se cree que esta nueva técnica proporciona una herramienta que es una simple, pero sofisticada manera de desarrollar complejos sistemas de control de eventos discretos. Son estas funciones con la que será vital para el éxito del sistema de fabricación ágil. [31]

Los controladores de redes de Petri son ilustrados considerando las siguientes estructuras de las redes de Petri: Las lecturas de los sensores en una transición, nivel de acción en un lugar, impulso de la acción en un lugar, nivel de acción de una transición e impulso de acción en una transición. [31]

b. Lecturas de Sensores en una Transición

Una transición estándar con lecturas de los sensores se muestra en la Figura 9. En la teoría de la red de Petri, una transición puede solo ser encendida si el número de marcas en el lugar de la entrada no es cero y se produce una señal que permite la transición. En un controlador de red de Petri un entorno propicio se puede utilizar como una condición adicional para la red de Petri de tal manera que si la lógica de una lectura del sensor es alta, entonces la condición se cumple. Del mismo modo, un arco inhibidor se puede utilizar como una condición adicional para la red de Petri de tal manera que si la lógica de una lectura del sensor es baja el requisito se cumple. La transición se activa cuando todas las condiciones se cumplen, y una o más condiciones previas sufren un cambio en el estado de la lógica. Esto se muestra en los casos a, b, c, d en la Figura 8. La habilitación de los sensores puede encender la transición cuando el estado de los sensores va de bajo a alto (flaco de subida ↑), {casos A y B}. La deshabilitación de los sensores puede disparar las transiciones cuando el estado del sensor va de alto a bajo (flanco de bajada ↓), {caso C}. Si las condiciones previas de las lecturas del sensor ya están satisfechas y una muestra entra en un lugar de entrada que antes no tenía muestra, la presencia de esta muestra también disparara la transición (flaco de subida ↑), {caso D}. En el método TPL, cuando una transición se enciende, se retira la marca del lugar lógico actual y se agrega una marca en el lugar lógico siguiente. [31]

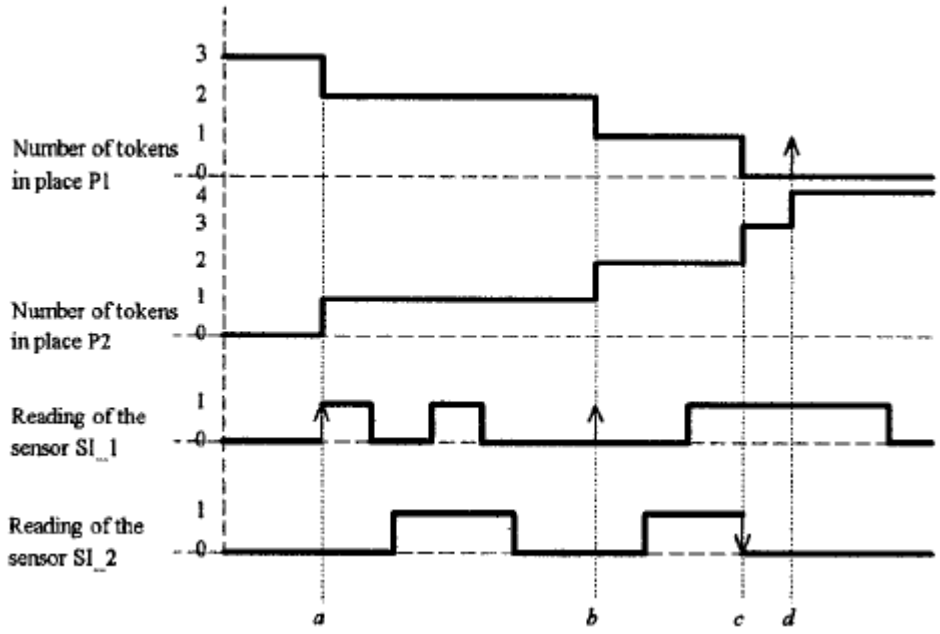

Figura 8. Encendido del controlador de red de Petri mostrado en la Figura 2 [31]

Esto se logra mediante el uso de un contador en cada lugar para representar a las muestras. Cuando una transición es encendida, para simular el paso de una muestra, la entrada del contador es decrementada y la salida del contador es incrementada en 1. El programa de controlador de red de Petri para la transición estándar con la lectura de los sensores se muestra en la Figura 9. y se da su forma en lógica de escalera en la Figura 10. [31]

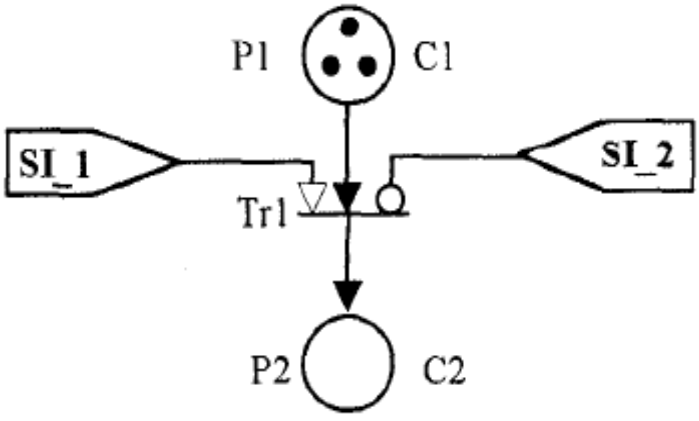

Figura 9. Lectura de sensores en una transicion en un controlador de red de Petri. [31]

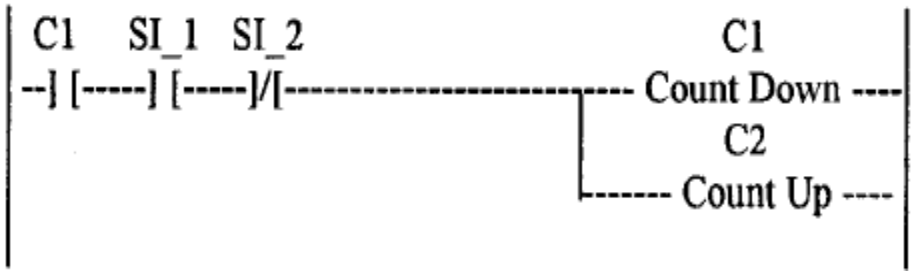

Figura 10. Diagrama de Logica de escalera para el control de red de petri de la Figura 9. [31]

c. Nivel de Acción en un Lugar

En un controlador de red de Petri, las acciones pueden ser asignadas a lugares llamados lugares de control. Si hay una condición en la transición de salida, entonces la acción se denomina nivel de acción, porque cuando una marca se deposita en un lugar de control, las acciones están permitidas hasta el momento en que la marca se retira del lugar. La Figura 11(a) muestra el lugar P2 en la red de Petri en la que una acción es asignada al nivel. Un nivel de acción en un lugar determinado dentro de una red de Petri se produce sólo si el número de marcas en el lugar no es cero. La Figura 11(b) muestra un lugar P2 en un controlador de red de Petri en la que una acción se le asigna el nivel. En un controlador de red de Petri, un nivel de acción en un lugar es controlado por un contador o una bandera en el lugar. Si el valor de la cuenta del contador es mayor que cero o la bandera relacionada se enciende, entonces todas las acciones relacionadas con el lugar se habilitan. La habilitación de la red de Petri se muestra en la Figura 11 (a). Y se da en la Figura 12. También, el diagrama de lógica de escalera para la red de control de Petri que se muestra en la Figura 11 (b) está dada en la Figura 13. [31]

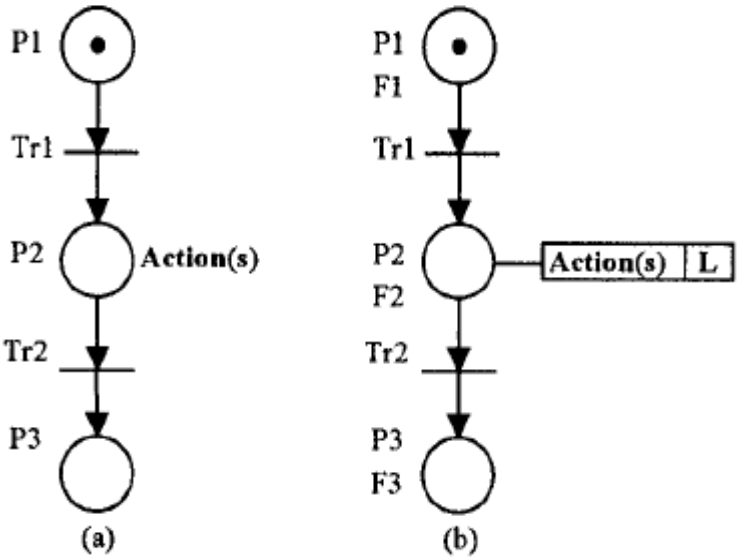

Figura 11. (a) Nivel de Acción asignado en un lugar de la Red de Petri.
(b) Nivel de acción en un control de lugar de un controlador de red de Petri. [31]

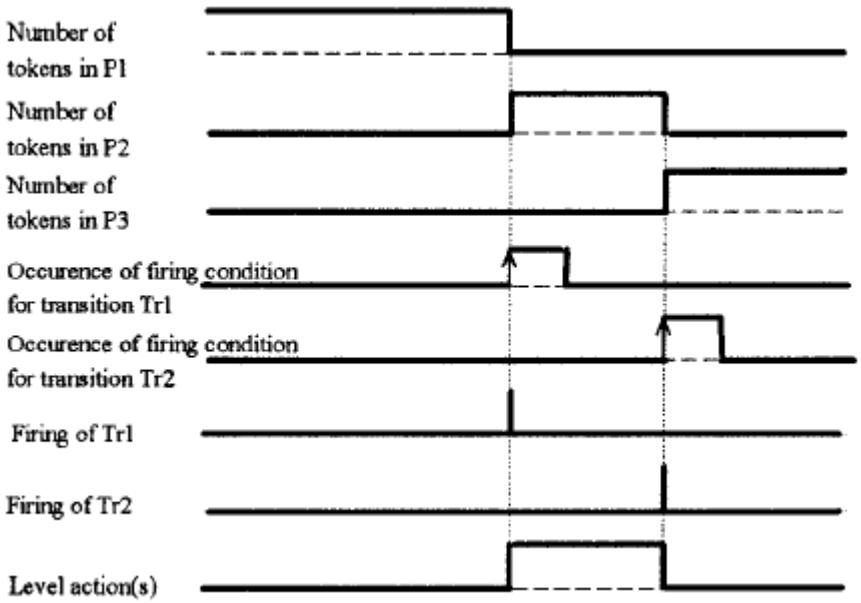

Figura 12. Accionamiento de la red de Petri mostrada en la Figura 5. (a) [31]

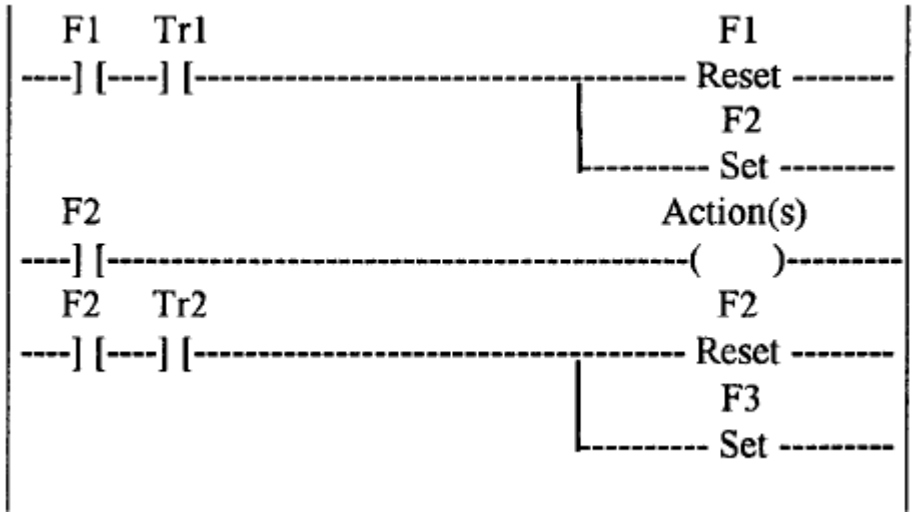

Figura 13. Diagrama de Lógica de Escalera para el controlador de red de Petri de la Figura 11 (b). [31]

d. Impulso de Acción en un Lugar

Si no hay ninguna condición sobre la transición de salida del lugar de control, entonces, una marca solo permanecerá en el lugar por un tiempo muy corto [↑]. La acción asignada a este lugar de control es entonces llamada impulso de acción. En este caso, cuando una marca se coloca en un lugar de control, las acciones son habilitadas e inmediatamente se deshabilitan las marcas y estas son removidas del lugar, por lo tanto, creando así un impulso de acción. La Figura 14 (a) muestra el lugar P2 en la red de Petri en el que se le asigna un impulso de acción. Un impulso de acción en un lugar determinado dentro de una red de Petri se produce sólo si el número de marcas en el lugar no es cero. En la Figura 14 (b) se muestra un lugar P2 en un controlador de red de Petri en el que se le asigna un impulso de acción. En un controlador de Red de Petri, un impulso de acción es controlado por un contador o una bandera en el lugar. Si el valor de la cuenta del contador en el lugar de control es mayor que cero o la bandera relacionada está habilitada entonces todas las acciones asociadas con el lugar se habilitan. El diagrama de la red de Petri se muestra en la Figura 14 (a) y su accionamiento se muestra en la Figura 15. También, el controlador de red de Petri se da desde la Figura 14 (b) y el correspondiente diagrama de lógica de escalera se muestra en la Figura 16. [31]

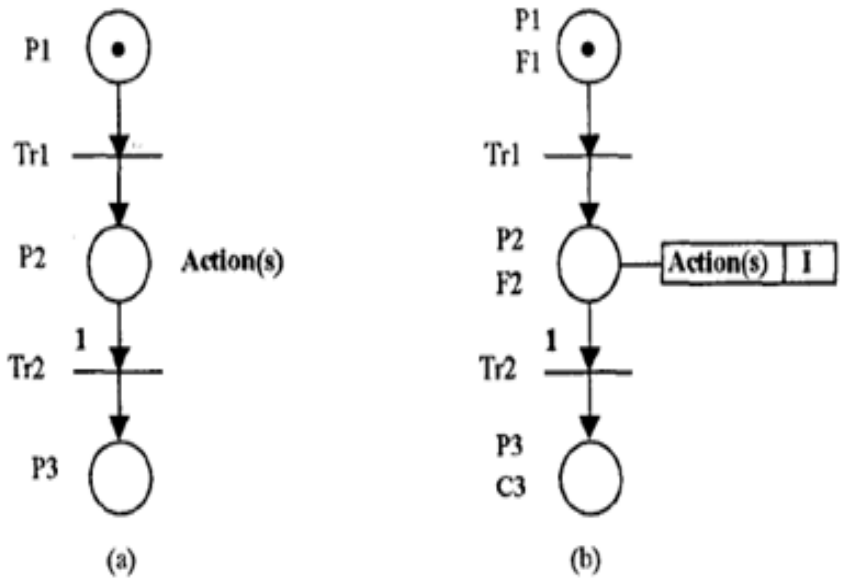

Figura 14. (a) Accion de impulso asignada a un lugar en una red de Petri. (b) Accion de Impulso asignada a un lugar de Control en una red de Petri. [31]

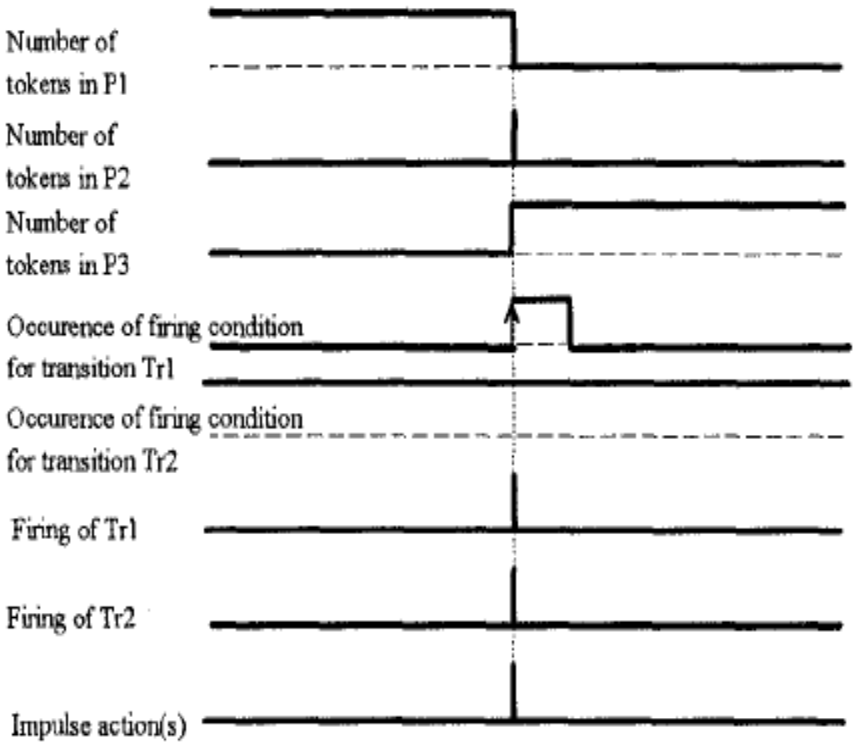

Figura 15. Encendido de la red de Petri de la Figura 14 (a). [31]

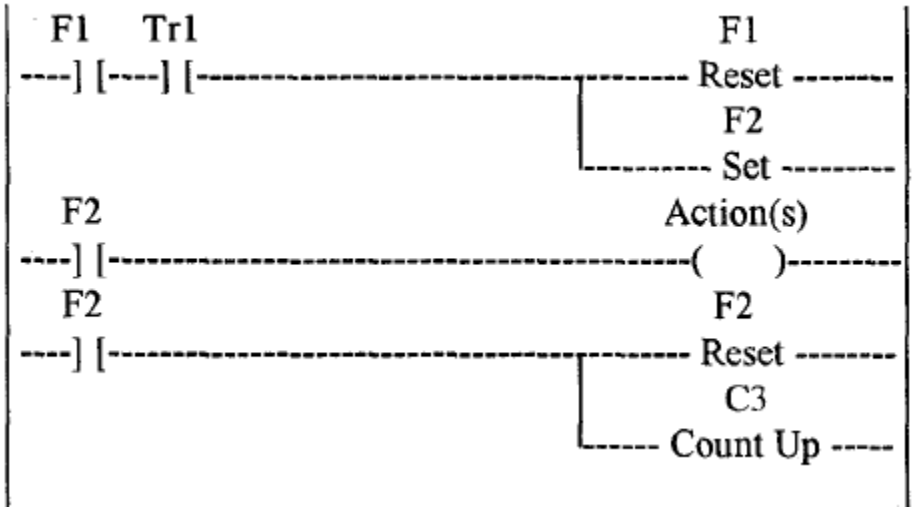

Figura 16. Diagrama de Logica de Escalera para el controlador de red de Petri mostrado en la Figura 14 (b) [31]

e. Nivel de Acción de una Transición.

Al considerar un nivel de acción, es importante tener en cuenta que dos son las señales requeridas para activar este nivel de acción, una señal para habilitar la acción y una para deshabilitar la acción. Este requisito de dos señales se consigue fácilmente en un nivel de acción en un lugar como se muestra en la Figura 11 (b). Sin embargo, en el caso de un nivel de accion de una transición no puede ser lograda, a menos que una transición jerárquica se despliegue. [31]

La transición jerárquica Tr1 se muestra en la Figura 17(a) esta es una transición jerárquica. Hay dos condiciones para la transición Tr1. El primero es un evento de habilitación (Tr1e) el cual es utilizado para habilitar la acción o acciones. La segunda condición para la transición Tr1 es un evento deshabilitante (Tr1d) que se utiliza para deshabilitar la acción o acciones. Cuando la condición de disparo Tr1e ocurre, el número de marcas en el lugar de la entrada P1 es decrementado en 1 y el número de marcas dentro de la transición jerárquica Tr1 es incrementado en 1. Una acción de nivel en una transición jerárquica dentro de una red de Petri ocurre solo si el número de marcas en el lugar en la transición jerárquica no es cero. Cuando la condición de disparo TR1d ocurre, el número de marcas en el lugar de transición es decrementado en 1 y el número de marcas en el lugar de salida P2 es incrementado en 1. Para lograr estos efectos en un controlador de red de Petri, un contador tiene que ser asignado a la transición. Cuando Tr12e ocurre, la bandera F1 asociada con el lugar P1 es restablecida y el contador C12 es incrementado en 1. Por lo tanto, cuando el valor de la cuenta en C12 no es cero,

esto permite las acciones relacionadas. Cuando TR12d ocurre, el contador C12 es decrementado y el contador C2 asociado con el lugar P2 es incrementado en 1. Es evidente que esta transición jerarquica es necesaria ya que no existe tal característica en las redes de Petri comunes. Durante el tiempo entre las condiciones que permiten el evento TR12e y el deshabilitar el evento TR12d, las acciones serán permitidas. El encendido de la red de Petri se muestra en la Figura 17 (a) y se da en la Figura 18, también el diagrama en lógica de escalera para el controlador de red de Petri de la Figura 17(b) se muestra en Figura 19. [31]

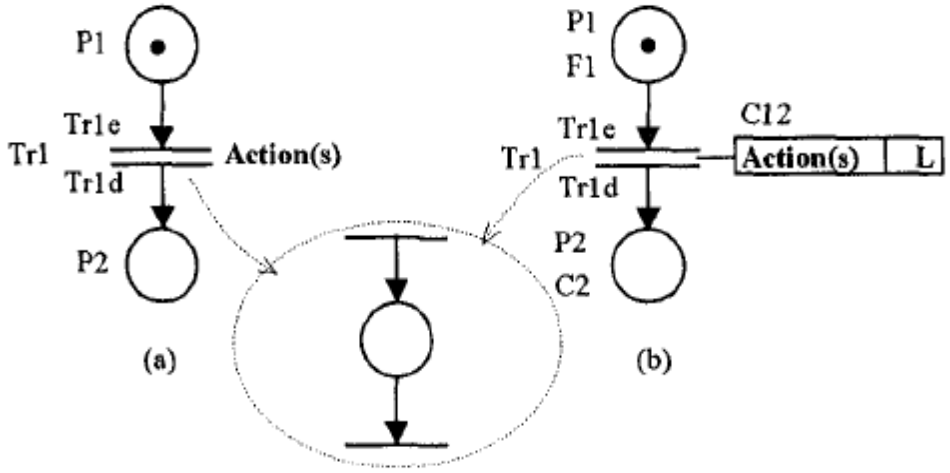

Figura 17. (a) Nivel de accion asignado a una transicion en una red de Petri. (b) Nivel de accion asignadao a una transicion de control en un controlador de red de Petri. [31]

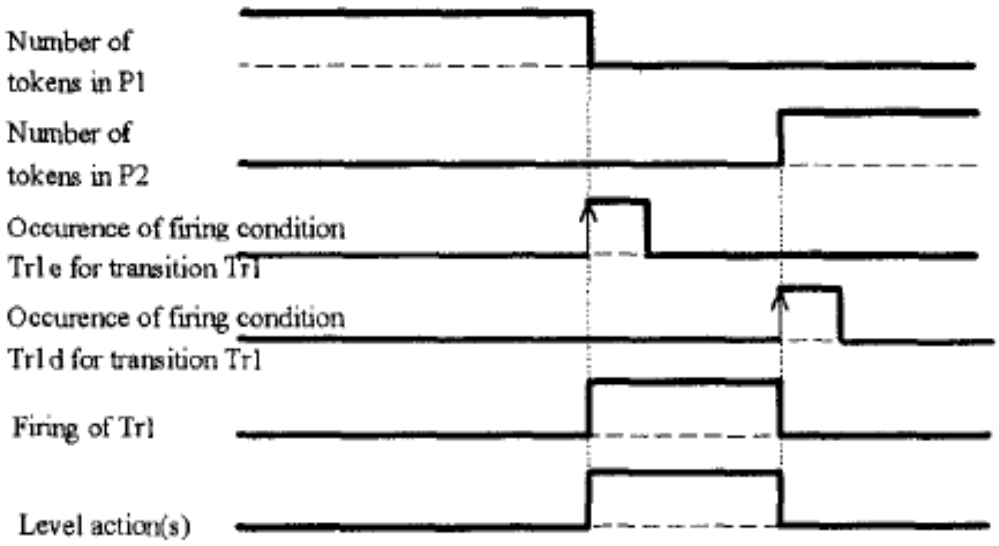

Figura 18. Accionamiento de la red de Petri mostrada en la Figura 17 (a) [31]

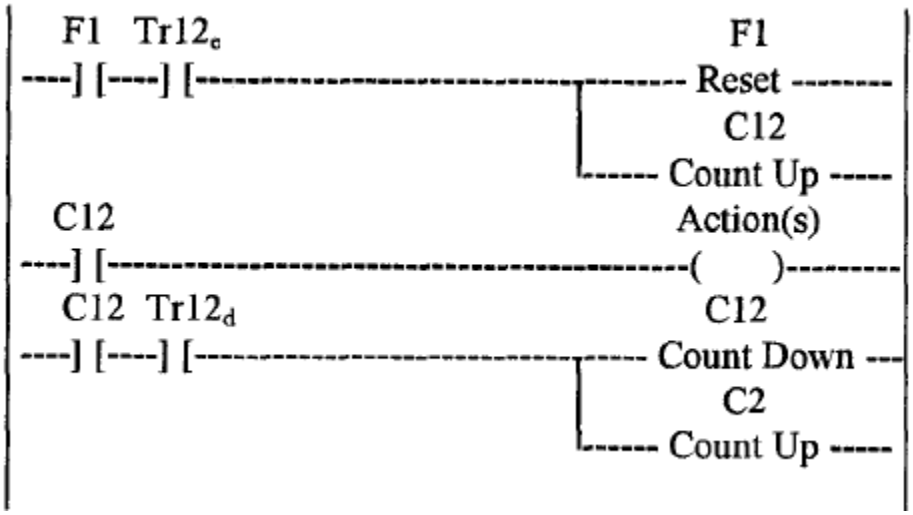

Figura 19. Diagrama de Logica de Escalera para el controlador de red de Petri de la Figura 17 (b) [31]

f. Impulso de Acción en una Transición

En un controlador de red de Petri, los impulsos de acción pueden ser asignadas a las transiciones. En este caso las acciones son impulsos de acción, porque la acción asociada con la transición se lleva a cabo como la transición de disparo. La Figura 20 (a) muestra la transición Tr1 en la red de Petri, en la cual una acción de impulso le es asignada. Si hay una marca en el lugar P1 y se produce una condición para la transición Tr1 ocurre entonces que una marca es removida del lugar P1 y pasa al lugar P2. La acción de impulso en la transición Tr1 ocurre solo cuando la transición se dispara. La Figura 20 (b) muestra la transición Tr1 en la que una acción se le asigna para un controlador de red de Petri. En el controlador de red de Petri, las acciones de impulso se llevan a cabo cuando la bandera de entrada es reseteada y la bandera de salida se establece. En el caso de contadores, esto corresponde a conteos descendentes y conteos ascendentes de estos mismos. El accionamiento de la red de Petri se muestra en la Figura 20 (a) y se grafica en la Figura 21. También el diagrama en lógica de escalera para el controlador de red de Petri de la Figura 20 (b) y se da en la Figura 22. [31]

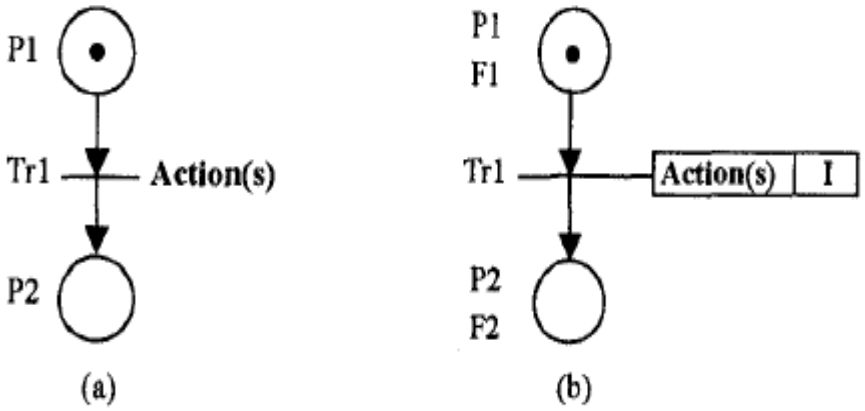

Figura 20. (a) Impulso de Accion asignado a una transicion en una red de Petri. (b) Impulso de Accion asignado a una transicion de control en un controlador de red de Petri. [31]

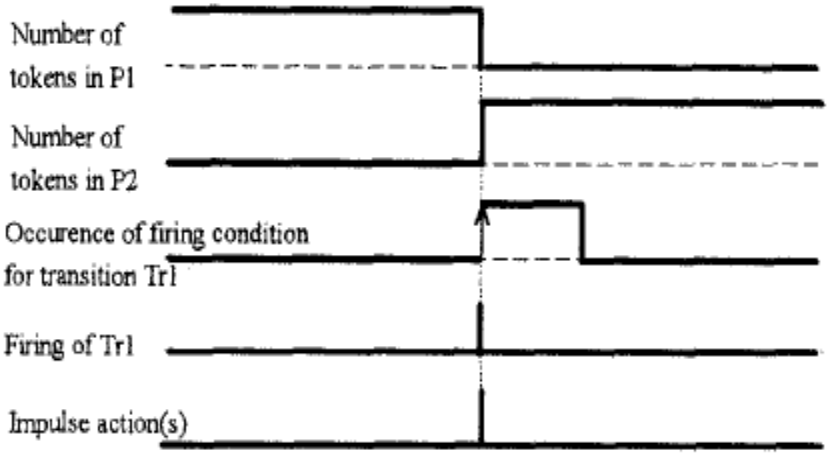

Figura 21. Accionamiento de la red de Petri de la Figura 20 (a) [31]

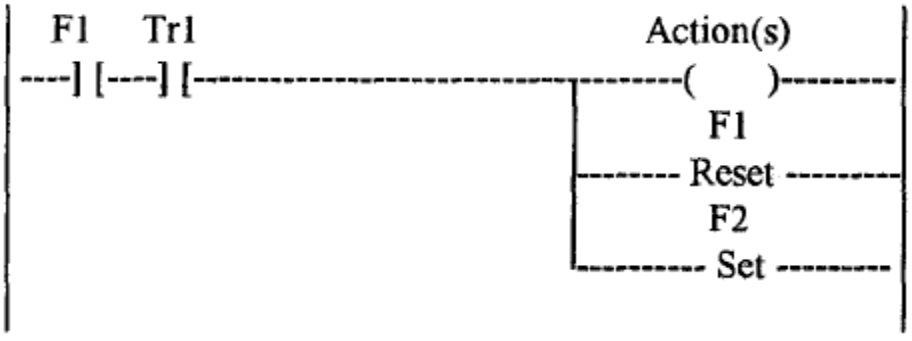

Figura 22. Diagrama de Lógica de Escalera para el controlador de red de Petri de la Figura 20 (b) [31]

1.3.2. EJEMPLO DE FABRICACION DE SISTEMAS

La fabricación de sistemas mostrada en la Figura 23, representa un componente de los procesos de selección que puede ser controlado por prácticamente cualquier PLC. En este ejemplo, una banda transportadora es manejada por el motor A1 (actuador 1), acá una selección aleatoria de las parte tipo A y de las partes tipo B son colocadas en la transportadora, denominando estas como la parte As y la parte Bs, las cuales necesitan ser identificadas y separadas. Esto se hace mediante dos sensores, un sensor de proximidad S1 (Sensor 1) y un sensor reflectivo infrarrojo S2 (sensor 2). Mediante el uso de esto dos sensores se pueden distinguir entre las partes As y Bs. Por medio del solenoide A2 (actuador 2), la parte As puede ser expulsada hacia el almacenamiento I. Y la parte Bs, mientras tanto, continúa en la transportadora y es desviada hacia el almacenamiento II. Un emisor/detector infrarrojo S3 (sensor 3) es usado para determinar si existe o no un componente enfrente del solenoide tipo A2. Si el sensor S3 es activado, el solenoide tipo A2 puede ser usado para expulsar ya sea una parte A o una parte B en almacenamiento I. Si no se acciona esta muestra, el componente se lleva hacia el almacenamiento II por la transportadora. En la tabla 1 se explican las lecturas de los sensores. [31]

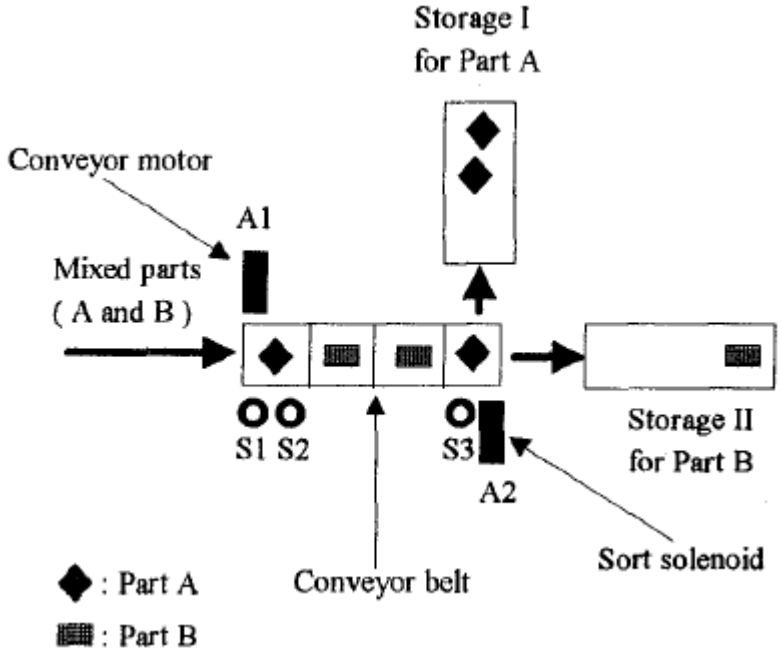

Figura 23. Fabricacion del sistema de una banda transportadora [31]

LECTURA DE LOS SENSORES	INTERPRETACIONES
S1&(S2 negado)	Parte A
S1 & S2	Parte B
S3	Parte A o Parte B

Tabla 2. Lógica de la lectura de los sensores [31]

1.3.3. CONTROLADOR DE RED DE PETRI PARA FABRICACION DE SISTEMAS

Un controlador de red de Petri (PNC) se muestra en la Figura 24, este diseño del controlador se hace para el modelamiento del sistema de la banda transportadora del ítem anterior y utiliza el modelo de red de Petri de fabricación de sistemas.

Acá una base de estructura se implementa para los dos tipos de parte As y Bs, utilizando los lugares P1 a P8, donde los lugares P1, P3, P5 y P7 son usados para construir una base para partes As y los lugares P2, P4, P6 y P8 son usados para construir bases para las partes Bs. Se tiene en cuenta que los arcos inhibidores en las transiciones Tr3 a Tr8, construyen una base para ambos tipos de partes. En el controlador las partes As necesitan ser puestas en el almacenamiento I y las partes Bs necesitan ser puestas en el almacenamiento II. Esto se logra, mediante el uso de los lugares P9 y P10. Una vez hecho esto las lecturas del sensor se añaden al Controlador de Red de Petri. La distinción entre las partes As y Bs se logra mediante el uso del sensor Sl_1 y el sensor Sl_2 y la transición Tr1 y Tr2. La lectura del sensor Sl_3 es usada en Tr7 y Tr8, para detectar la presencia de un componente y la transición Tr9 y Tr10 para detectar la ausencia de un componente. Finalmente, la acción de control del motor de la transportadora es asignada al lugar P11 y la acción del solenoide tipo es asignada a la transición Tr9. Una vez, el controlador de red de Petri ha sido diseñado, entonces las banderas y los contadores son entonces asignados a los lugares como se muestra en la tabla 2. [31]

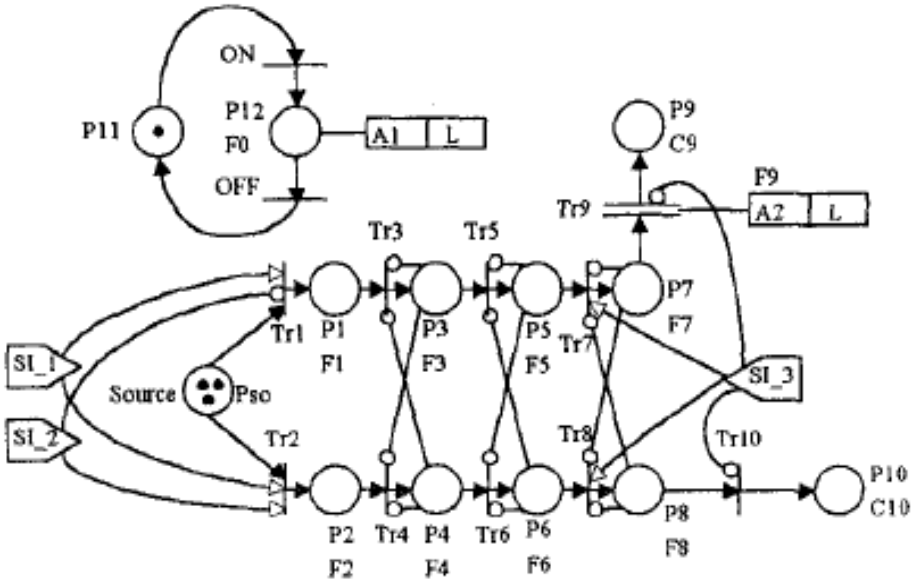

Figura 24. Controlador de red de Petri para sistema de banda transportadora [31]

LUGARES EN EL PNC	INTERPRETACIÓN
Ps0	Componentes de Inicio
P1	Primer lugar de la cinta transportadora para la parte A
P2	Primer lugar de la cinta transportadora para la parte B
P3	Segundo lugar de la cinta transportadora para la parte A
P4	Segundo lugar de la cinta transportadora para la parte B
P5	Tercer lugar de la cinta transportadora para la parte A
P6	Tercer lugar de la cinta transportadora para la parte B
P7	Cuarto lugar de la cinta transportadora para la parte A
P8	Cuarto lugar de la cinta transportadora para la parte B
P9	Lugar para las partes As en el almacenamiento I
P10	Lugar para las partes Bs en el almacenamiento II
P11	Motor de la Transportadora apagado
P12	Motor de la transportadora encendido

Tabla 3. Asignación de lugares para el controlador de red de Petri de la Figura 17.

1.3.4. DIAGRAMA DE LOGICA DE ESCALERA PARA UN CONTROLADOR DE RED DE PETRI

El diagrama de lógica de escalera, se muestra en la Figura 25, este se obtuvo para el controlador de red de Petri dado en la Figura 24, se logra mediante el uso directo del mapeo del controlador de red de Petri a la lógica de escalera, los símbolos de la lógica de escalera son definidos en la Tabla 4. El programa en lógica de escalera ha sido estructurado de tal manera que en el reglón cero inicia el sistema, del reglón 1 al 10 se representan las transiciones Tr1 a Tr10 respectivamente. Y finalmente el reglón 11 representa la acción de nivel A1 (operación del motor de la transportadora) y en el lugar P0 y reglón 12 representa la acción de nivel A2 (operación de solenoide tipo) y la transición Tr9. Para adoptar estos conceptos con mayor claridad, puede ser agregada documentación al sistema. [31]

SÍMBOLOS DEL DIAGRAMA DE LÓGICA DE ESCALERA	DEFINICIÓN
F	Bandera
C	Contador
SI	Entrada de Sensor
A	Acción (Salida)

Tabla 4. Definición de las variables en el diagrama de Lógica de escalera [31]

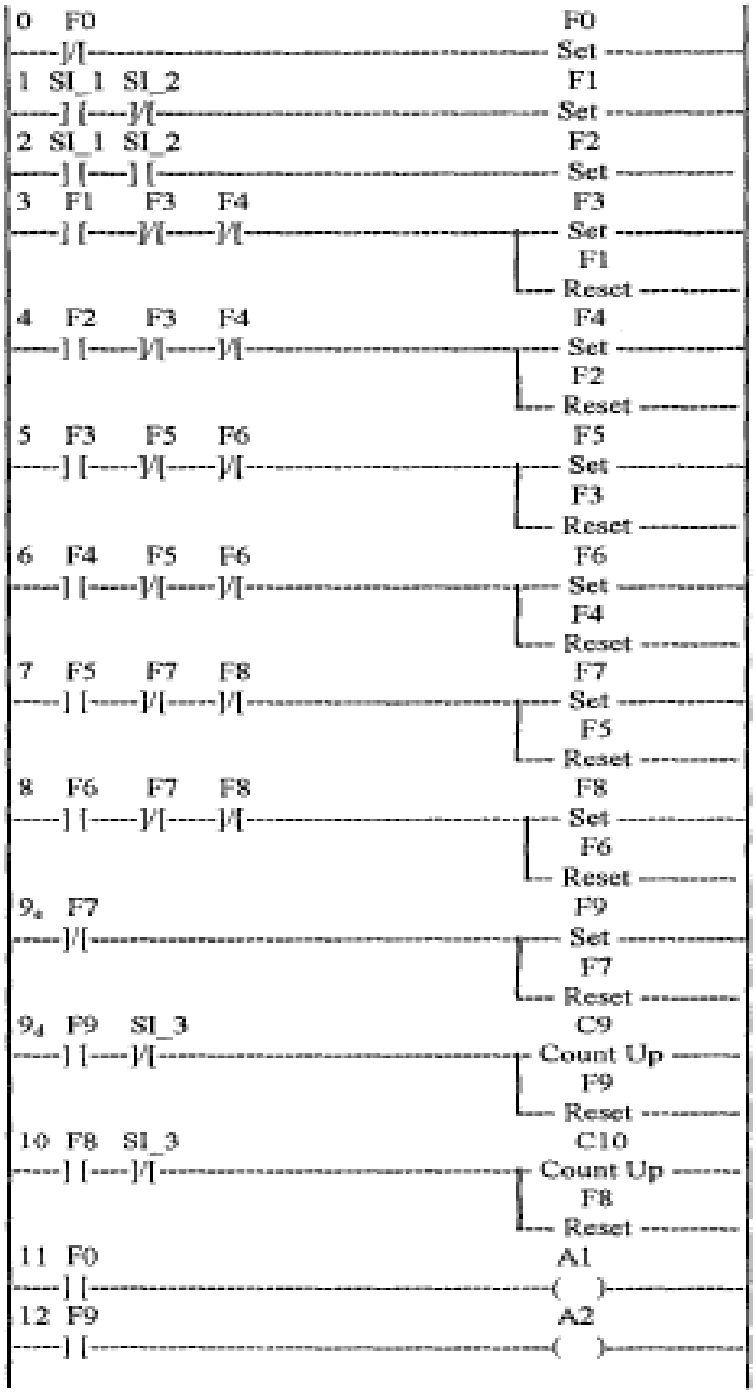

Figura 25. Diagrama de Lógica de escalera para el controlador de red de Petri de la Figura 24. [31]

RECOMENDACIONES

Para la selección adecuada del modelo a trabajar de la Red de Petri, se debe realizar un estudio exhaustivo de las diversas metodologías existentes para diseñar los algoritmos de control que se utilizan en la programación de los PLC's. Sin embargo, las técnicas que prevalecen son las técnicas gráficas, ya que las técnicas algebraicas de simplificación y verificación de algoritmos realizados con Redes de Petri son complejas y dispendiosas.

En nuestras universidades es necesario replantear los cursos referentes a la enseñanza del funcionamiento y programación de PLC's. En el plan de estudios se debe prestar mayor atención al levantamiento formal de los requerimientos y a su validación, al modelado formal de la planta y del controlador, a la validación de los modelos, a la implementación del software y a su verificación. Además, es necesario incorporar los criterios de calidad de software definidos en el ISO/ IEC 9126.

Es importante seguir en el desarrollo e investigación de las metodologías y modelos con Redes de Petri, ya que es un área prioritaria en el futuro, dada la complejidad de los procesos de automatización con los cuales la industria se está enfrentando en la actualidad y la practicidad en la técnica gráfica que presentan este tipo de redes para la implementación de estos sistemas.

CONCLUSIONES

Desde hace más de 48 años, las Redes de Petri han expuesto su utilidad práctica en muchas aéreas y en especial, en el modelado de sistemas complejos. A pesar de la gran cantidad de artículos existentes, su generalización en el ámbito colombiano es muy limitada. Este trabajo permite al lector una visión conceptual de las familias de las redes de Petri y una revisión cronológica de las investigaciones vinculadas con los PLC's. Estas investigaciones generan un conocimiento sobre el impacto de las mismas en el diseño e implementación de algoritmos de control para controladores lógicos programables.

Mediante la Metodología de Paso de testigo o lógica TPL y haciendo adecuaciones a las redes de Petri, podemos obtener la representación de este tipo de redes al lenguaje que mayormente es trabajado en la industria de la automatización el cual como se ha mencionado con anterioridad es el lenguaje en diagrama de escalera o Ladder.

Las redes de Petri se han establecido como la metodología de diseño más completa en lo que tiene que ver con robustez y aplicabilidad para casos de alta complejidad al afrontar problemas de diseño de control.

BIBLIOGRAFIA

[1] http://en.wikipedia.org/wiki/Petri_net ; http://es.wikipedia.org/wiki/Red_de_Petri

[2] http://en.wikipedia.org/wiki/Ladder_logic ;
http://es.wikipedia.org/wiki/Lenguaje_Ladder

[3] *http://computacion.cs.cinvestav.mx/~ameneses/pub/tesis/mtesis/node7.html*

[4] Y. Cai, I. Nishi, T, Sekiguchi*. "Modeling by Petri nets with invariants for sequencial control systems"*. *Electrical engineering in Japan*, Vol. 115, No 5. 1995.

[5] P. Deussen "Partial order verification of Programmable Logic Controllers". *J-M Colom and M.Koutny: ICATPN 2001*, LNCS 2075, pp 144-163, 2001.

[6] G. Frey. "Software Quality in Logic Controller Programming". *Proceeding of IEEE SMC 2002*, Tunisia, Oct. 2002.

[7] G. Frey. "Hierarchical design of logic controllers using Signal Interpreted Petri Nets". *Proceeding of IFAC AHDS 2003*, France, pp 401-406, June 2003.

[8] G. Frey. "PLC programming with Signal Interpreted Petri Nets". *Proceeding of ICATPN*, LNCS 2679, pp 401-406, Netherlands, June 2003.

[9] L. E. Holloway, B. H. Krogh. "Syntesis of feedback control logic for a class of controllers Petri nets". *IEEE Transactions on Automatic Control*. Vol. 35, No. 5, pp 514-523, May 1990.

[10] M. A. Jafari, T. 0. Boucher, "A Rule - Based System For Generating Ladder Logic Control Program From a High Level System Model", *Journal of Intelligent Manufacturing, **5,*** 1994, pp. 103 - 120.

[11] E. Jiménez. *Técnicas de automatización avanzada en procesos industriales*. PhD Tesis. Ed. Serv. ublicaciones de la Universidad de la Rioja. Logroño, 2001.

[12] K. John. *IEC 61131-3: Programming industrial automation systems. Concepts and programming anguajes.* Springer-Verlag, Berlin, Germany, 2001.

[13] A.H. Jones and M. UZAM. \ *A New Petri-Net-Based Synthesis Techique for Supervisory Control of Discrete Event Systems",* Trrk J Elec Engin, VOL. 10, N0.1 2002, TUBITAK

[14] A.H Jones, M. Uzam, A. H. Khan, D. Karimzadgan, S.B. Kenway, *A General Methodology for Converting Petri Nets Into Ladder Logic: The TPLL Methodology*", Proc. of CIMAT'96, pp. 357-362, Grenoble, France, May 29-31, 1996.

[15] J. S. Lee, P. L. Hsu. "A improved evaluation of ladder logic diagrams and Petri nets for sequence controller design in manufacturing systems"*. International Journal of Advanced Manufacturing Technology*. Vol. 24, No. 3-4, pp 279-287. August 2004.

[16] J. S. Lee, P. L. Hsu. "A systematic approach for the sequence controller design in manufacturing systems". *International Journal of Advanced Manufacturing Technology*. Vol. 25, No. 7-8, pp 754-760. April 2005.

[17] J. Martínez, M. Silva, M. Velilla. "Realización cableada de Redes de Petri binarias". *Questiió*, Vol. 6, No. 1, Març 1982.

[18] T. Mertke, G. Frey. "Formal Verification of PLC-Programs Generated from SIPN". *Proceedings of IEEE SMC 2001*, pp 2700- 2705.

[19] I. Moon. "Modeling programmable logic controllers for logic verification". *IEEE Control Systems Magazine,* pp 53-59. April 1994.

[20] G. Mušič, D. Matko. "Petri nets based supervisory control of flexible manufacturing plants" *in Prepr. 8th IFAC Symp. On large Scales Systems: Theory & Applications*, Vol. 2, Rio Patras, Greece, 1998.

[21] T. Murata. "Petri nets and Marked Graphs. Mathematical models pf concurrent computation*". The American Matematical Monthly*, Vol. 89, No. 8, pp 552-566, Oct 1982.

[22] T. Murata. "Petri nets: Properties, Analysis and application*". Proceedings of the IEEE*, Vol 77, No 4, pp 541-580, April 1989.

[23] D.W. Pessen. *Industrial Automation: Circuit Design & Components*. Ed. Wiley Interscience Publication, USA, 1989

[24] J.L. Peterson. "Petri nets". *ACM Computer Survey*, Vol. 9, No. 3, pp 223-252, September 1977.

[25] S. Petersson, B. Lennartson. "Hybrid Modeling Focused on Hybrid Petri Nets". *European Worshop on Hybrid Systems*, Grenoble (Fr), pp 303-309, 1995.

[26] P.J. Ramadge and W.M. Wonham, *The Control of Discrete Event Systems*", Proc. of IEEE, vol. 77, no. 5, pp. 81 - 98, 1989.

[27] P.J. Ramadge and W.M. Wonham, *Modular Feedback Logic for Discrete Event Systems*", SIAM Journal of Control and Optimization, vol. 25, no. 5, pp. 1202 - 1218, September 1987

[28] Castellanos Carlos. Consideraciones para el modelado de sistemas mediante Redes de Petri. Revista ciencia e Ingeniería. Vol. 27, N° 2, p.p. 49-58, abril- julio, 2006.

[29] L.W. Schruben. "Graphical model structures for discrete event simulation*". Proceeding of the 1992 winter simulation conference.* Ed J.J Swain.

[30] A. Taholakian, W. M. Hales. "PN⇔PLC: A methodology for designing, simulating and coding PLC based control system using Petri nets" *International Journal In Production Research*, Vol. 35, No. 6 pp 1743-1762, 1997.

[31] M. Uzam, A.H. Jones and N. Ajlouni \ *Conversion of Petri Net Controllers for Manufacturing Systems into Ladder Logic Diagrams*", Proc. Of IEEE, 1996.

[32] Murillo Soto, Luis Diego. Redes de Petri: Modelado e implementación de algoritmos para autómatas programables. Tecnología en Marcha**,** *Vol. 21, N.° 4, pp. 102-125, Octubre-Diciembre 2008.*

ANEXOS

ANEXO 1. Resumen del Estándar Internacional

IEC 61131-1.