Journal of Engineering and Technology for Industrial Applications, 2018. Edition. 13.Vol: 04
https://www.itegam-jetia.org
ISSN ONLINE: 2447-0228
DOI: https://dx.doi.org/10.5935/2447-0228.20180006

Formal design on GHENeSys IEC-1131 compatible of discrete event systems with PLCs

Elio R. Avila Rodríguez[1]

[1]ETECSA S.A - Ave. 1ro de Enero, Edif. 33, Rpto. Santos, Las Tunas, Cuba.

Email: elio.avila@etecsa.cu

Received: January 24th, 2018

Accepted: February 27th, 2018

Published: March 30th, 2018

ABSTRACT

Although when is demonstrated the effectiveness of the modelling of Petri nets as a formal method of design automation with PLCs, with a strong interest in both teaching professionals such as engineering, their use in industrial solutions is not widespread, mainly given fact by the complexity of application that sometimes involves, as well as the limitations that still exist in the correspondence model-code program PLC. This paper discusses the use of a formal design methodology, which uses the extended hierarchical network GHENeSys IEC-1131 Compatible, to particular case of automation with PLCs of Discrete Event Systems. This has among its purposes to contribute to the solution of the above limitations.

Keywords: Petri Nets, Programmable Logic Controllers, Discrete Event Systems, Verification and Validation.

Diseño formal sobre GHENeSys IEC-1131 compatible de sistemas de eventos dicretos con PLCs

RESUMEN

Aún cuando se ha demostrado la efectividad del modelado sobre redes de Petri como método formal de diseño de automatizaciones con PLCs, con marcado interés de profesionales tanto de la docencia como la ingeniería, su utilización en soluciones industriales no resulta ampliamente difundido, hecho dado principalmente por la complejidad que en ocasiones implica su aplicación, así como por las limitaciones que todavía existen en la correspondencia modelo-programa en código PLC. El presente trabajo trata sobre el empleo de una metodología de diseño formal, que utiliza la red jerárquica extendida GHENeSys IEC-1131 Compatible, al caso particular de automatizaciones con PLCs de Sistemas de Eventos Discretos. Esta tiene entre sus propósitos contribuir a la solución de las limitaciones antes mencionadas.

Palabras Clave: Redes de Petri, Controladores Lógicos Programables, Sistemas de Eventos Discretos, Verificación y Validación.

I. INTRODUCCIÓN

Los Controladores Lógicos Programables (PLCs) son computadoras diseñadas para trabajar en ambientes industriales, y para su operación cuentan con una parte de hardware y otra de software. La norma internacional IEC 61131-3 define la sintaxis, y en menor medida, la semántica de cinco lenguajes de programación, dos de ellos lenguajes de texto: Instruction List (IL) y Structured Text (ST), dos gráficos: Ladder Diagram (LD) y Function Block Diagram (FBD), y uno estructurado: Secuential Funtion Chart (SFC), el LD sigue siendo el favorito al ser utilizado por 96% de sus usuarios. La norma sólo describe los lenguajes, pero no las metodologías de diseño.

Aunque algunos analistas prevén el decrecimiento en el uso y el reemplazo de los PLCs por nuevos dispositivos aparecidos en el mercado, estudios recientes demuestran el poder de adaptabilidad conferido por sus fabricantes a las condiciones industriales contemporáneas, así como la popularidad mantenida por éstos [1-2], que los convierte en los dispositivos de automatización industrial de mayor uso en el mercado

internacional desde los años 1970 hasta la fecha [3-4]. La Tabla I muestra el comportamiento comparativo de sus áreas de aplicación, a partir de la información obtenida en dos estudios realizados en la última década [1-2].

Tabla 1: Aplicación de los PLCs por áreas.

Nro	Área de aplicación	Año 2001 (%)	Año 2007 (%)
1	Control de máquinas	87	84
2	Control de procesos	58	74
3	Control de movimiento	40	55
4	Control por lotes	26	31
5	Aplicaciones de diagnóstico	18	25
6	Aplicaciones de seguridad	-	1
7	Otras	3	5

Fuente: Autor, (2018).

Muchas de estas aplicaciones comprenden el manejo de información cuya naturaleza de ocurrencia es aleatoria y en intervalos discretos de tiempo, denominados *eventos*, los que modifican los estados por los que transita el sistema en su operación. Ejemplos de eventos resultan el cambio de información brindada por un sensor y el inicio o fin de una acción. Estos sistemas se denominan Sistemas de Eventos Discretos (DES) y debido a su naturaleza discreta, no es posible analizarlos a partir de modelos matemáticos basados en ecuaciones diferenciales, tratados en la Teoría de Control Clásica para sistemas de variables continuas.

Ante esta limitante, en la búsqueda de modelos matemáticos adecuados para representar los DES, se destaca la Teoría del Control Supervisorio desarrollada por Ramage y Wonham [5]. En ésta se define un DES como "un sistema dinámico que involucra la posible ocurrencia de eventos físicos en intervalos irregulares desconocidos". El control de DES asume que se puede impedir la ocurrencia de ciertos eventos del sistema cuando se desee, si se está prevenido de su ocurrencia. Para modelar este control, se separa el conjunto total de eventos **Σ**, en **eventos incontrolables y controlables (Σ = Σi + Σc)**. Los eventos **Σc** se pueden impedir en cualquier momento, mientras que los eventos **Σi** se modelan como aquellos que el agente controlador no puede influir sobre ellos.

Por otra parte, se separa el conjunto total de eventos en **observables**, que son aquellos que se puede conocer su ocurrencia a través de sensores o cálculos, y los **no observables**, para los que no es posible conocerlo, es decir, $\Sigma = \Sigma_o + \Sigma_n$. De aquí se infiere que un evento incontrolable pueda ser a su vez observable.

Sin dudas, las redes de Petri (PNs) se erigen como el método de modelado/representación mayormente utilizado en el diseño formal de automatizaciones con PLCs [3][4]; creadas por Carl Adam Petri en 1962, tienen como fortaleza su formalismo gráfico-matemático bien fundamentado y resultar útiles para describir y estudiar sistemas que se caracterizan por ser concurrentes, sincrónicos, distribuidos, paralelos, no determinísticos y/o estocásticos [4][6][7].

En este contexto, la síntesis de programas de PLCs basados en modelos desarrollados sobre PNs, resulta un caso particular dentro de la síntesis de controladores de DES, en el cual las variables involucradas son binarias; es decir, solo pueden tomar dos posibles valores ("1" ó "0", activo o inactivo, ON u OFF). En la Figura1 se muestra cómo en este caso el controlador lo constituye el PLC, y sus señales de entrada/salida lo son: la información proveniente de los sensores de campo (**eventos incontrolables y observables**) y las acciones del mando a los actuadores (**eventos controlables**), respectivamente.

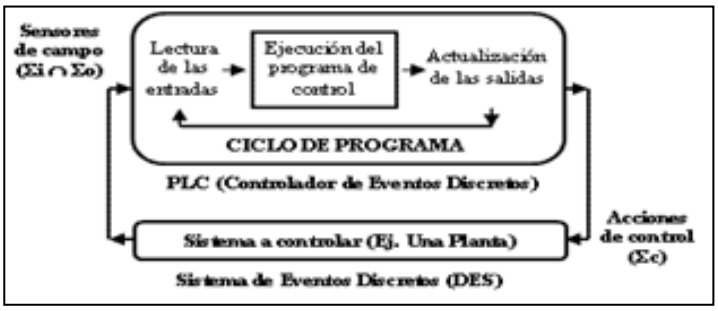


Figura 1: Control de un DES con PLCs.
Fuente: Autor, (2018).

Aunque son indiscutibles los logros alcanzados en este sentido, debido a las ventajas que introduce en cuanto a seguridad, calidad, generalidad, transparencia, tiempos y costos de ejecución de las soluciones obtenidas, estos métodos no ha sido ampliamente aplicados en la industria (Murillo, 2008). El hecho está dado principalmente por las limitaciones que aún existen en la correspondencia modelo PNs - programa de PLC, así como por la imposibilidad de tratar fielmente particularidades de este equipamiento, como lo son: el ciclo de programa y su paralelismo inherente, la jerarquía, la diferenciación de las acciones de impulso y nivel, la manipulación de información no solo binaria, la adherencia a la norma internacional IEC 61131-3, entre otras. Además, los métodos desarrollados hasta el momento resultan complejos de aplicar, no contando con la suficiente simplicidad, generalidad y facilidades de aplicación que les resulte atractivo a los profesionales de esta rama.

En este trabajo se presenta una metodología basada en modelos sobre la PNs jerárquica y extendida denominada GHENeSys IEC-1131 Compatible, como método formal de diseño de DES con PLCs, ésta persigue simplificar su aplicación y acortar la brecha entre el modelado en PNs y la norma internacional IEC 61131-3. El trabajo se organiza de la siguiente forma: en la sección 2 se explica la metodología propuesta, en la sección 3 se muestran los resultados de aplicación de dicha metodología sobre un caso de estudio. Finalmente se recogen las conclusiones del trabajo y las referencias bibliográficas en las secciones 4 y 5.

II. METODOLOGÍA

II.1. RED JERÁRQUICA EXTENDIDA GHENeSys IEC- 1131 COMPATIBLE

La red GHENeSys IEC 1131 Compatible tiene sus orígenes en la red jerárquica extendida GHENeSys (General Hierarchical Enhanced Net System) [8], es el resultado de expandir esta última al campo de aplicación de los PLCs y un acercamiento a la compatibilidad de las PNs y los lenguajes estandarizados en la norma internacional IEC 1131, para la cual se ha definido una metodología de diseño formal [9].

II.1.1. DEFINICIÓN - GHENeSys IEC-1131

GHENeSys IEC-1131 es compatible se define como una quíntupla N = (P, T, F, M0, Q), tal que:

P es un conjunto finito y no vacío de lugares, con P = Pc ∪ Pn ∪ Pm, siendo Pc, Pn y Pm conjuntos dados.

T es un conjunto finito y no vacío de transiciones, con T = Tc ∪ Tu, siendo Tc y Tu conjuntos dados. Por lo anterior, se cumple que P ∪ T ≠ Ø. Debe satisfacerse además que P ∩ T = Ø.

F es la relación de flujo y define un conjunto no vacío de arcos que interconectan los lugares a las transiciones y viceversa, y está compuesta por F ⊆ In(Pc x T) ∪ En(Pc x T) ∪ (Pn x T) ∪ (T x Pn) ∪ (Pm x T) ∪ (T x Pm) ≠ Ø.

M0 es un marcaje inicial.

Q es una función que asocia un conjunto finito de acciones de control a algunos lugares o al disparo de algunas transiciones, siendo Q = Ql ∪ Qp, con Ql = O ∪ Ō y Qp = S ∪ R.

En la Figura 2 se muestran los elementos que conforman una red GHENeSys IEC-1131 Compatible.

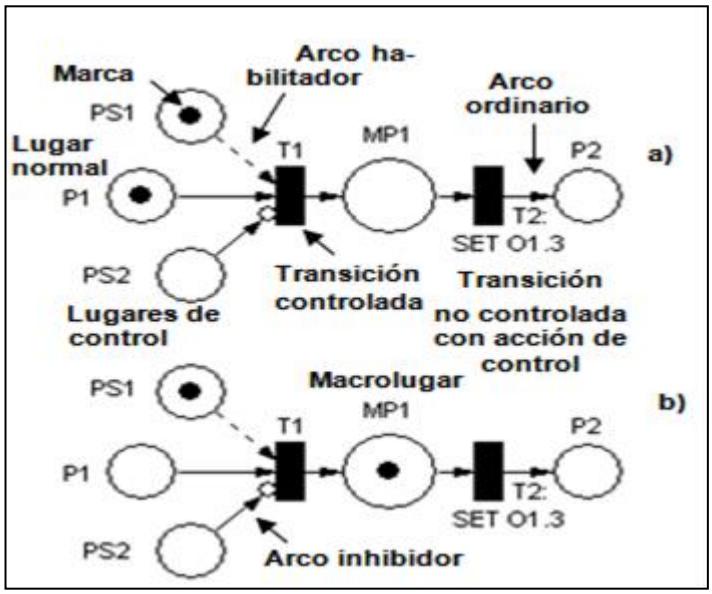


Figura 2: Elementos de GHENeSys IEC 1131 Compatible y disparo de una transición a) antes del disparo b) después del disparo.
Fuente: Autor, (2018).

Según el inciso i) los lugares pueden ser de tres tipos, lugares de control (Pc, Pc ⊂ P), lugares normales o comunes (Pn, Pn ⊂ P) y los lugares "macros" o macrolugares (Pm, Pm ⊂ P), representados por círculos de menor, mediano y mayor tamaño, respectivamente. Seguidamente se define su significado:

- Los Pc (llamados en GHENeSys Pseudoboxes) son lugares de marcaje persistente, pues no lo pierden con el disparo de las transiciones, se identifican con las letras PS y pueden representar:
- Información que brindan los sensores del proceso (bits asociados a las entradas del PLC, ejemplo pulsadores, microswitch, etc.).
- Información de las señales de salida del PLC (bits relacionados con la activación de los actuadores o dispositivos de control).
- Información asociada a otras subredes que representan variables globales del sistema (dependencia funcional) u otras partes de la propia red (se corresponden con bits internos del PLC, conocidos también como "banderas").
- Los bits de salida de temporizadores y contadores.
- Las salidas binarias de otros bloques funcionales (Ej. comparación, suma, resta, etc.).
- Los Pn se identifican con la letra P y representan estados por los que transita el sistema o tareas a ejecutar por este.
- Los Pm se identifican con las letras MP y constituyen un caso especial de lugar al agrupar o encapsular varias operaciones u otras partes de la red (subredes). Tienen un tratamiento similar a los bloques funcionales del lenguaje LD, y pueden representar:
- Llamadas a módulos o subredes de menor jerarquía, relacionados típicamente con tareas independientes, subprogramas o subrutinas.
- La activación de temporizadores, contadores y la ejecución de los demás bloques funcionales del PLC (Ej. comparación, suma, movimiento de datos, PID, etc.).

Las transiciones están divididas en dos subconjuntos, según el inciso ii) pueden ser transiciones controladas (Tc) o no controladas (Tu). Las controladas son aquellas cuyo conjunto de lugares de control de entrada sea no vacío, es decir, Tc := {t ∈ T | ${}^{(Pc)}t \neq \emptyset$}, mientras que las no controladas son aquellas cuyos lugares de entrada solo sean lugares normales o macrolugares (Tu := {t ∈ T | ${}^{(Pc)}t = \emptyset$ }).

El inciso v) comprende la función Q, que contempla los dos tipos de acciones posibles de utilizar en los PLC (de impulso o de nivel). Los valores que Q puede tomar son: "0" (desactivado) ó "1" (activado). Como ya se ha visto, el disparo de una transición t ∈ T ejecuta aquellas acciones de impulso (instrucciones del tipo SET o RESET del PLC, relacionadas con los subconjuntos S y R por ese orden) asociadas a ella (Qp(t) = S(t) ∪ R(t)), inmediatamente después de su disparo. Las acciones de nivel (instrucciones del tipo asignación y asignación negada, correspondiéndose respectivamente con los subconjuntos O y Ō), se ejecutarán en el momento que se deposita la marca en el lugar p ∈ Pn asociado a ellas (Ql(p) = O(p) ∪ Ō(p)), y permanecerán activadas mientras p se encuentre marcado (mientras M(p) = 1, con p ∈ Pn → Ql(p) estarán activadas). Es posible definir tantas acciones como se requieran en cada transición o lugar asociado a ellas.

II.1.2. DEFINICIÓN CONJUNTO DE LUGARES DE ENTRADA O PRECONDICIONES DE T

Para una transición (t ∈ T), se define el conjunto de lugares de entrada o precondiciones de t, como: ${}^{(p)}t := \{p \in P \mid (p,t) \in F$, con p ∈ Pc ∪ Pn ∪ Pm }. Mientras que el conjunto de lugares de salida o postcondiciones de t, se define como: $t^{(p)} := \{p \in P \mid (t,p) \in F$, con p ∈ Pn ∪ Pm}.

En otras palabras, los lugares comunes y los macrolugares pueden interconectarse tanto a la entrada como a la salida de las transiciones, y según el punto iii), se conectan a éstas por medio de arcos ordinarios ((Pn x T) ∪ (T x Pn) ⊂ F). Por otra parte, los lugares de control solo pueden ser lugares de entrada a las transiciones, y tomando en consideración el punto iii), se conectan a éstas solamente a través de arcos habilitadores (En (Pc x T) ⊂ F) o inhibidores (In(Pc x T) ⊂ F).

II.1.3. MODELO NO CONTROLADO (N_u) DE N

Se define el **modelo no controlado** (N_u) de N, a aquel conformado solamente por lugares comunes, transiciones no controladas y arcos ordinarios que los interconectan (no incorporan los lugares de control, los macrolugares ni las acciones); es decir, es aquel $N_u \subseteq N$ tal que: $N_u = (P_n, T_u, F_o, M_0)$, siendo $F_o \subseteq (P_n \times T) \cup (T \times P_n)$. Mientras que se define el **modelo controlado** (N_c) como aquel que hace uso de todos los elementos de GHENeSys IEC-1131 Compatible, incluyendo los lugares de control y las acciones ($N_c = N$).

De la definición anterior se establece que un modelo no controlado, comprenderá solamente aquellos estados por los que transita el sistema o tareas a ejecutar, no así los lugares de control. Por el contrario, el modelo controlado sí incorpora los lugares de control.

En la Figura 3 se muestra, de forma ilustrativa, la descomposición del modelo en subredes, la secuencia de ejecución que se sigue y el tratamiento dado a los bloques funcionales a partir de la subred asociada al arranque y parada de un motor. Notar la activación de un temporizador TON, y el chequeo de su bit asociado que indica la culminación del tiempo prefijado.

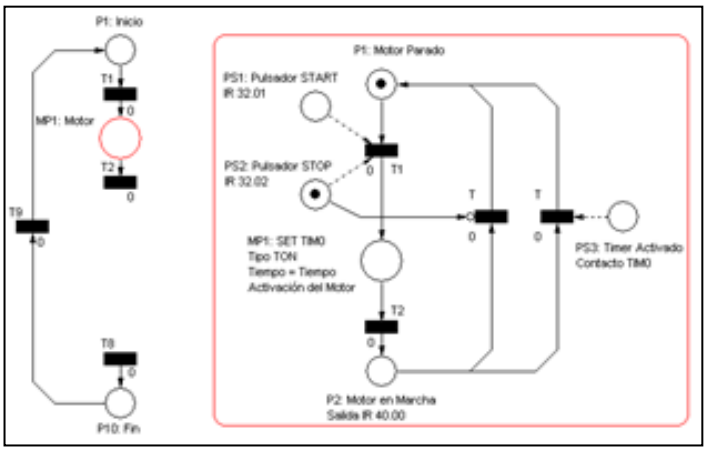


Figura 3: Secuencia de ejecución de GHENeSys IEC-1131 Compatible atendiendo a su nivel jerárquico.
Fuente: Autor, (2018).

II.2. MODELADO SOBRE GHENESYS IEC- 1131 COMPATIBLE PARA EL CASO PARTICULAR DE DES CON PLCS

El diseño formal de automatizaciones con PLCs comprende las fases mostradas en la Figura 4 [3]. Seguidamente trataremos aspectos relacionados con las etapas de formalización, verificación e implementación, para el caso que nos ocupa de los DES que emplean PLCs como dispositivos de control.

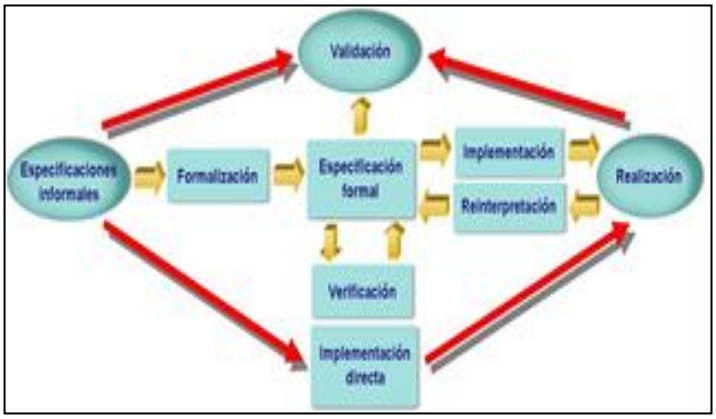


Figura 4: Fases del diseño formal de automatizaciones con PLCs.
Fuente: [3].

II.2.1. ETAPA DE FORMALIZACIÓN

Entre los elementos que recoge la metodología de diseño formal basada en GHENeSys IEC-1131 Compatible [9-11], se encuentra la construcción de modelos jerárquicos a partir de la descomposición en la mayor cantidad de subredes posibles, que se correspondan con los bloques funcionales del sistema; así como modelarlas, siempre que sea posible, en las clasificaciones más bajas de las PNs, con el propósito de facilitar su verificación, análisis e implementación posterior.

Los DES, dada su propia dinámica, donde por lo general cada bloque funcional que los forma cuenta con dos estados como: "activado" o "desactivado" (Ej. motor en marcha o detenido, válvula activada o desactivada, sensor activado o desactivado, etc.), siendo uno de ellos su estado inicial, al que usualmente se retorna en algún momento de su operación. Además, al manejar información binaria por la propia naturaleza de las variables implicadas en el proceso, permite definir para ellos un caso particular, en el que se pueden asumir las siguientes restricciones sobre el modelo:

Toda subred que lo forma se clasifica como **propia**, al contar con un lugar común de inicio y otro de fin del elemento que modela ($\forall$ SN $\subseteq$ N $\rightarrow \exists$ (P_{inicio} y P_{fin}) $\in$ Pn) [8]).

Para cada subred estará definido un marcaje inicial binario (M(0) = 1).

El modelo **no controlado** se clasifica entre las redes Máquinas de Estado (MS) [6], es decir, se circunscribe al grupo de las redes ordinarias (el peso de los arcos que lo conforman es unitario) y toda transición cuenta con un único lugar normal de entrada y uno de salida ($\forall$ t $\in$ T y p $\in$ Pn $\rightarrow$ $|^{(p)}t| = |t^{(p)}| = 1$).

De esta forma el espacio total de estados del sistema quedará conformado utilizando estructuras condicionales (de decisión o conflictos) (Murata, 1989). A los efectos del modelo, esto ocurre cuando varias transiciones comparten el mismo lugar de entrada (forman un conjunto C := {t | t $\in$ $^{(p)}$t, con p $\in$ Pn y |p| = 1}, siendo su cardinal |C| $\geq$ 2). Si tenemos en cuenta lo anterior, se puede plantear entonces que esta clasificación es suficiente para modelar prácticamente la totalidad de los programas de DES controlados por PLCs, al tener en cuenta que éstos son posibles de construir a partir de estructuras IF-THEN-ELSE anidadas [8].

Para que una transición t $\in$ T esté habilitada todas sus precondiciones deben ser "verdaderas", las restricciones asumidas reducen esto al cumplimiento de las siguientes reglas:

Para $^{(p)}$t, con p $\in$ Pn $\cup$ Pm y |p| = 1 $\rightarrow \exists$ M(p) = 1.

Para $^{(p)}$t, con p $\in$ Pc y En(p x t) $\in$ F $\rightarrow \exists$ M(p) = 1.

Para $^{(p)}$t, con p $\in$ Pc y In(p x t) $\in$ F $\rightarrow \exists$ M(p) = 0.

El disparo de una transición t $\in$ T provocará entonces:

La eliminación del marcaje en su lugar de entrada que no sea de control (para p $\in$ $^{(p)}$t y p $\in$ Pn $\cup$ Pm $\rightarrow$ M(p) = 0) .

El depósito de una marca en su lugar de salida (para aquel p $\in$ $t^{(p)}$ $\rightarrow$ M(p) = 1).

La ejecución de aquellas acciones de impulso asociadas a t inmediatamente después de su disparo (Qp(t)).

La ejecución de aquellas acciones de nivel asociadas al lugar de salida, si este es un lugar común (Ql(p), con p $\in$ Pn), o el bloque funcional modelado por un macrolugar (Pm), si fuera este el caso, al instante en que se deposita la marca en él.

II.2.2. ETAPA DE VERIFICACIÓN

La verificación es la etapa donde se comprueban sobre el modelo propiedades formalizadas, como es el caso de vivacidad y seguridad [3][9][10], las que resultan las principales a comprobar para el caso de los programas de automatizaciones con PLCs.

La verificación para GHENeSys IEC-1131 Compatible se realiza sobre el **modelo no controlado**, al considerar que formalmente no existe diferencia entre el modelo controlado y el no controlado, pues en ambos se recoge el comportamiento

deseado del sistema a controlar y abarcan el espacio total de estados del sistema modelado ($|Pn \in N_c| = |P \in N_u|$) [9-11].

II.2.1.1 PROPOSICIÓN

Un modelo construido basado en las restricciones recogidas en el acápite 2.2.1, es posible verificarlo a partir del método de las técnicas de reducción o descomposición, al estar conformado por redes puras [6][7].

Demostración**:** El modelo antes mencionado queda libre de autolazos al ser clasificada toda subred que lo integra como propia, contiene al menos dos lugares (uno de inicio y otro de fin) y una transición ($\forall$ SN $\subseteq$ N $\rightarrow \exists$ (P_{inicio} y P_{fin}) $\in$ Pn).
estar restringido a redes MS ($\forall$ t $\in$ T y p $\in$ Pn $\rightarrow |{}^{(p)}t| = |t^{(p)}| = 1$),
Por ello, $\forall$ t $\in$ T y p $\in$ P $\rightarrow {}^{(p)}t \neq t^{(p)}$.

II.2.2.2 PROPOSICIÓN DE UNA RED MÍNIMA

Una **red mínima** (N_{min}) es aquella que se logre reducir solamente a un lugar y una transición ($N_{min} \subseteq$ N tal que: N_{min} = ($\forall$ t $\in$ T y p$\in$ Pn $\rightarrow$ |t| = |p| = 1. Si esto se logra es porque la red resulta reversible; es decir, regresa a su marcaje o estado inicial [6], y por tanto, es **fuertemente conectada** [6], pues para dos nodos cualesquiera de la red existirán caminos directos que los interconecten.

Partiendo de lo anterior, sobre una red reducida a mínima, y por tanto, fuertemente conectada, se pueden comprobar las propiedades de vivacidad y seguridad aplicando los teoremas de Murata [6] para redes MS (según las restricciones asumidas en el acápite 2.2.1). Al poseer un marcaje inicial la red será **viva**, y **segura** al contar dicho marcaje con solo una marca.

II.2.2.2.1 TEOREMA

El marcaje inicial binario resulta condición necesaria y suficiente para que toda red que se clasifique como MS y se logre reducir a mínima sea viva y segura.

Otra de las propiedades importantes de todo programa de PLC es su determinismo. Esto significa que debe responder de igual manera ante las mismas condiciones de operación; es decir, al encontrarse en un estado determinado, para cada combinación de sus entradas debe responder con idéntica combinación de sus salidas. Si esto no se cumple, entonces el controlador tendrá una respuesta inespecífica en determinadas situaciones de operación, dependiendo para su correcto funcionamiento del posible estado aleatorio de ciertos elementos de su implementación [12][13].

Proposición: Una red GHENeSys IEC-1131 Compatible, que es segura y cumple las restricciones asumidas en el acápite 2.2.1 es determinística, si el **modelo controlado** satisface las siguientes condiciones:

- Para cada estructura condicional (p, {t1, t2, ...}, con p $\in$ Pn, Pm), las condiciones asociadas con t1, t2, ... resultan mutuamente excluyentes.
- Para cada par de lugares (P_i y P_j, con p $\in$ Pn, Pm), tal que sus operaciones sean incompatibles, se cumpla que $M(P_i) + M(P_j) \leq 1$.

Lo anterior garantiza que el disparo de una transición no dependa del marcaje de su postcondición, pues solamente estará marcado un lugar común o macrolugar a la vez ($\forall$ P_i, $P_j \in$ Pn $\cup$ Pm $\rightarrow M(P_i) + M(P_j) = 1$). Esto posibilita la obtención de redes seguras desde la propia etapa de formalización y facilita además la implementación posterior a su programa en código de PLC.

Para la obtención de un modelo más refinado jerárquicamente es común su descomposición en subredes de diferentes niveles jerárquicos. Una subred es una abstracción que se utiliza para modelar una parte de la red que por razones de simplicidad, transparencia o conveniencia se sustituye por un elemento de red que la identifica. Esto evita el problema de la explosión de estados (elevado número de estados que imposibilita su análisis y comprensión), al poder analizar las subredes por separado con mayor facilidad. El resultado global del sistema se obtiene unificando los resultados parciales de cada subred analizada.

II.2.2.2.2 TEOREMA

Si partimos del hecho que un modelo sobre PNs está conformado por el conjunto total de subredes para él definidas (N = $SN_1 \cup SN_2 \cup \ldots \cup SN_n$, siendo n el número total de subredes que lo integran), es posible plantear que si toda SN $\in$ N es viva, segura y determinística, entonces N también lo será.

II.2.3 ETAPA DE IMPLEMENTACIÓN

La implementación tiene el objetivo de traducir el modelo obtenido a su correspondiente programa en código PLC [3], y se conforma combinando las estructuras típicas AND y OR que han sido definidas para el modelo [10][12]. Para una estructura AND como se muestra en la Fig. 5 a) quedará de la siguiente forma:

If $M(P_i) = 1$ and $\forall$ $M(P_{en}) = 1$ and $\forall$ $M(P_{in}) = 0$, con P_{en} := {Pc $\in {}^{(P)}$ti | En(Pc,ti) $\in$ F} y P_{in} := {Pc $\in {}^{(P)}$ti | In(Pc,ti) $\in$ F}
Then RESET[(Pi) and $\forall$ R(ti)] y SET[(Pj) and $\forall$ S(ti)]

If $M(P_j) = 1$, con $P_j \in$ Pn $\cup$ Pm
Then [OUT($\forall$ $O(P_j)$] and [neg_OUT($\forall$ $\bar{O}(P_j)$], para Pj $\in$ Pn. Si Pj $\in$ Pm se ejecuta la función que modela el macrolugar.

Para una combinación AND-OR del tipo mostrado en la Fig. 4 b), resulta como sigue:

If $M(P_i) = 1$ and [($\forall$ $M(P_{en1}) = 1$ and $\forall$ $M(P_{in1}) = 0$) or ($\forall$ $M(P_{en2}) = 1$ and $\forall$ $M(P_{in2}) = 0$) or … or ($\forall$ $M(P_{enn}) = 1$ and $\forall$ $M(P_{inn}) = 0$] con
P_{eni} := {Pc $\in {}^{(P)}$ti | En(Pc,ti) $\in$ F} y P_{ini} := {Pc $\in {}^{(P)}$ti | In(Pc,ti) $\in$ F} y i = 1,2,…,n
Then RESET[(P_i) and $\forall$ $R(t_i)$] y
SET[(P_j) and $\forall$ $S(t_i)$], cumpliendo que:
$t_1 = t_2 = \ldots = t_n$; por tanto,
$R(t_1) = R(t_2) = \ldots = R(t_n)$ y

III RESULTADOS Y DISCUSIÓN

Para ilustrar los elementos de la metodología de diseño formal a partir de GHENeSys IEC-1131 Compatible para DES con PLCs, descritos en el acápite 2, nos auxiliaremos del caso de estudio mostrado en la Figura 5. Este problema didáctico, tomado de [12], consta de un carro que se mueve cíclicamente en una plataforma de izquierda a derecha. En su estado inicial puede encontrarse arbitrariamente en cualquier posición dentro del recorrido permisible en la plataforma. Al presionar el pulsador de START el carro se moverá inicialmente a la derecha, y una vez activado el sensor de fin de carrera en ese extremo, se detendrá por un tiempo de 20 seg. para la carga y descarga de los pasajeros, posteriormente se moverá hacia la izquierda, donde al activarse el

sensor de fin de carrera en ese extremo, esperará igual tiempo antes de retornar a la derecha. El carro ejecutará el ciclo anterior hasta que se oprima el pulsador de STOP o se ejecute dicho ciclo un número de 10 veces, deteniéndose en ese último caso a la izquierda. Se mantendrá en ese estado hasta que se presione nuevamente el pulsador de START, repitiéndose nuevamente el proceso descrito anteriormente.

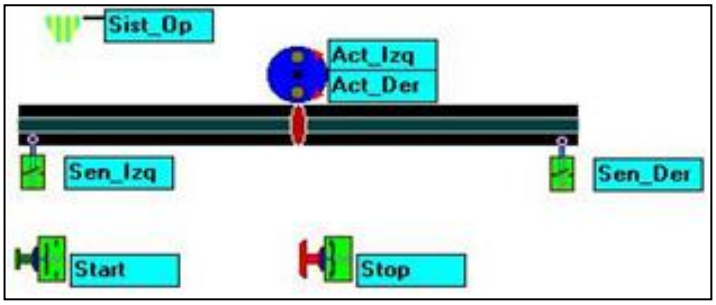


Figura 5: Sistema de control de carro que se mueve en 2 sentidos.
Fuente: [12].

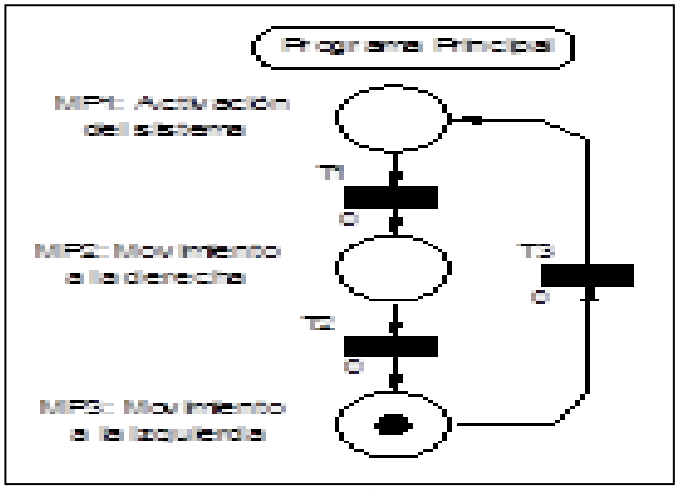
Para garantizar las especificaciones anteriores, se consideraron las señales que aparecen en la Tabla II.

Tabla 2: Señales considerado.

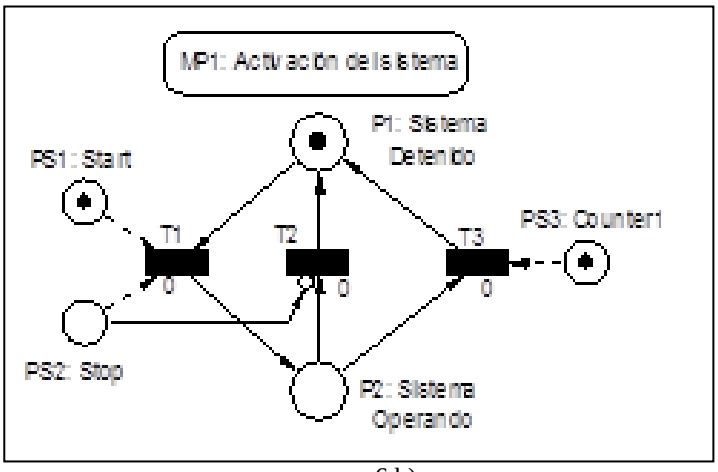
Nro	Señal	Identif	Tipo
1	Pulsador START	Start	ED
2	Pulsador STOP	Stop	ED
3	Sensor fin carrera derecho	Sen_Der	ED
4	Sensor fin carrera inquierdo	Sen_Izq	ED
5	Activación movimiento derecha	Act_Der	SD
6	Activación movimiento izquierda	Act_Izq	SD
7	Sistema operando	Sist_Op	Bandera
8	Timer detención carro	Timer1	Temporizador
9	Contador ciclos	Counter1	Contador

Fuente: Autor, (2018).

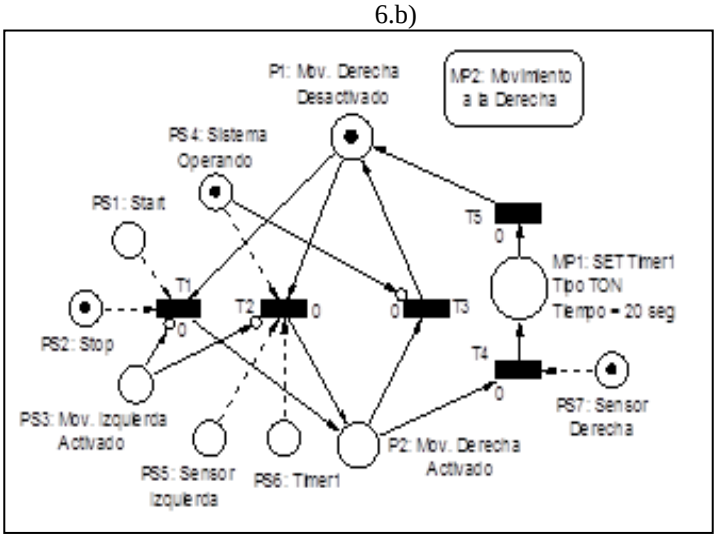
En la Figura 5 se muestra modelo desarrollado que satisface dichas especificaciones. En el modelo se consideraron 3 subredes (Figura 6 a)), que se corresponden con los bloques funcionales: activación del sistema (Figura 6 b)), movimiento del carro a la derecha (Figura 6 c)) y movimiento del carro a la izquierda (Figura 6 d)).



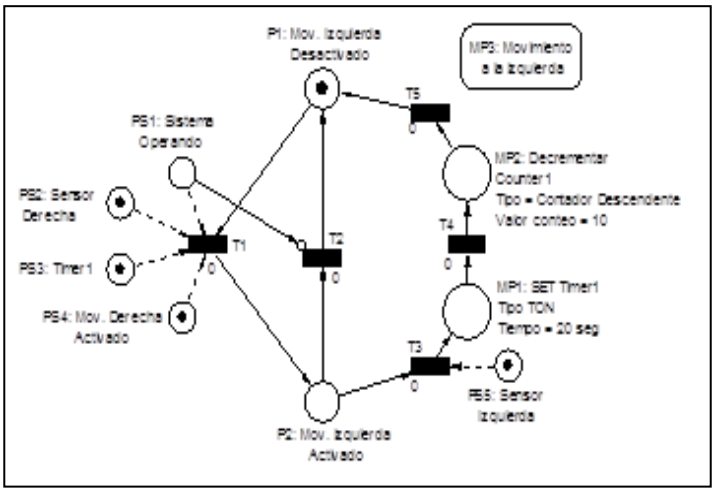
6.a)



6.b)



6.c)



6.d)

Figura 6: a) Programa principal, b) subred activación del sistema, c) subred movimiento a la derecha, d) subred movimiento a la izquierda.
Fuente: Autor, (2018).

Por su parte, la Figura 8 muestra el programa asociado al modelo anterior.

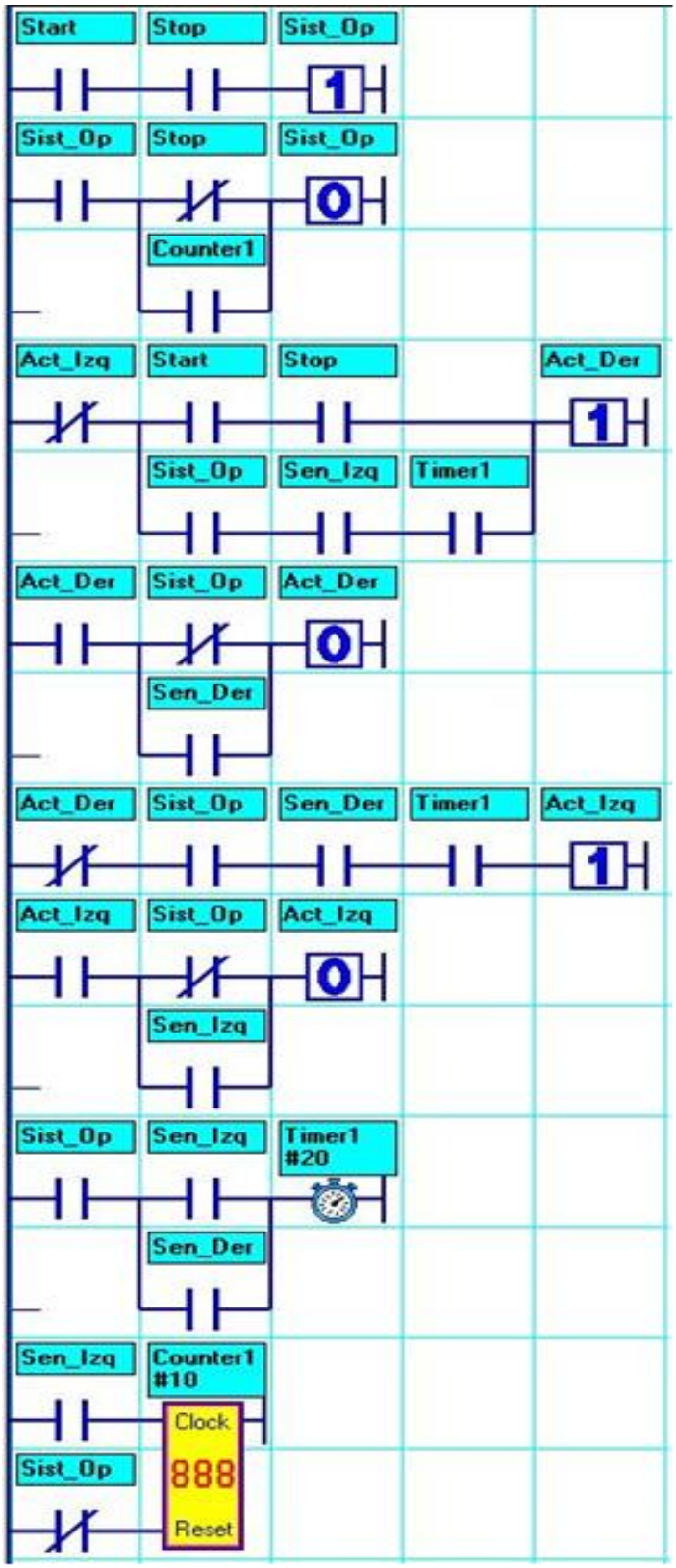


Figura 7: Programa del modelo de la Figura 6, a) subred activación del sistema, b) subred movimiento a la derecha, d) subred movimiento a la izquierda.
Fuente: Autor, (2018).

IV. CONCLUSIONES

Se presentarán las conclusiones que se deriven del trabajo realizado. Toda conclusión debe estar fundamentada en lo expuesto y discutido en el trabajo y debe reflejar el cumplimiento de los objetivos.

V. REFERENCIAS

[1] Mintchel, G.A., "**Power Up Programming with Graphical Modeling**". Control Engineering, Diciembre, Vol 1. 2000.

[2] Dick Johnson, "**Research on Programmable Logic Controllers**". Control Engineering. 2007.

[3] Frey, G. and Litz, L., "**Formal Methods in PLC Programming**". Proceeding of IEEE Conference on Systems Man and Cybernetics, SMC'2000, Nashville, October 8-11. 2000.

[4] Murillo, L. D., "**Redes de Petri: Modelado e implementación de algoritmos para autómatas programables**". Revista Tecnología en Marcha, Vol. 21, No. 4, Octubre-Diciembre 2008, pp. 102-125.

[5] Ramadge, P.J. y Wonham, W.M., "**The Control of Discrete Event Systems**". Proceeding of the IEEE, Vol. 77, 1989, pp.81-97.

[6] Murata, T., "**Petri Nets: Properties, Analysis and Applications**". Proceeding. of IEEE, Vol. 77, No. 4, Abril. 1989.

[7] Desel, J. y Esparza, J., "**Free Choice Petri Nets**". Cambridge University Press. Great Britain. 1995.

[8] González, P. y Silva J.R., "**GHENeSys: Uma Rede Estendida para a Modelagem, Analise e Projeto de Sistemas Complexos**". Proceeding of SBAI'2001, Sao Paulo, November, 2001.

[9] Avila, E., "**Modelado en PNs de Automatizaciones con PLCs de la Industria Azucarera**". Tesis en opción del Título de Master en Automática. Departamento de Control Automático. Facultad de Ingeniería Eléctrica. Universidad de Oriente. Santiago de Cuba. Julio. 2002.

[10] Avila, E.; Benítez, I. y Silva, J.R., "**Verificación de Programas de PLCs Modelados sobre Redes de Petri. Métodos Gráficos vs Algebraicos**". Memorias XI Congreso Latinoamericano de Control Automático, CLCA'2004, Mayo, 2004.

[11] Avila, E.; Benítez, I. y Silva, J.R., "**Modelado en Redes de Petri como Método Formal de Automatizaciones de Edificios Inteligentes**". Memorias II Taller Internacional de Automática para el Ahorro Energético, INFO'2005, Mayo, 2005.

[12] Lee, G. B. et al, "**Automatic Generation of Ladder Diagram with Control Petri Net**". Journal of Intelligent Manufacturing, No. 15, pp. 245-252, 2004.

[13] Frey, G. y Litz, L., "**Corretness Analisys of Petri Net Based Logic Controllers**". Proceeding of American Control Conference ACC 2000, Chicago, pp. 28-30, Junio, 2000.