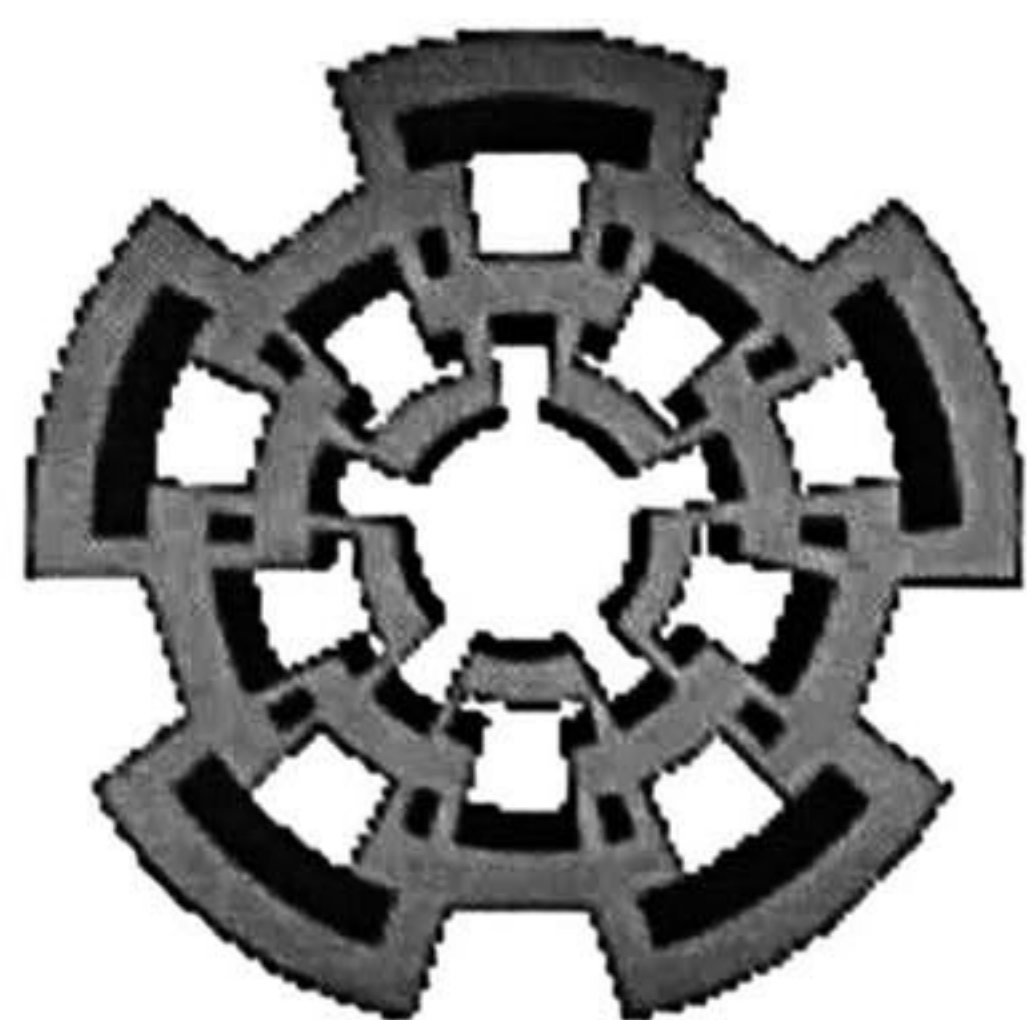


xx(79888.1)



CINVESTAV

Centro de Investigación y de Estudios Avanzados del IPN
Unidad Guadalajara

CINVESTAV I.P.N.
SECCION DE INFORMACION
Y DOCUMENTACION

Seguimiento de Modelo en SED usando RPI

Tesis que presenta
Alejandra Santoyo Sanchez



Para obtener el grado de
Maestro en Ciencias

En la especialidad de
Ingeniería Eléctrica

Guadalajara, Jal., Julio de 1999

CLASIF.	
ADQUIS.	TESIS-1999
FECHA:	19-x-99
PROCED.	Depto. Serv.
	5

Bib1

Seguimiento de Modelo en SED usando RPI

**Tesis de Maestría en Ciencias
Ingeniería Eléctrica**

Por:

Alejandra Santoyo Sanchez

Licenciado en Informática
Universidad de Guadalajara 1993-1997

Becario del CONACyT, expediente no. **121139**

Directores de Tesis

Dr. Antonio Ramírez Treviño

Dra. Ofelia Begovich Mendoza

CINVESTAV del IPN Unidad Guadalajara, Julio de 1999

Agradecimientos

Primeramente a Dios por haberme permitido llegar a mi meta.

A mis padres y hermanos por su apoyo incondicional.

A mis asesores, Dr. Antonio Ramírez Treviño y Dra. Ofelia Begovich Mendoza, por ayudarme a llevar a buen fin el presente trabajo.

A los revisores de la tesis, Dr. Luis Ernesto López Mellado, Dr. Ricardo Raúl Jacinto Montes y Dr. Raúl Ernesto González Torres, por ayudarme a detectar y corregir los errores de este trabajo.

A los profesores del CINVESTAV y a todos aquellos que han ayudado a mi formación académica y personal.

A María Elena y Luis Isidro por sus sugerencias.

A mi “hermanito” Memo, sin su ayuda no hubiese corregido varios errores.

A Toñito sin su ayuda no hubiese salido el ejemplar.

A Lupita y Tere por su ayuda durante mi estancia en CINVESTAV.

Finalmente Carlos Alberto, Santiago, Juanito, Chava, Ana Raquel, José Olivas, Aide, Marcos, Adolfo e Isidro por haberme soportado durante este tiempo.

A todo ellos muchas gracias.

Índice de Figuras

1.1	Sistema “ <i>Nivel de líquido</i> ” descrito como un <i>SC</i> .	5
1.2	Sistema “ <i>Nivel de líquido</i> ” descrito como <i>SED</i> .	6
1.3	<i>AF</i> del sistema “ <i>Nivel de líquido</i> ”	8
1.4	<i>RP</i> del sistema “ <i>Nivel de líquido</i> ”	9
1.5	Esquema de un control supervisor.	11
2.1	<i>RP</i> viva, no viva y parcialmente viva.	18
2.2	<i>RP</i> no-limitada, 1-limitada y 2-limitada.	19
2.3	<i>RPI</i> del sistema “ <i>Nivel de líquido</i> ”	21
2.4	<i>AF</i> de máquina.	24
2.5	<i>AF</i> de la especificación.	24
2.6	<i>AF</i> de la intersección del supervisor y el sistema.	25
2.7	Tipos de lenguajes	27
2.8	<i>RP</i> de máquina.	28
2.9	Especificación.	28
2.10	Comportamiento del sistema bajo supervisión.	29
2.11	Sistema de Manufactura.	30
2.12	Modelo del sistema de manufactura.	31
2.13	Modelo de referencia o especificación.	32
2.14	Supervisor para el sistema de manufactura.	32
3.1	Diagrama de funciones φ , φ_{S_f} y VR .	37
3.2	<i>RPI</i> de la válvula 1 .	39
3.3	<i>RPI</i> del modelo interno del sistema “ <i>Nivel de líquido</i> ”.	39
3.4	<i>RPI</i> del sistema “ <i>Nivel de líquido</i> ”	40
3.5	Sistema “ <i>Coche eléctrico</i> ”	41
3.6	<i>RPI</i> de los actuadores Derecha e Izquierda .	41
3.7	<i>RPI</i> del modelo interno del sistema “ <i>Coche eléctrico</i> ”	41
3.8	<i>RPI</i> del sistema “ <i>Coche eléctrico</i> ”	42
3.9	Esquema del problema de “ <i>seguimiento de modelo con información total del estado</i> ”	46
4.1	Especificación.	61
4.2	Marcado actual del sistema de manufactura.	61
4.3	Esquema para el problema de seguimiento de modelo para el “ <i>Sistema de manufactura</i> ”	62
4.4	Sistema “ <i>Coche eléctrico</i> ”	66

4.5	<i>RPI</i> del sistema “Coche eléctrico”	67
4.6	<i>RPI</i> del sistema “Coche eléctrico”	67
4.7	Especificación	67
4.8	Esquema para el problema de seguimiento de modelo para el sistema “Coche eléctrico”	68

Índice General

0.1	Abreviaturas	1
0.2	Notación	2
1	Panorama general	3
1.1	Introducción	4
1.2	Sistemas de Eventos Discretos	4
1.3	Técnicas para modelado de <i>SED</i>	6
1.3.1	Técnicas basadas en estados	6
1.3.2	Técnicas Híbridas	8
1.4	Ventajas y desventajas de los formalismos de modelado	8
1.5	Estado del arte en control de <i>SED</i>	9
1.5.1	Controladores para <i>SED</i>	10
1.6	Definición del problema .	12
1.7	Objetivos y metas	12
1.8	Conclusiones .	12
2	Preliminares matemáticos	13
2.1	Introducción	14
2.2	Lenguajes, cadenas y proyecciones	14
2.2.1	Operaciones con cadenas y lenguajes	15
2.3	Redes de Petri	15
2.3.1	Red de Petri ordinaria	15
2.3.2	Redes de Petri Interpretadas	19
2.4	Teoría de control supervisor	22
2.4.1	Supervisor en autómatas finitos	22
2.4.2	Supervisor en redes de Petri	24
2.5	Conclusiones	30
3	El problema de seguimiento de modelo	35
3.1	Introducción	36
3.2	Definiciones Básicas .	36
3.2.1	La función φ	36
3.2.2	Ejemplo de modelado	38
3.2.3	Lenguajes de una <i>RPI</i>	41
3.2.4	Controlabilidad	43
3.3	El problema de seguimiento de modelo	44
3.4	Condiciones para el seguimiento de modelo	45





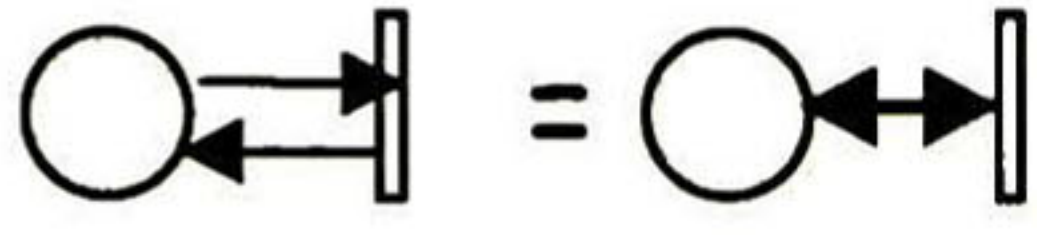
3.5 Conclusiones .	51
4 Algoritmo para el seguimiento de modelo	53
4.1 Introducción	54
4.2 El algoritmo	54
4.3 Ejemplo	55
4.4 Conclusiones	66
5 Conclusiones y trabajo futuro	69
5.1 Conclusiones	69
5.2 Trabajo Futuro	70

0.1. ABREVIATURAS

0.1 Abreviaturas

<i>ABREVIATURA</i>	<i>DESCRIPCIÓN</i>
<i>RP</i>	Red de Petri
<i>RPI</i>	Red de Petri Interpretada
<i>SC</i>	Sistemas Continuos
<i>SED</i>	Sistema de Eventos Discretos
<i>SM</i>	Sistema de Manufactura
<i>SMF</i>	Sistema de Manufactura Flexible

0.2 Notación

<i>SÍMBOLO</i>	<i>DESCRIPCIÓN</i>
R_m	Modelo en <i>RPI</i> que representa el modelo a seguir.
S_f	Modelo en <i>RPI</i> que representa el modelo del sistema a controlar.
$L(M_n^*)$	Lenguaje generado a partir del marcado n en la <i>RPI</i> denotada como $*$ formada por las salidas del sistema.
M_n^*	Vector que contiene el conjunto de marcas asociadas a la <i>RPI</i> denotada como $*$ en el instante n .
σ_n	Palabra de entrada, equivale a $\sigma_n = \lambda(t_1) \lambda(t_2) \dots \lambda(t_n)$.
w_k	Palabra de salida, equivale a $\varphi(M_n) \varphi(M_{n+1}) \dots \varphi(M_{n+k})$, donde M_n es el marcado actual.
T_N	Conjunto de transiciones de la <i>RPI</i> denotada como N .
\mathbb{N}	Conjunto de números naturales.
$M_i \xrightarrow{t_i}$	El marcado M_i habilita a la transición t_i .
$\neg M_i \xrightarrow{t}$	El marcado M_i no habilita ninguna transición.
$M_i \xrightarrow{t_i} M_j$	El marcado M_j es alcanzado desde M_i por el disparo de la transición t_i .
Σ_*	Alfabeto de entrada o alfabeto de señales de los actuadores de una <i>RPI</i> denotada como $*$.
Φ_*	Alfabeto de salida o alfabeto de señales de sensores de una <i>RPI</i> denotada como $*$.
$[*]^n$	Es un vector de dimensión n que contienen elementos del conjunto $*$.
\mathcal{L}_{DP}	En <i>RP</i> lenguaje de bloqueo y prefijo.
	En <i>AF</i> se utiliza para indicar que el estado es inicial.
	En <i>AF</i> se utiliza para indicar que el estado es marcado.
	En <i>AF</i> se utiliza para indicar que el estado es inicial y marcado.
	En <i>RP</i> simboliza un arco del lugar a la transición y viceversa, es decir:
	

Capítulo 1

Panorama general

Resumen: Este capítulo presenta brevemente los conceptos básicos de sistemas de eventos discretos así como las herramientas que existen para modelarlos y analizarlos. También presenta brevemente las metodologías que hasta el momento abordan el problema de control en los sistemas de eventos discretos y los principales problemas prácticos que se tienen para su implementación. Finalmente, define el problema de *seguimiento de modelo* y las ventajas de este esquema de control.

1.1 Introducción

Cuando usted está usando una computadora y con el *Explorador* desea copiar un archivo de la unidad *C* a la unidad *A*, entonces usted comienza a dar órdenes a la computadora para indicarle qué acciones deberá tomar para copiar el archivo. Primero abre el *menú* que aparece al seleccionar con el ratón el botón de *Inicio*, posteriormente selecciona la opción *Programas*, para finalmente entrar al *Explorador* y copiar el archivo. Se puede decir que usted está dando una serie de órdenes a la computadora para que se comporte como usted quiere, es decir usted está controlando a la computadora.

Hablando en términos de *Sistemas de Eventos Discretos (SED)*, usted es un *supervisor* que le indica al sistema (computadora) qué hacer. Además tanto la computadora como usted se están comportando como *SED's*; puesto que las órdenes (numerables, indivisibles) y los estados que alcanza la computadora (en espera, copiando, finalización de tarea, falla, etc.) se pueden ver como operaciones atómicas y numerables.

Cada una de las órdenes que emite el operador es un *evento*, y particularmente un *evento controlable* porque el supervisor (operador) tiene la capacidad de habilitarlo o deshabilitarlo. Durante el desarrollo de la tarea pueden ocurrir eventos no esperados, que provoquen retraso o que no permitan copiar el archivo (e.g. se interrumpe el suministro de energía eléctrica, etc.). Estas operaciones que no emite el operador (por medio de órdenes), pero que causan cambios en el estado del sistema se les denomina *eventos incontrolables* (o eventos internos).

Como puede observarse, en un *SED* existen acciones o actividades que hacen que el estado del sistema cambie. A estas acciones se les llama *eventos*. A su vez, el estado del sistema es una descripción de la actividad que éste realiza. En este caso, el *SED* es la computadora y contiene los eventos: *abrir menú*, *copiar archivo* y los estados son *archivo copiado*, *disco A dañado*, *disco A sin espacio*, entre otros. Se debe notar que los eventos son acciones (aquí denotados por verbos) mientras que los estados son los diferentes escenarios que puede alcanzar el sistema.

Por otro lado, también se observa que un *controlador* en este contexto, es un *SED*, el cuál dependiendo del estado del sistema a controlar, emite una secuencia de eventos (ley de control) que permiten que el *SED* alcance el estado deseado.

1.2 Sistemas de Eventos Discretos

Definición 1.1 *Un SED es aquel cuyo espacio de estados es numerable (aunque posiblemente infinito), y donde el estado cambia abruptamente en respuesta a eventos que ocurren, en general, de manera asíncrona, el paradigma de ecuaciones diferenciales no sirve para modelarlos, la activación de eventos no depende del tiempo [9].*

De acuerdo con esta definición, los sistemas de información, de manufactura, de transporte, logísticos, de protocolos de comunicación, entre muchos otros, pertenecen a esta clase de sistemas.

“Por ejemplo, tomemos el caso de un aeropuerto con cinco puertas de embarque, una pista para aterrizar y otra para despegar, donde los eventos pueden ser: *avión aterrizando*, *avión despegando*, *inicio de embarque*, *fin de embarque*. El estado del aeropuerto se puede describir como el siguiente vector:

$$e = \begin{bmatrix} \text{número de aviones en puertas de embarque,} \\ \text{avión en pista de despegue,} \\ \text{avión en pista de aterrizaje} \end{bmatrix}$$

Como se observa el estado es numerable ya que e sólo tiene entradas enteras. El estado pudiera ser en un momento dado: $e_1 = [3, 0, 0]^T$ que se interpreta como tres aviones en puerta de embarque, ningún avión en pista de despegue y ningún avión en pista de aterrizaje. Al darse el evento *avión aterrizando* el estado cambia a $e_2 = [3, 0, 1]^T$, es decir, tres aviones en puertas de embarque, ningún avión en pista de despegue y un avión en pista de aterrizaje.

Sin embargo no todos los sistemas son de eventos discretos, tomemos por ejemplo el sistema de nivel de la figura 1.1. De acuerdo con [16] éste se puede modelar alrededor de un punto de operación como:

$$\dot{h}(t) = \frac{-1}{AR}h(t) + \frac{1}{A}q_1(t)$$

Donde q_1 representa el flujo de entrada, A es el área de la base del tanque, h es la altura o nivel del líquido y R es la resistencia al flujo de salida. El número de valores que puede tomar h no es numerable, por lo tanto no es un *SED*.

A los sistemas que pueden modelarse mediante el uso de ecuaciones diferenciales se les llama *sistemas continuos (SC)*, ya que los eventos o acontecimientos que tienen que ver con cambios en el sistema están directamente relacionados con un tiempo continuo.

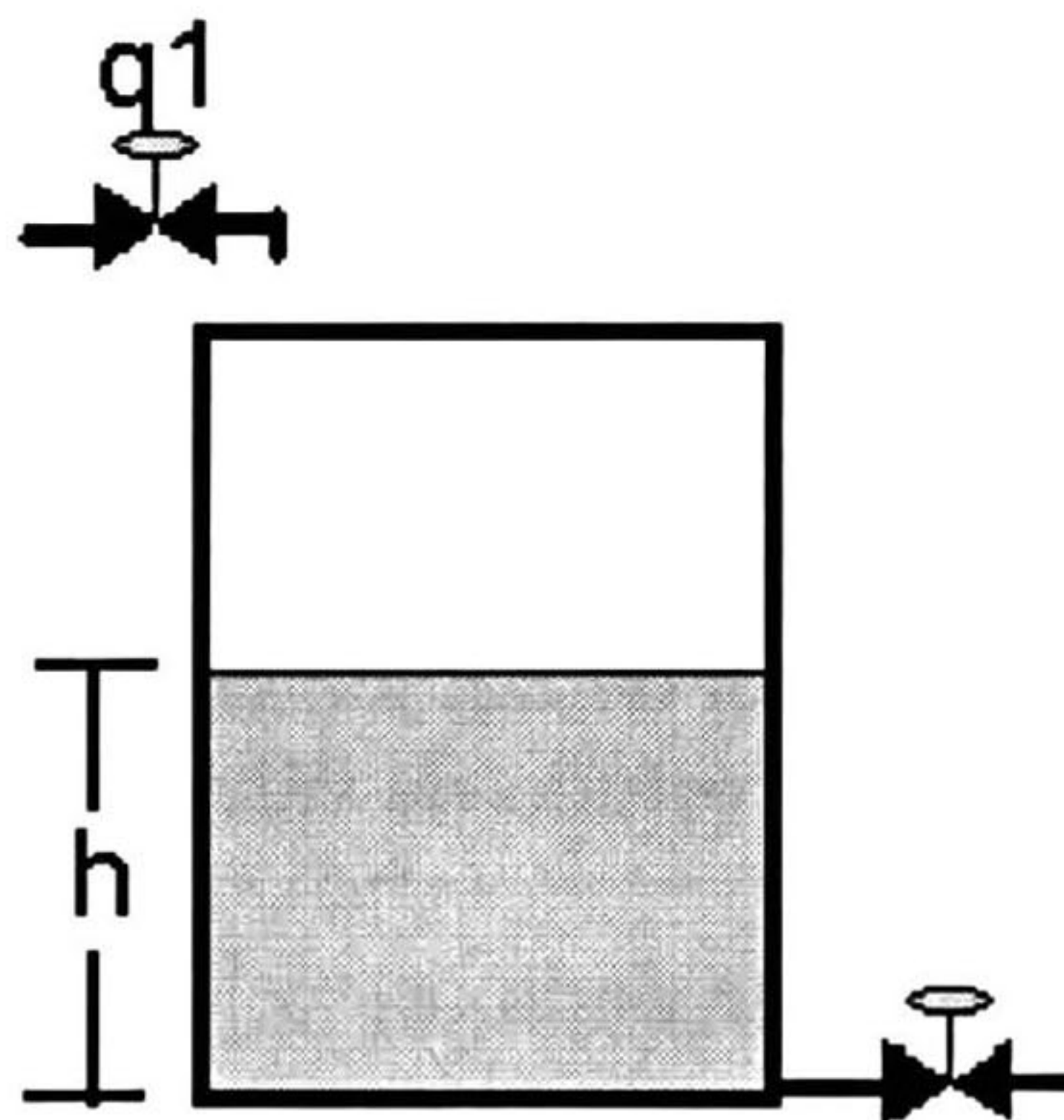


Figura 1.1: Sistema “Nivel de líquido” descrito como un *SC*.

Sin embargo, en algunos casos, el comportamiento de un *SC* se puede abstraer y sus estados pueden inducir una clase de índice finito, por tanto se obtendrá una cantidad de estados numerable (no es que ocurra así, sólo es que se ha abstraído el comportamiento a un número finito de estados).

La figura 1.2 muestra el ejemplo de un tanque de agua visto como *SED* basado en tres estados del nivel: *vacío* (cuando el agua no alcanza ninguno de los sensores); *bajo* (cuando el agua rebasa sólo al sensor **Bajo**); y *alto* (cuando el agua rebasa al sensor **Alto**); también se tienen dos estados de la válvula 1: *abierta* o *cerrada*: La válvula 2 tiene un estado fijo y no se puede mover.

A continuación se muestra cómo modelar este sistema usando algunas de las herramientas que existen para tal efecto.

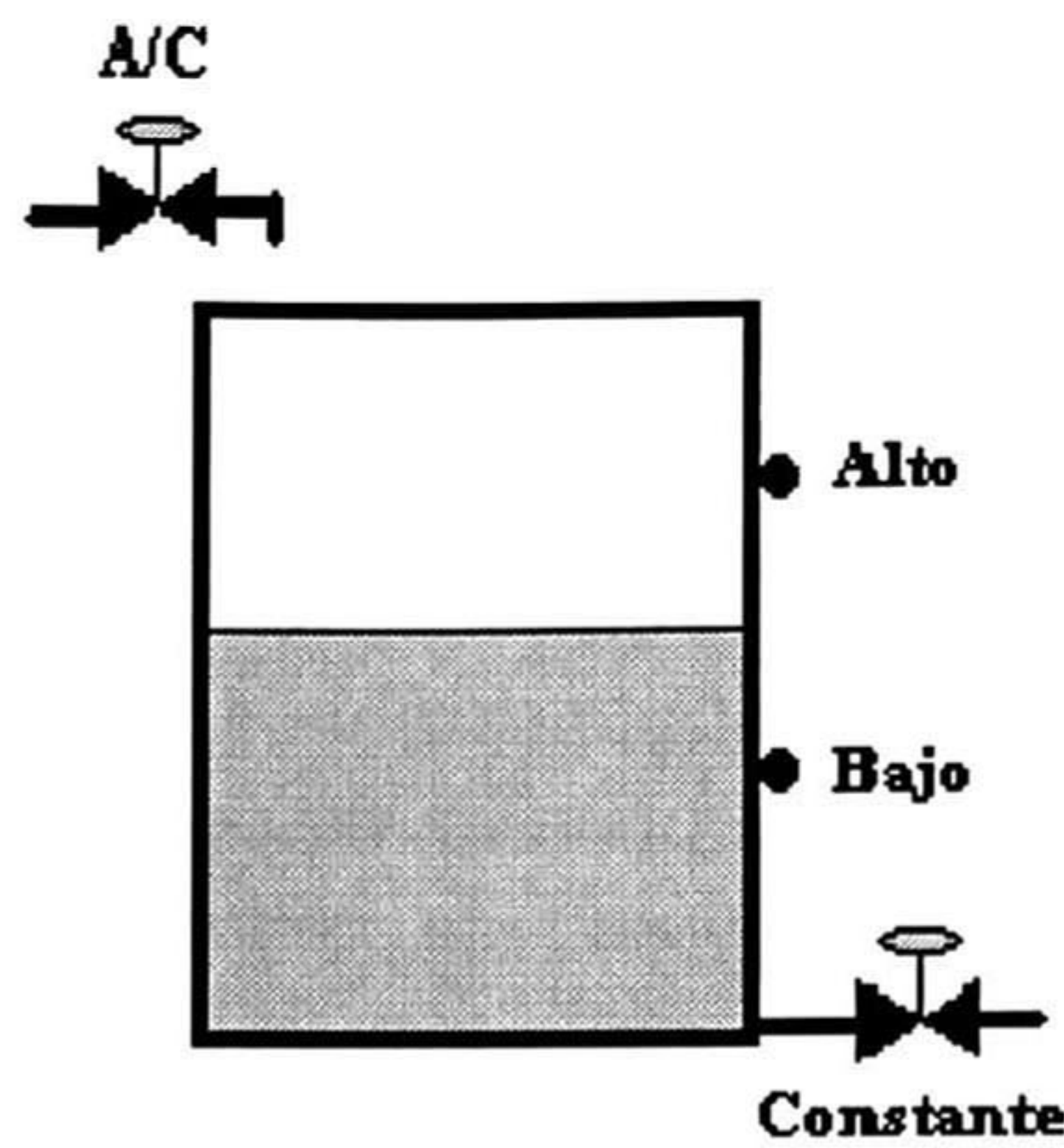


Figura 1.2: Sistema “*Nivel de líquido*” descrito como *SED*.

1.3 Técnicas para modelado de *SED*

Existen diferentes técnicas que describen o modelan el comportamiento de los *SED*, entre éstas se encuentran técnicas basadas en estados, actividades e híbridas.

En esta sección se explicarán dos técnicas: las basadas en estados y las técnicas híbridas, utilizando el ejemplo “*nivel de líquido*”.

Para el caso del sistema “*nivel de líquido*” los eventos pueden ser: *abrir válvula 1*, *cerrar válvula 1*, *nivel rebasando al sensor Bajo*, *nivel no rebasa al sensor Bajo*, *nivel rebasando al sensor Alto* y *nivel no rebasa al sensor Alto*. El estado del tanque se puede describir como el siguiente vector:

$$e = \begin{bmatrix} \text{nivel de líquido en tanque,} \\ \text{estado de válvula 1} \end{bmatrix}$$

El estado puede ser en un momento dado: $e_1 = [\text{vacío}, \text{cerrada}]^T$ que se interpreta como nivel de líquido en tanque *vacío* (cuando el agua no alcanza ninguno de los sensores) y válvula 1 *cerrada*. Al darse el evento *abrir válvula* el estado cambia a $e_2 = [\text{vacío}, \text{abierta}]^T$, es decir, nivel de líquido en tanque *vacío* y válvula 1 *abierta*.

El *comportamiento* del sistema es el siguiente: “El nivel de líquido en tanque es *vacío* y el estado de la válvula 1 es *cerrada*, cuando se da el evento *abrir válvula 1*, el nivel de líquido en el tanque subirá hasta rebasar el sensor **Bajo**, mientras el estado de la válvula 1 sea *abierta* continuará subiendo el nivel de líquido hasta tocar el sensor **Alto**, entonces se da el evento *cerrar válvula 1*, la válvula 1 cambia de estado a *cerrada* y el nivel de líquido en el tanque disminuirá de la siguiente forma: primero, no rebasará al sensor **Alto** y posteriormente no rebasará al sensor **Bajo**”. En este momento el nivel de líquido en el tanque es *vacío* y el estado de la válvula 1 es *cerrada*, que son las condiciones iniciales del sistema.

1.3.1 Técnicas basadas en estados

En la representación de estados se utilizan principalmente dos técnicas para la representación de *SED*: las *tablas de estados* y los *autómatas finitos*.

Representación tabular

En la representación *tabular* se tiene toda la información del sistema por medio de una tabla. Las columnas representan las condiciones sobre los eventos del sistema y los renglones representan los posibles estados, y se interpreta de la siguiente forma: “El primer renglón representa los estados del sistema, a partir del segundo renglón cada renglón contiene un conjunto de estados divididos de la siguiente manera: un estado dentro de un círculo que representa al estado actual o mantenido del sistema, y otros estados que representan al conjunto de estados alcanzables por la ocurrencia de un evento a partir del estado mantenido”. El aumento en el número de variables de entrada del sistema origina un crecimiento exponencial en el número de columnas de la tabla, por lo que esta forma de representación se ve limitada en su manejo; además el número de estados también se incrementa exponencialmente.

Por ejemplo, para el sistema “*nivel de líquido*” 1.2 la representación tabular es:

Vc	Va	Bc	Ba	Ac	Aa
● Vc	Va				
Vc	● Va		Ba		
		Bc	● Ba		Aa
Vc		● Bc	Ba		
				Ac	● Aa
		Bc		● Ac	Aa

donde los estados Vc , Va , Bc , Ba , Ac , Aa representan los diferentes niveles del tanque y el estado de la válvula 1 asociados de la siguiente manera:

- Vc nivel de líquido en tanque *vacío*, válvula 1 *cerrada*.
- Va nivel de líquido en tanque *vacío*, válvula 1 *abierta*.
- Bc nivel de líquido en tanque *bajo*, válvula 1 *cerrada*.
- Ba nivel de líquido en tanque *bajo*, válvula 1 *abierta*.
- Ac nivel de líquido en tanque *alto*, válvula 1 *cerrada*.
- Aa nivel de líquido en tanque *alto*, válvula 1 *abierta*.

Autómatas finitos

La representación de sistemas utilizando *autómatas finitos* mejora la representación tabular en lo que concierne a la claridad y compacidad; tiene las siguientes características:

1. Se puede saber si un estado se alcanza viendo si éste aparece en el grafo.
2. Es posible verificar propiedades tales como controlabilidad, estabilidad, etc.

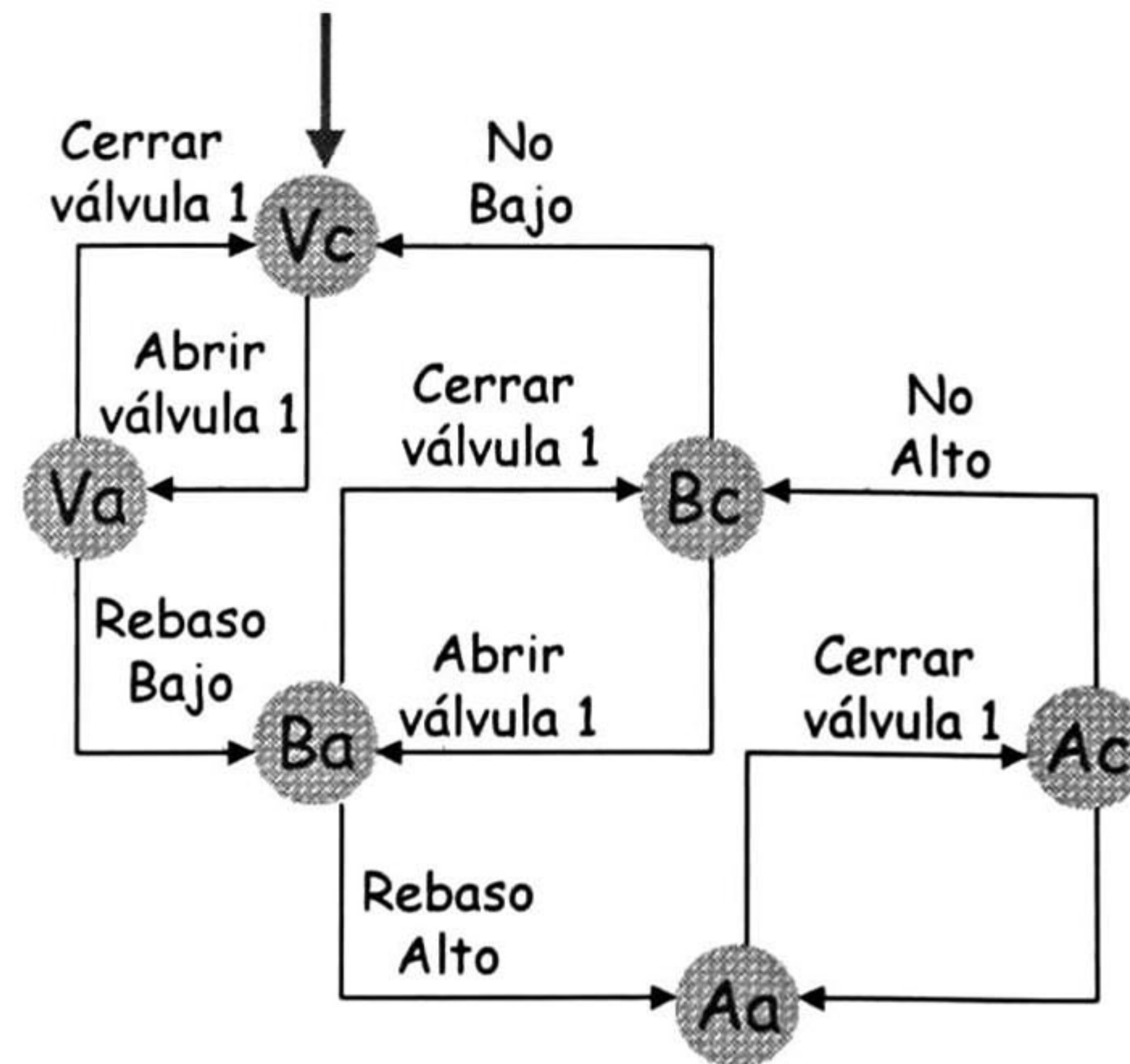


Figura 1.3: AF del sistema “Nivel de líquido”

El AF que representa al sistema de la figura 1.2 se muestra en la figura 1.3 en donde los estados y eventos corresponden a los mostrados en la representación tabular.

Desafortunadamente la explosión combinatoria persiste, sobre todo en los sistemas concurrentes ya que el aumento en el número de componentes en el sistema origina un crecimiento exponencial. Para mayor información sobre AF vea las referencias [1], [9] y [20].

1.3.2 Técnicas Híbridas

Entre los formalismos híbridos que existen actualmente para modelar, analizar, simular y controlar SED destacan las RP por sus características surgidas de su naturaleza gráfica y su soporte matemático: claridad en la descripción y facilidad para representar comportamientos complejos que incluyan secuencias, concurrencias, paralelismo, sincronizaciones e intercambio de información, entre otras.

La RP que representa al sistema de “nivel de líquido”, se muestra en la figura 1.4 en donde los estados y eventos corresponden a los mostrados en la figura.

Aquí no se detallan más a las RP porque se estudiarán en el capítulo 2.

Una RP es capaz de representar un sistema dotando su evolución de un significado. Así, una RP con una interpretación adecuada puede representar funcionalmente sistemas lógicos, un programa, etc.

1.4 Ventajas y desventajas de los formalismos de modelado

En muchos de los problemas prácticos de control, un SED consiste de un gran número de componentes que operan concurrentemente. Así, el número de estados en la representación de AF del sistema crece exponencialmente con el número de componentes paralelos, esto se puede considerar como una desventaja de los AF . Por otro lado las RP mantienen representaciones compactas aún en presencia de sistemas con gran número de componentes concurrentes.

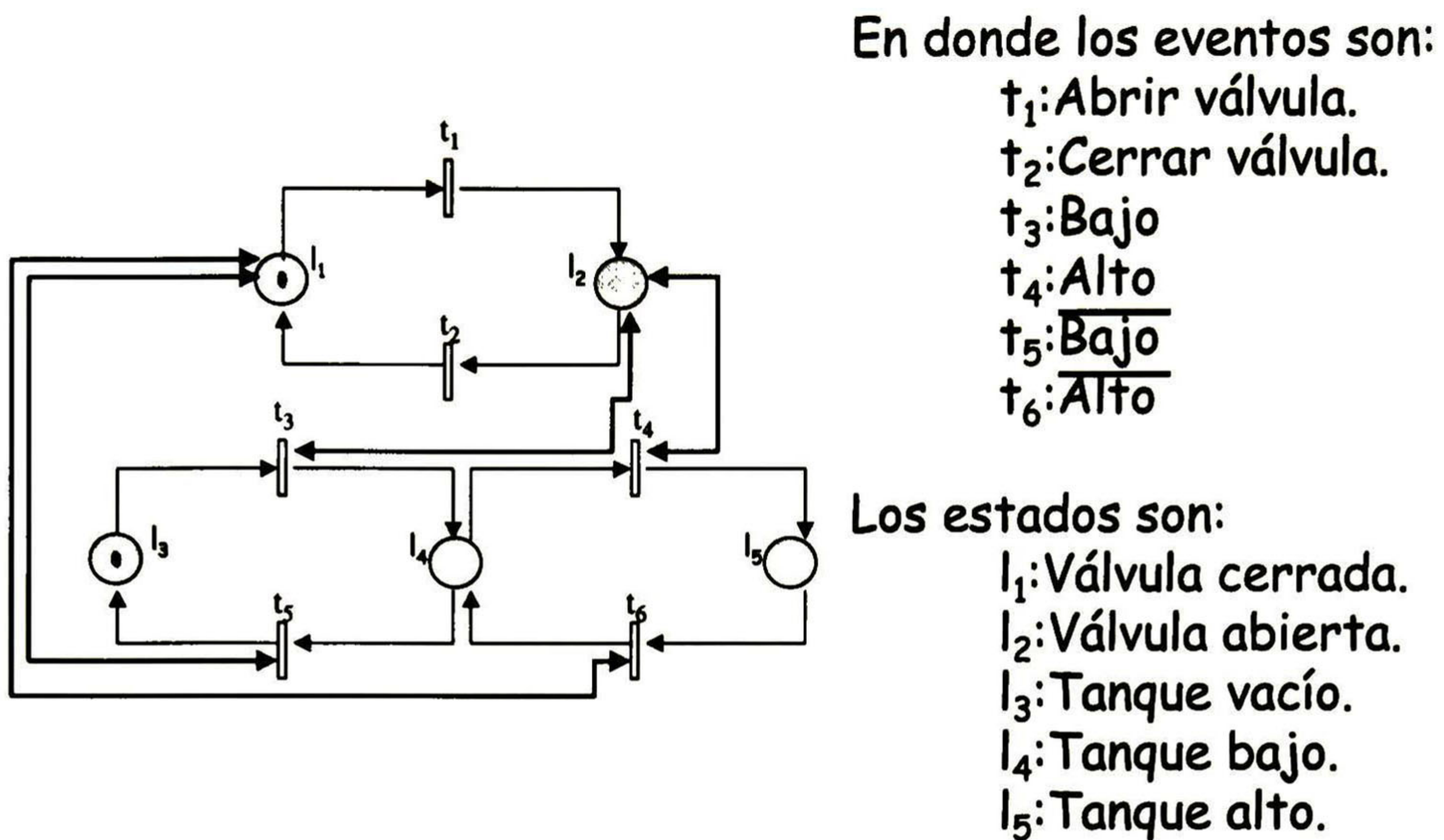


Figura 1.4: *RP* del sistema “Nivel de líquido”

La potencia de modelado también se ve limitada en los *AF*: sólo pueden representar lenguajes regulares. Mientras que una *RP* puede representar lenguajes regulares, algunos sensitivos al contexto y otros libres de contexto [9], [10].

En el campo de análisis las herramientas están equilibradas, ya que lo que se puede analizar con *AF* también se puede analizar (con la misma complejidad) con *RP*. Desafortunadamente el análisis con *RP* de lenguajes que no son regulares resulta, en muchos casos, en problemas NP e incluso, irresolubles (Referencias [9], [18]).

1.5 Estado del arte en control de *SED*

Generalmente un *SED* es representado como un lenguaje, en donde su comportamiento es especificado por una secuencia de eventos enumerados en un orden de ocurrencia, y se puede modificar el comportamiento del sistema a una especificación manipulando sus entradas.

Por tanto, el problema de control en *SED* se plantea en dos etapas:

1. Bajo ¿qué? condiciones el comportamiento de un *SED* puede ser restringido a una especificación dada; y
2. ¿Cómo? diseñar el agente que restrinja el comportamiento del *SED* a la especificación.

El primer caso se refiere al estudio de la controlabilidad y el segundo se refiere al diseño e implementación del controlador.

1.5.1 Controladores para *SED*

El propósito de un controlador para *SED* es establecer y mantener una sucesión de eventos deseados para lograr un comportamiento deseado. Existen diversos diseños para controladores en *SED*, como la teoría de control Supervisor [8] [11] [17] [19] [20], el control de procedimientos [?], los lugares de control [6], los controladores óptimos [5] [6], y los circuitos de comandos [6], entre otros.

La principal diferencia entre la aproximación de control Supervisor y otras aproximaciones es la siguiente. Los investigadores que trabajan en control supervisor se preocupan porque el controlador quede especificado en la misma herramienta matemática que se utiliza para describir al *SED*, mientras que para otras aproximaciones esto no es importante. Así, Wonham utiliza autómatas finitos [1] [9] y Giua redes de Petri [4] [10] [12] [18]. Sin embargo, Heloisa o Dicesare [5] [6] permiten que el controlador sea una máquina Turing, mientras que para describir el *SED* utilizan *RP*.

A continuación se menciona sucintamente la teoría de control supervisor tanto en *AF* como en *RP*, debido a que este trabajo se basa en esta teoría.

1.5.1.1 Teoría de control supervisor

El problema de restringir el comportamiento de un sistema a un comportamiento deseado ha sido estudiado bajo el punto de vista de control supervisor, en estos trabajos se tienen como hipótesis:

- La especificación y el sistema tienen la misma condición inicial.
- La especificación es estática, es decir no es variante en los eventos o en el tiempo.

El control supervisor está compuesto de:

- El *SED* a controlar.
- El agente que restringe al *SED* a un comportamiento deseado (supervisor).
- La secuencia de eventos ocurridos llamada γ .
- La secuencia de control β que es suministrada al *SED*.

La figura 1.5 muestra el esquema de control supervisor.

Ramadge y Wonham [17] han trabajado en el área de control de *SED* utilizando una teoría de control que ha sido desarrollada para estructuras de máquinas de estados (*autómatas finitos*), esta teoría sirve para diseñar agentes que permiten restringir el comportamiento de un *SED* a un lenguaje especificado previamente. Wonham y Ramadge han establecido las condiciones de existencia y la estrategia de diseño de dichos agentes (llamados *supervisores* utilizando *AF*). Giua [8] [11] presentó las condiciones de existencia de supervisores cuando el *SED* y supervisor están descritos en *RP*.

En seguida se presentan las filosofías de diseño de los supervisores de Wonham y Giua.

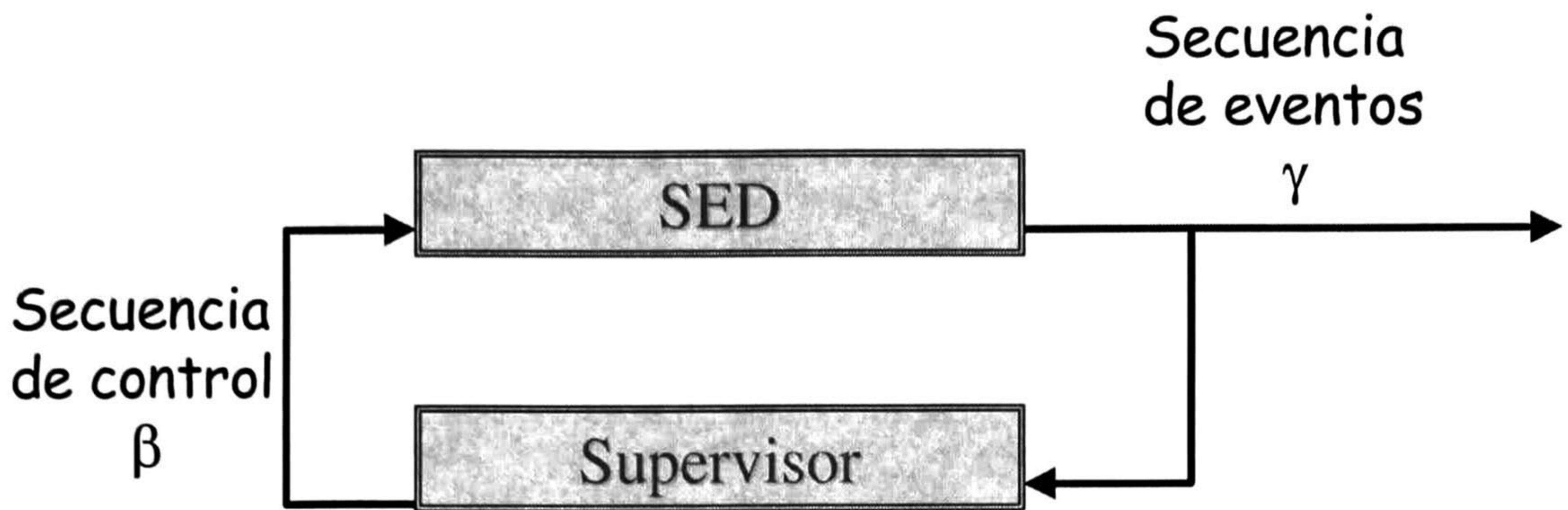


Figura 1.5: Esquema de un control supervisor.

1.5.1.2 Supervisor por Wonham [17], [19], [20] En la filosofía empleada por Wonham para máquinas de estado encontramos los siguientes pasos:

1. Conocer el lenguaje generado por la especificación (restricción de funcionamiento).
2. Probar que la especificación es controlable con respecto al lenguaje generado por el *SED*.
3. Diseñar un supervisor que sea mínimamente restrictivo.

1.5.1.3 Supervisor por Giua [8] El trabajo de Giua puede verse como una extensión a los trabajos de Ramadge y Wonham, ya que estudia algunos de los lenguajes sensitivos a contexto que son un superconjunto de los lenguajes regulares. Giua presenta las condiciones para la existencia de un supervisor en *RP*. Un supervisor en *RP* está basado en el diseño de un supervisor monolítico el cuál se realiza con los siguientes pasos:

1. Conocer el lenguaje generado por la especificación (restricción de funcionamiento). Para esto se utiliza un modelo en *RP*.
2. Probar que la especificación sea controlable con respecto a el lenguaje generado por el *SED*.
3. Diseñar una estructura burda para un supervisor (que es sintetizada por medio de la composición concurrente de sus diferentes módulos).
4. Se refina la estructura para evitar marcados indeseables.

El problema que presenta este supervisor se encuentra en el modelado, ya que se realiza un crecimiento exponencial de transiciones y con ello se vuelve difícil el análisis y el modelado.

1.6 Definición del problema

En los *Sistemas de Manufactura Flexible (SMF)*, los planes de proceso pueden cambiar continuamente y una nueva especificación puede darse sin importar el estado en que se encuentre el *SMF* en ese momento. Teniéndose que el lenguaje de la planta y de la especificación cambian y por tanto se deben rediseñar controladores supervisores continuamente. Para evitarse el problema de rediseñar continuamente un controlador, el presente trabajo plantea el estudio del *problema de seguimiento de modelo* en *SED*. La definición de este problema se da a continuación.

Definición 1.2 Sea R_m la especificación que se desea realizar y sea S_f el sistema físico o planta. El problema de *seguimiento de modelo* consiste en encontrar un controlador, que sin importar el estado actual del S_f , lleve la salida de S_f poco a poco a la salida actual de R_m , y una vez que ambas salidas sean iguales el comportamiento de salida de S_f sea igual al comportamiento de salida de R_m .

1.7 Objetivos y metas

El principal objetivo de la tesis es obtener condiciones suficientes para la existencia de un controlador que permita el seguimiento de modelo, así como proponer una metodología para el diseño de dicho controlador, usando en este caso *RP* interpretadas.

1.8 Conclusiones

En este capítulo se presentó un panorama sobre los sistemas de eventos discretos así como las herramientas que existen para modelarlos y analizarlos. Las técnicas de modelado mencionadas son: las *tablas de estados*, los *autómatas finitos* y las *redes de Petri*. Las ventajas y desventajas de estas técnicas de modelado se enfocan en dos puntos: *representación* y *potencia de modelado*, donde se menciona que la red de Petri ofrece las mismas ventajas que los autómatas finitos pero además tiene una representación más compacta y una mayor potencia de modelado. También se mencionan algunas metodologías para el diseño de controladores y es presentada la *teoría de control supervisor* ya que el *problema de seguimiento de modelo* se fundamenta en esta teoría. Finalmente, se define el *problema de seguimiento de modelo* como encontrar un controlador que sin importar el estado del sistema lo lleve poco a poco al comportamiento deseado y lo restrinja a la especificación.

Capítulo 2

Preliminares matemáticos

Resumen: Este capítulo presenta conceptos básicos relacionados con lenguajes, máquinas de estado finito, *RP* y *RPI*, que serán la base para estudiar los problemas de control en eventos discretos. También se muestran los principales resultados que han sido presentados en la *teoría de control supervisor*, ya sea utilizando *AF* o *RP*.

2.1 Introducción

Este capítulo presenta algunas definiciones y resultados que se tienen en AF , RP y RPI . También muestra cómo realizar control supervisor usando AF y RP . Todos estos son resultados ya clásicos. Para el estudio de autómatas son buenas referencias [1] [9]. El libro [18] es una buena lectura para cursos introductorios en RP mientras que [4] es un texto avanzado recomendable. Resúmenes buenos en RP se encuentran en los artículos [15] [10].

Para el estudio de control supervisor se encuentra el artículo clásico [17]; algunos trabajos más recientes se encuentran en [19] [20] [3] [?] [11] donde se estudia el problema de observabilidad y AF no deterministas.

Por otro lado, los trabajos [8] [6] abordan el problema de control supervisor en RP .

2.2 Lenguajes, cadenas y proyecciones

En la teoría de control supervisor los conceptos de lenguaje son necesarios, aquí se definen estos conceptos. El primer concepto que se tiene es el de *símbolo*, el cual es simplemente una etiqueta. Por ejemplo puede ser la letra a o cualquier otro carácter (incluyendo números, etc.).

Definición 2.1 *Un alfabeto Σ es un conjunto finito de símbolos*

Definición 2.2 *Una cadena o palabra es la yuxtaposición de símbolos que pertenecen a algún alfabeto.*

Por ejemplo, si $\Sigma = \{a, b, c\}$, se pueden tener las palabras $w_1 = aaa$, $w_2 = cbba$, $w_3 = bbbbbb\dots$

Definición 2.3 *La longitud de una cadena o palabra es simplemente la cantidad de símbolos que contenga. La longitud de w_i se representa como $|w_i|$.*

Por ejemplo $|w_1| = 3$ mientras que $|w_3| = |\mathbb{N}|$. También se puede tener la palabra de longitud cero (i.e. vacía o sin símbolos). La palabra vacía w_i será igual al símbolo ε , donde $\varepsilon \notin \Sigma$, ($w_i = \varepsilon$).

Definición 2.4 Σ^+ *es el conjunto de todas las cadenas o palabras formadas con símbolos de Σ . La longitud de las palabras debe ser mayor que cero (no se permiten las palabras vacías).*

Definición 2.5 $\Sigma^* := \{w\} \cup \Sigma^+$, *donde $w = \varepsilon$. Es decir es el conjunto de todas las cadenas o palabras formadas con símbolos de Σ incluyendo la cadena vacía.*

Definición 2.6 *Un lenguaje formal \mathcal{L} sobre Σ es cualquier subconjunto de Σ^* .*

2.2.1 Operaciones con cadenas y lenguajes

Definición 2.7 La concatenación de cadenas es una función que anexa a una cadena otra cadena, se define como $cat : \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$:

$$cat(w_1, w_2) = \begin{cases} w_1 & \text{si } w_2 = \varepsilon \text{ y } w_1 \in \Sigma^* \\ w_2 & \text{si } w_1 = \varepsilon \text{ y } w_2 \in \Sigma^* \\ w_1 w_2 & \text{si } w_1, w_2 \in \Sigma^+ \end{cases}$$

Definición 2.8 La **proyección natural** de un alfabeto Σ sobre otro alfabeto Σ_i es un mapeo que sirve para anular los elementos de Σ que no pertenecen a Σ_i ; se denota por Pr y se define como:

$$Pr : \Sigma^* \rightarrow \Sigma_i^*$$

y al aplicarla tenemos:

$$\begin{aligned} Pr(\varepsilon) &= \varepsilon \\ Pr(\sigma) &= \begin{cases} \varepsilon & \text{si } \sigma \notin \Sigma_i \\ \sigma & \text{si } \sigma \in \Sigma_i \end{cases} \\ Pr(s\sigma) &= Pr(s)Pr(\sigma) \text{ donde } s \in \Sigma^*, \sigma \in \Sigma \end{aligned}$$

Note que $Pr(s)$ es la concatenación de las proyecciones naturales sobre cada $\sigma \in s$.

Definición 2.9 La **proyección inversa** $Pr^{-1} : \Sigma_i^* \rightarrow 2^{\Sigma^*}$ de un alfabeto Σ_i sobre Σ es la imagen inversa de la proyección natural de Σ sobre Σ_i . Se denota como Pr^{-1} y se define como: $Pr^{-1}(\mathcal{L}) : \{s \in \Sigma^* | Pr(s) \in \mathcal{L}\}$, donde $\mathcal{L} \subseteq \Sigma_i^* \subseteq \Sigma^*$.

A continuación se presentan conceptos básicos relacionados con RP y RPI con el fin de estudiar la teoría de control supervisor desarrollada por Giua, y presentar la herramienta de modelado a utilizar para resolver el problema de seguimiento de modelo.

2.3 Redes de Petri

En esta sección se presentan los conceptos, propiedades básicas y lenguajes generados por las redes de Petri, así como una clase de red que asocia el significado físico del sistema a la estructura de la red, estas redes son las Redes de Petri Interpretadas.

2.3.1 Red de Petri ordinaria

Definición 2.10 Una **Red de Petri ordinaria** (RP) es la cuádrupla:

$$RP = (X, M, M_0, \Delta) \text{ donde:}$$

- X es el **grafo** $X = \langle L, T, E, S \rangle$, donde:

- $L = \{l_1, l_2, \dots, l_n\}$ es un conjunto de n elementos llamados **lugares**, los cuales se representan gráficamente mediante círculos,

- $T = \{t_1, t_2, \dots, t_m\}$ es el conjunto de m elementos llamados *transiciones*, los cuales se representan mediante barras o segmentos de línea,
- $E : L \times T \rightarrow \{0, 1\}$ es la *función de entrada* que representa los arcos que van de los lugares hacia las transiciones, la cual se define de la siguiente forma:

$$E(l_i, t_j) = \begin{cases} 1 & \text{si el arco de } l_i \text{ a } t_j \text{ existe} \\ 0 & \text{en caso contrario} \end{cases}$$

- $S : L \times T \rightarrow \{0, 1\}$ la *función de salida* que representa los arcos que van de las transiciones hacia los lugares, definida como sigue:

$$S(l_i, t_j) = \begin{cases} 1 & \text{si el arco de } t_j \text{ a } l_i \text{ existe} \\ 0 & \text{en caso contrario} \end{cases}$$

- M es una *función de marcado* definida como $M : L \rightarrow \mathbb{N} \cup \{0\}$ es la asignación de un número entero no negativo de elementos llamados *marcas* a cada lugar de X . Una *marca* se representa por medio de un punto en el interior de un lugar. La distribución de las marcas, o el *marcado*, representa el estado de la Red de Petri. Por notación, $M = M(L)$ y M será llamado simplemente el *marcado*.
- M_0 es un *marcado inicial*, el cual es una asignación arbitraria de marcas en el momento inicial.
- Δ son las siguientes *reglas de evolución* :
 - a) Regla de habilitación. Una transición $t_k \in T$ de X está habilitada si $\forall l_i \in L$ de X , $M(l_i) \geq E(l_i, t_k)$. Si una transición t_k está habilitada, entonces se puede disparar.
 - b) Regla de disparo. Si una transición habilitada t_j es disparada en un marcado M_k entonces se alcanza el nuevo marcado M_{k+1} , eliminando $E(l_i, t_j)$ marcas de sus lugares de entrada l_i y añadiendo $S(l_i, t_j)$ marcas a sus lugares de salida l_i .

Definición 2.11 Sean M_i y M_j dos marcados de una RP. Se dice que M_i y M_j son *marcados consecutivos*, $M_i \xrightarrow{t_k} M_j$, si M_j se alcanza disparando una transición t_k habilitada en M_i .

Definición 2.12 Una *secuencia disparable* aplicable a partir del marcado M_0 se representa por una secuencia de transiciones tal que el disparo de cada transición conduce a un marcado que habilita la transición siguiente de la secuencia.

Si $M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j} M_2 \xrightarrow{t_k} \dots \xrightarrow{t_r} M_q$, se dirá que la secuencia $\sigma = t_i t_j \dots t_r$ está habilitada y que es *disparable* a partir de M_0 . Y se denota como: $M_0 \xrightarrow{\sigma} M_q$

Definición 2.13 El conjunto de secuencias disparables en RP a partir de M_0 es el *lenguaje de disparos*: $L(RP, M_0^{RP}) = \left\{ \sigma \mid M_0 \xrightarrow{\sigma} M \right\}$

Definición 2.14 Un marcado M_k es *alcanzable* en RP a partir de un marcado inicial M_0 si y sólo si existe una secuencia $\sigma \in L(RP, M_0^{RP})$ tal que $M_0 \xrightarrow{\sigma} M_k$.

Definición 2.15 Al conjunto de todos los marcados alcanzables se le llama *conjunto de alcanzabilidad* y se denota como $\mathcal{R}(RP) = \left\{ M_k \mid M_0 \xrightarrow{\sigma} M_k \text{ y } \sigma \in L(RP, M_0^{RP}) \right\}$

Nota 2.1 Cada marcado de la red de Petri es un estado del sistema que modela, por lo que se usará marcado o estado indistintamente para referirse a la distribución de marcas en la red de Petri.

Definición 2.16 Un marcado M_x es *coalcanzable* a partir de un marcado M_k en la red de Petri RP si y sólo si existe una secuencia de disparos aplicable a partir de $M_k \in \mathcal{R}(RP)$ que llevan del estado M_k a un estado deseado o final, es decir, existen la secuencia σ_1 y σ_2 tales que: $M_0 \xrightarrow{\sigma_1} M_k \xrightarrow{\sigma_2} M_x$ estado final.

Definición 2.17 Sean $n = |L|$ el número de lugares en la red y $m = |T|$ el número de transiciones de una RP . La estructura del grafo X de la RP se puede representar por medio de dos matrices:

- Se denomina *matriz de entrada o incidencia previa* a la matriz:

$$C^- = [c_{i,j}^-]_{n \times m}$$

donde $c_{i,j}^- = E(l_i, t_j)$.

- Se denomina *matriz de salida o incidencia posterior* a la matriz:

$$C^+ = [c_{i,j}^+]_{n \times m}$$

donde cada elemento de la matriz $c_{i,j}^+ = S(l_i, t_j)$.

- Se denomina *matriz de incidencia* a la matriz:

$$C = C^+ - C^-$$

Definición 2.18 *Ecuación de estados de una RP* Sea N una RP y $M_k \in \mathcal{R}(N)$. Entonces al disparar una transición t_j habilitada en M_k se alcanza un nuevo marcado M_{k+1} el cual puede ser calculado mediante la siguiente ecuación:

$$M_{k+1} = M_k + C\vec{v}$$

donde \vec{v} es el vector de disparo que posee un 1 en la j -ésima posición, si la transición t_j se dispara y un 0 en caso contrario.

Características estructurales de las Redes de Petri ordinarias

En este trabajo será necesario determinar las características de *vivacidad* y *acotamiento* de una RP , por esta razón se mencionan en esta memoria. Existen otras características tales como *ciclicidad*, *conflictividad*, *exclusión mutua*. Un lector interesado puede consultar las referencias [4] [12] [18].

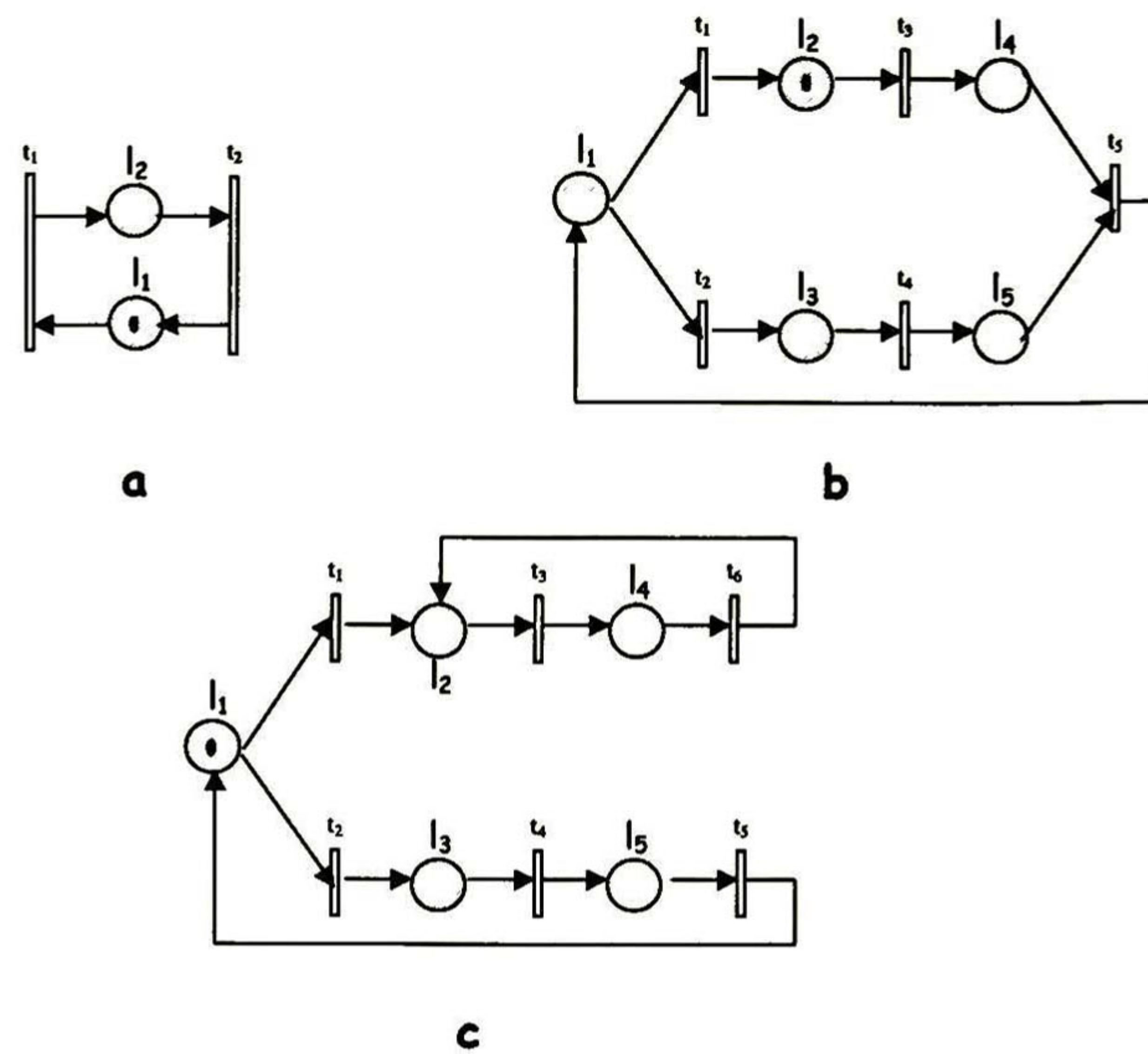


Figura 2.1: *RP* viva, no viva y parcialmente viva.

Definición 2.19 Una *transición* t es *viva* para un marcado inicial M_0 en una *RP* ssi existe una secuencia de disparos σ a partir de cualquier marcado M alcanzable desde M_0 , que comprenda a t :

$$\forall M \in \mathcal{R}(RP) \exists \sigma : M \xrightarrow{\sigma} M' \mid t \in \sigma$$

Definición 2.20 Una *RP* es *viva* para M_0 si y sólo si todas sus transiciones son vivas para M_0 .

La propiedad de **vivacidad** se utiliza para caracterizar el *bloqueo de un sistema*.

Una *RP* viva es la *RP* de la figura 2.1 a, mientras que una *RP* no viva es la *RP* de la figura 2.1 b.

Si una *RP* es viva entonces el sistema que modela no se bloquea debido a que todas sus transiciones pueden llegar a dispararse. El caso contrario no es cierto, porque existen redes que no son vivas y no se bloquean, estas redes se denominan *parcialmente vivas*. Un ejemplo de este tipo de redes la *RP* de la figura 2.1 c.

Definición 2.21 Una *RP* es *estructuralmente viva* si existe un marcado M_0 finito para el cual la *RP* es viva.

Definición 2.22 Un *lugar* l es *k-limitado* o *k-acotado* para M_0 ssi existe un número entero k tal que $M(l) \leq k$ para cualquier marcado $M \in \mathcal{R}(RP)$. Al menor entero k que cumple con la desigualdad anterior se le denomina *límite del lugar* l .

Definición 2.23 Una *RP* es *k-limitada* o *k-acotada* para M_0 ssi todos los lugares son *k-limitados* para M_0 :

$$\forall l \in L \text{ y } \forall M \in \mathcal{R}(RP), M(l) \leq k$$

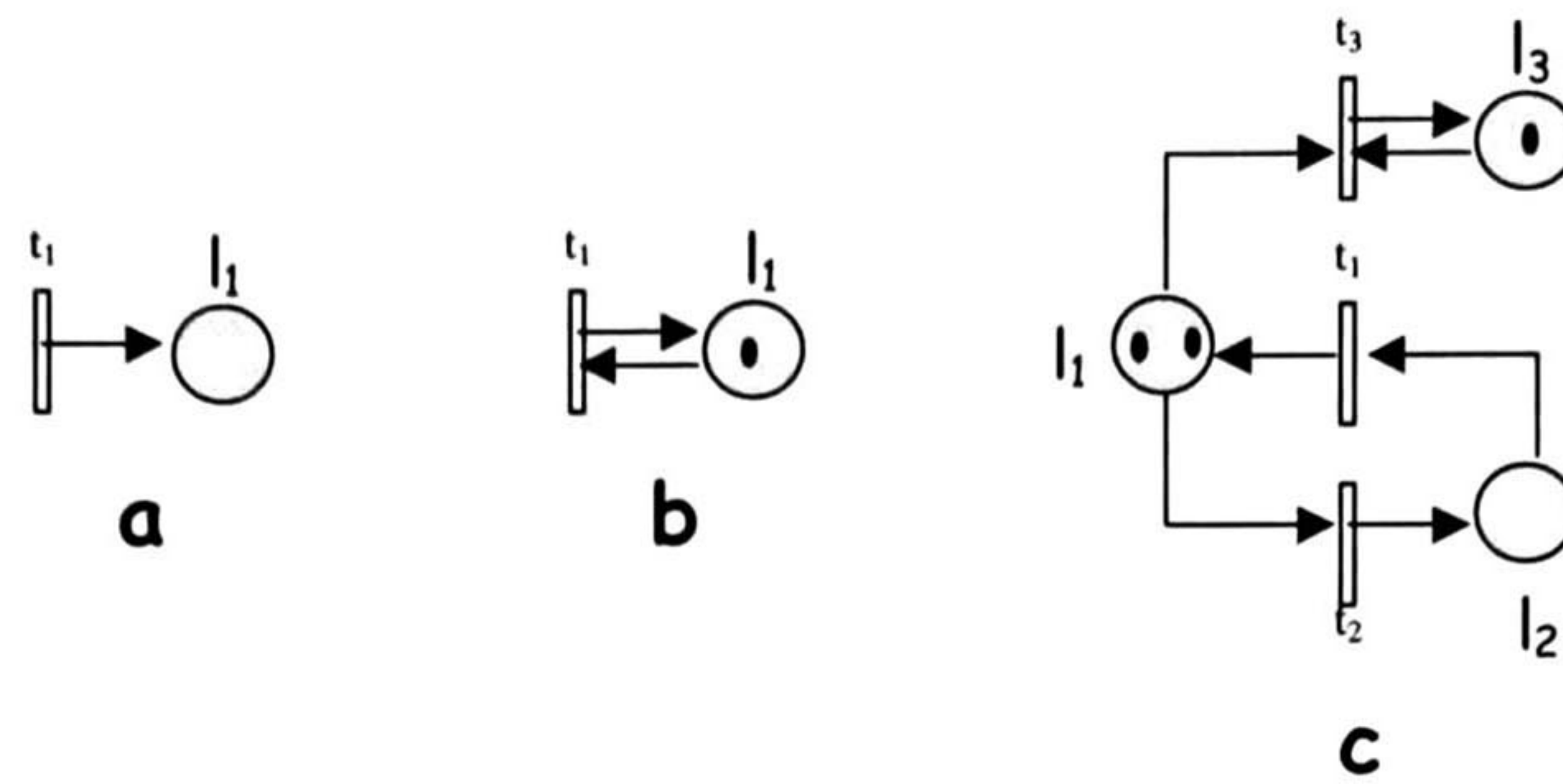


Figura 2.2: *RP no-limitada, 1-limitada y 2-limitada.*

La propiedad de limitación determina la finitud del número de estados del sistema representado por una *RP*. Las *RP* 2.2 b y c son redes con la propiedad de limitación, y la *RP* de la figura 2.2 a es una *RP no-acotada*.

Definición 2.24 Una *RP* es *estructuralmente limitada* si es limitada para cualquier marcado inicial finito.

La limitación estructural es una condición suficiente para la limitación.

Para el caso especial de una *RP* 1-limitada para M_0 surge la siguiente definición.

Definición 2.25 Una *RP* es *binaria* si todos sus lugares son 1-limitados para cualquier marcado $M \in \mathcal{R}(RP)$

Para resolver el *problema de seguimiento de modelo* en este trabajo se utiliza como formalismo de modelado las Redes de Petri Interpretadas, que son una extensión de las *RP* ordinarias, ya que al agregar las funciones de etiquetado (λ y φ) para relacionar estados con señales de sensores y transiciones con señales de actuadores, se tiene una herramienta que ofrece las ventajas de una *RP* ordinaria como herramienta matemática de modelado y que brinda la interpretación que necesitamos, la Red de Petri Interpretada.

A continuación se presenta la definición de las Redes de Petri Interpretadas y otras definiciones relacionadas con éstas.

2.3.2 Redes de Petri Interpretadas

Definición 2.26 Una *Red de Petri Interpretada* es la séxtupla:

$RPI = (N, \Sigma, \Phi, \lambda, \varphi_{RPI}, \Delta)$ donde:

- N es una *Red de Petri ordinaria*, donde $n = |L|$ el número de lugares en el grafo X y $m = |T|$ el número de transiciones del grafo X .
- Σ es un conjunto de s símbolos, llamado *alfabeto de entrada*, donde $s \leq m$
- Φ es un conjunto de p símbolos, llamado *alfabeto de salida*, donde $p \leq n$

- $\lambda : T \longrightarrow [\Sigma \cup \{\varepsilon\}]^m$ es una **función de etiquetado de las transiciones**, que indica cuando una transición puede ser controlada¹ $\forall t_i, t_j$ tal que $E(l_k, t_i) = 1$ y $E(l_k, t_j) = 1$ entonces si $\lambda(t_i) \in \Sigma$. $\lambda(t_i) \neq \lambda(t_j)$. Si $\lambda(t_i) = \varepsilon$ entonces la transición t_i no puede ser habilitada o inhabilitada. En este caso, se dice que t_i es una **transición incontrolable**, en otro caso es llamada **controlable**.
- $\varphi_{RPI} : \mathcal{R}(N) \longrightarrow [\Phi \cup \{\varepsilon\}]^n$ es una **función de etiquetado de los marcados**. En este trabajo sólo se abordará el caso de las RPI binarias, para las cuales $M(l_i) \in \{0, 1\}$ para todo $l_i \in L$, por lo cual la función φ_{RPI} se escribe como:

$$\varphi_{RPI}(M_k^{RPI}(l_i)) = \begin{cases} b_i & \text{si } M_k^{RPI}(l_i) = 1 \\ \varepsilon & \text{si } M_k^{RPI}(l_i) = 0 \end{cases} \quad \text{donde } b_i \in \Phi \cup \{\varepsilon\}$$

Si $\varphi_{RPI}(M_k^{RPI}(l_i)) = b_i \neq \varepsilon$, entonces l_i es llamado **lugar medible**, en caso contrario l_i es llamado **lugar no medible**. En este caso, el símbolo ε representa un evento de salida no medible de S_f .

- Δ son las siguientes **reglas de evolución** :

a) Regla de habilitación. Una transición $t_k \in T$ está **habilitada** si $\forall l_i \in L$, $M(l_i) \geq E(l_i, t_k)$.

b) Reglas de disparo.

- * Sea t_j una transición controlable habilitada en un mercado M_k^{RPI} . Si el evento de entrada $a_i = \lambda(t_j)$ está presente, entonces t_j debe dispararse,
- * Sea t_u una transición no controlable habilitada en un mercado M_k^{RPI} . Entonces t_u se puede disparar,
- * En ambos casos con el disparo de una transición se alcanza el mercado M_{k+1}^{RPI} , el cual puede ser calculado con la ecuación de estados de una RP.

La figura 2.3 muestra la RPI para el sistema de "Nivel de líquido" mostrado en el capítulo anterior. Esta RPI es binaria, en el cual las funciones λ y φ asocian los siguientes valores:

$$\begin{array}{ll} \lambda(t_1) = A & b_1 = \varepsilon \\ \lambda(t_2) = C & b_2 = \varepsilon \\ \lambda(t_3) = \varepsilon & b_3 = \varepsilon \\ \lambda(t_4) = \varepsilon & b_4 = \text{BAJO} \\ \lambda(t_5) = \varepsilon & b_5 = \text{ALTO} \\ \lambda(t_6) = \varepsilon & \end{array}$$

En el capítulo anterior se mencionó que para poder hacer el *seguimiento de modelo* es necesario que el sistema de manufactura sea *observable*, por lo cual es introducida la definición de *observabilidad* [2].

¹Note que $\varepsilon \notin \Sigma$

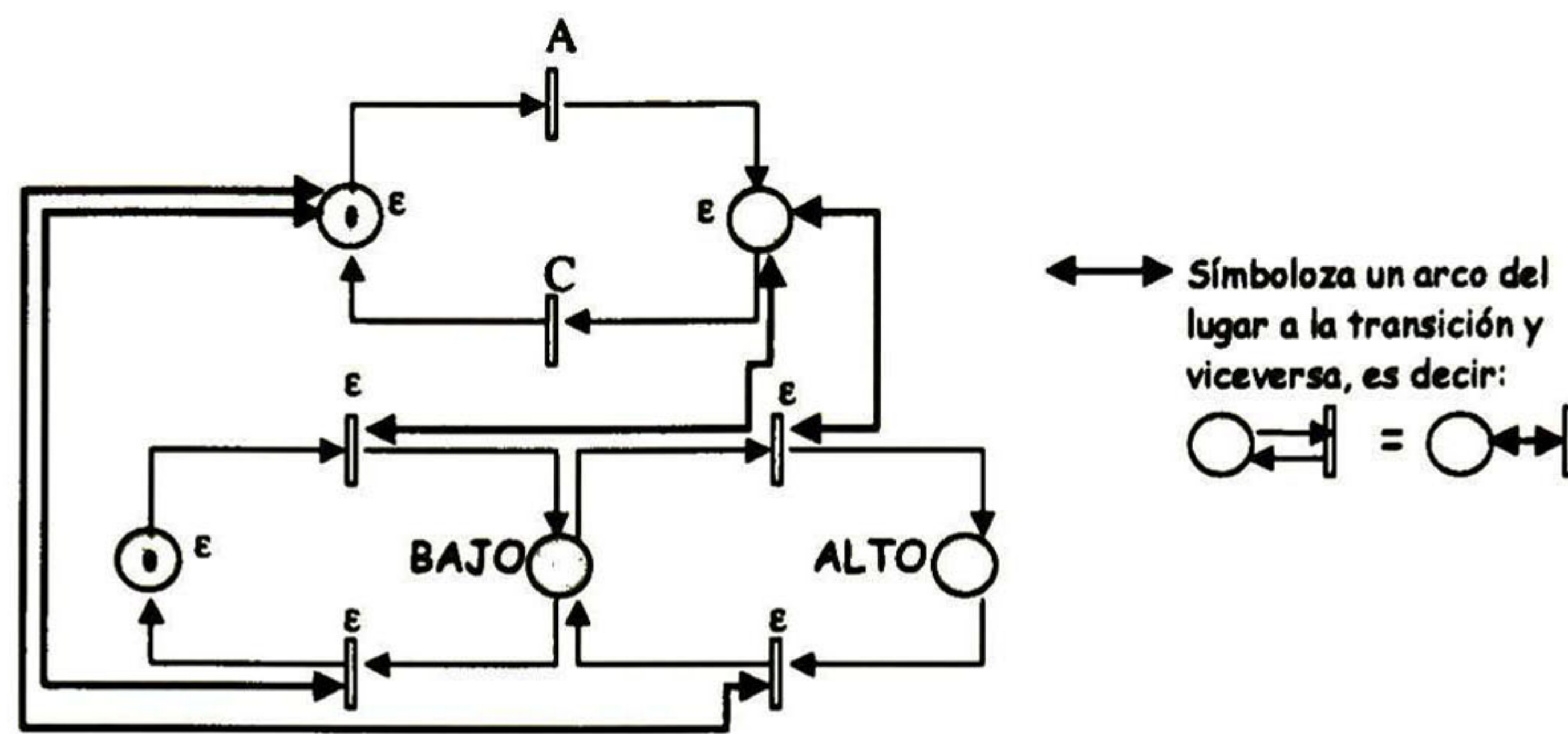


Figura 2.3: RPI del sistema "Nivel de líquido"

Definición 2.27 *Sistemas de eventos discretos observables.* Un SED modelado por medio de una RPI es observable si y sólo si dada la estructura de la RPI, $\forall M_0$ existe una función Ψ definida como

$$\Psi : \mathcal{L}_e \times \mathcal{L}_s \rightarrow \mathcal{R}(RPI)$$

Esto es, un sistema modelado por medio de una RPI es observable si y sólo si su estado actual (marcado actual) puede ser calculado a partir del conocimiento de su estructura, sus entradas y sus salidas. Si una palabra w es aceptada como una entrada, entonces alguna secuencia de transiciones t_i será disparada, donde $P(t_i)|_{transiciones\ controlables} = t_h t_k \dots t_m$ y $\lambda(t_h)\lambda(t_k)\dots\lambda(t_m) = w$ es una secuencia que preserva el orden de las transiciones controlables como los estados de la palabra w . Nótese que es posible alcanzar diferentes marcados con la misma entrada debido a que algunas transiciones incontrolables pueden dispararse en orden diferente.

La *controlabilidad* es la propiedad que indica si un sistema puede ser restringido a una especificación. La controlabilidad en RPI [14] se muestra a continuación.

Definición 2.28 Sea una RPI y \mathcal{M}_F el conjunto de estados finales deseados, y T_i es el conjunto de transiciones incontrolables, $t \in T_i \rightarrow \lambda(t) = \epsilon$ y M_x, M_y , etc son marcados alcanzables intermedios entre M_0 y M_f . La RPI es *controlable* ssi

1. $\forall M_f \in \mathcal{M}_F \exists \sigma = \sigma_a \sigma_b$ tal que $M_0 \xrightarrow{\sigma_a} M_x \xrightarrow{\sigma_b} M_f \quad M_x \in \mathcal{R}(RPI)$

y

2. $\forall M_x$ se satisface una de las dos condiciones siguientes:

- (a) $\nexists s \in T_i \mid M_x \xrightarrow{s}$

- (b) $\forall s \in T_i \mid M_x \xrightarrow{s} M_y$ y tomando M_y como M_0 , M_y cumple con 1.

Sin embargo, en el *seguimiento de modelo* se busca restringir el comportamiento del sistema a una especificación deseada en base al *lenguaje de salida* y esta definición nos muestra *controlabilidad en el lenguaje de entrada*. Por lo anterior es necesario hacer una extensión a esta definición; en el próximo capítulo se define la controlabilidad del lenguaje de salida

Con el fin de mostrar las diferencias existentes entre la *teoría de control supervisor* y el *seguimiento de modelo* a continuación se presentan a detalle los trabajos de Wonham y Giua.

2.4 Teoría de control supervisor

La teoría de control supervisor desarrollada por Wonham tiene como herramientas de especificación los autómatas finitos. Posteriormente Giua extendió estos resultados a RP . A continuación se presentan los trabajos de Wonham y Giua.

2.4.1 Supervisor en autómatas finitos

En la *teoría de control supervisor* de Wonham el SED es modelado como un lenguaje regular el cual se representa como un AF . El alfabeto sobre el cual se define este lenguaje se particiona en dos subconjuntos: uno de eventos *controlables* y otro de eventos *incontrolables*. De manera que el controlador que se diseñe sólo puede deshabilitar los eventos controlables.

La definición que emplea Wonham sobre autómatas finitos es la siguiente:

Definición 2.29 *Un autómata finito es una quintupla:*

$$A = (Y, \Sigma, \eta, y_0, Y_m)$$

formada por tres conjuntos finitos no vacíos Y, Σ, Y_m , una función $\eta : Y \times \Sigma \longrightarrow Y$ que define el comportamiento del autómata, y un elemento y_0 del conjunto Y . Los elementos de A reciben los siguientes nombres:

- Y : *Conjunto de estados.*
- Σ : *Conjunto de símbolos de entrada (alfabeto de entrada).*
- Y_m : *Conjunto de estados marcados ($Y_m \subseteq Y$).*
- η : *función de transición de estados.*
- y_0 : *estado inicial.*

La notación $\eta(y, w)!$, representa la extensión de la función $\eta : Y \times \Sigma \longrightarrow Y$ a $\eta : Y \times \Sigma^* \longrightarrow Y$ por las reglas:

- $\eta(y, \varepsilon) = y$
- $\eta(y, sw) = \eta(\eta(y, s), w)$

con $s, w \in \Sigma^*$

Los conceptos presentados por Wonham son los siguientes.

Definición 2.30 *El conjunto de palabras $s \in \Sigma^*$ tal que $\eta(y_0, s)!$ es el lenguaje a lazo cerrado de A , denotado por $\mathcal{L}(A)$.*

Definición 2.31 *El subconjunto de palabras $s \in \mathcal{L}(A)$ tal que $\eta(y_0, s) \in Y_m$ es el lenguaje marcado de A , denotado por $\mathcal{L}_m(A)$.*

Definición 2.32 Sea A un SED con $\Sigma = \Sigma_c \cup \Sigma_u$ donde Σ_c es el conjunto de eventos controlables y Σ_u es el conjunto de eventos incontrolables. Cada patrón de control es un subconjunto de eventos. El conjunto de todos los patrones de control se define como:

$$\Gamma = \{\gamma \in 2^\Sigma \mid \gamma \supseteq \Sigma_u\}$$

Un **supervisor** para A es cualquier mapeo $V : \mathcal{L}(A) \rightarrow \Gamma$. El par (A, V) se denota como V/A para indicar que A está bajo la supervisión de V . El comportamiento de lazo cerrado de V/A es el lenguaje $\mathcal{L}(V/A) \subseteq \mathcal{L}(A)$ descrito como sigue:

1. $\varepsilon \in \mathcal{L}(V/A)$.
2. Si $s \in \mathcal{L}(V/A)$, $\sigma \in V(s)$, y $s\sigma \in \mathcal{L}(A)$ entonces $s\sigma \in \mathcal{L}(V/A)$.
3. No inician otras cadenas en $\mathcal{L}(V/A)$.

Definición 2.33 El lenguaje marcado de V/A es $\mathcal{L}_m(V/A) = \mathcal{L}(V/A) \cap \mathcal{L}_m(A)$.

Definición 2.34 Se dice que V es **no-bloqueante** (para A) si $\overline{\mathcal{L}_m(V/A)} = \mathcal{L}(V/A)$

Para diseñar un supervisor se debe cumplir que el SED se pueda restringir a la especificación deseada. La propiedad que define esta característica se denomina controlabilidad y se presenta a continuación.

Definición 2.35 Sea G un SED con $\Sigma = \Sigma_c \cup \Sigma_u$. Para $K \subseteq \Sigma^*$ tenemos que K es **controlable** con respecto a $L(G)$ si $\overline{K}\Sigma_u \cap L(G) \subseteq \overline{K}$ donde \overline{K} es la cerradura de prefijos.

A continuación se presentan las condiciones necesarias y suficientes para la existencia de un Supervisor en AF

Teorema 1 Sea $K \subseteq L_m(A)$, $K \neq \emptyset$. Existe un control supervisor no-bloqueante V para A tal que $L_m(V/A) = K$ si y sólo si:

1. K es controlable con respecto a A , y
2. K es $L_m(A)$ cerrado.

La primer condición del teorema establece que A puede restringir su comportamiento con respecto a K , y la segunda condición indica que K siempre llega a estados marcados en A , por lo que no existe bloqueo. El diseño del supervisor de Wonham se basa en los siguientes puntos:

1. Diseñar un autómata finito (candidato a supervisor) que satisfaga la especificación.
2. Probar que la especificación es controlable con respecto a el lenguaje generado por el SED .
3. Diseñar un supervisor que sea mínimamente restrictivo.

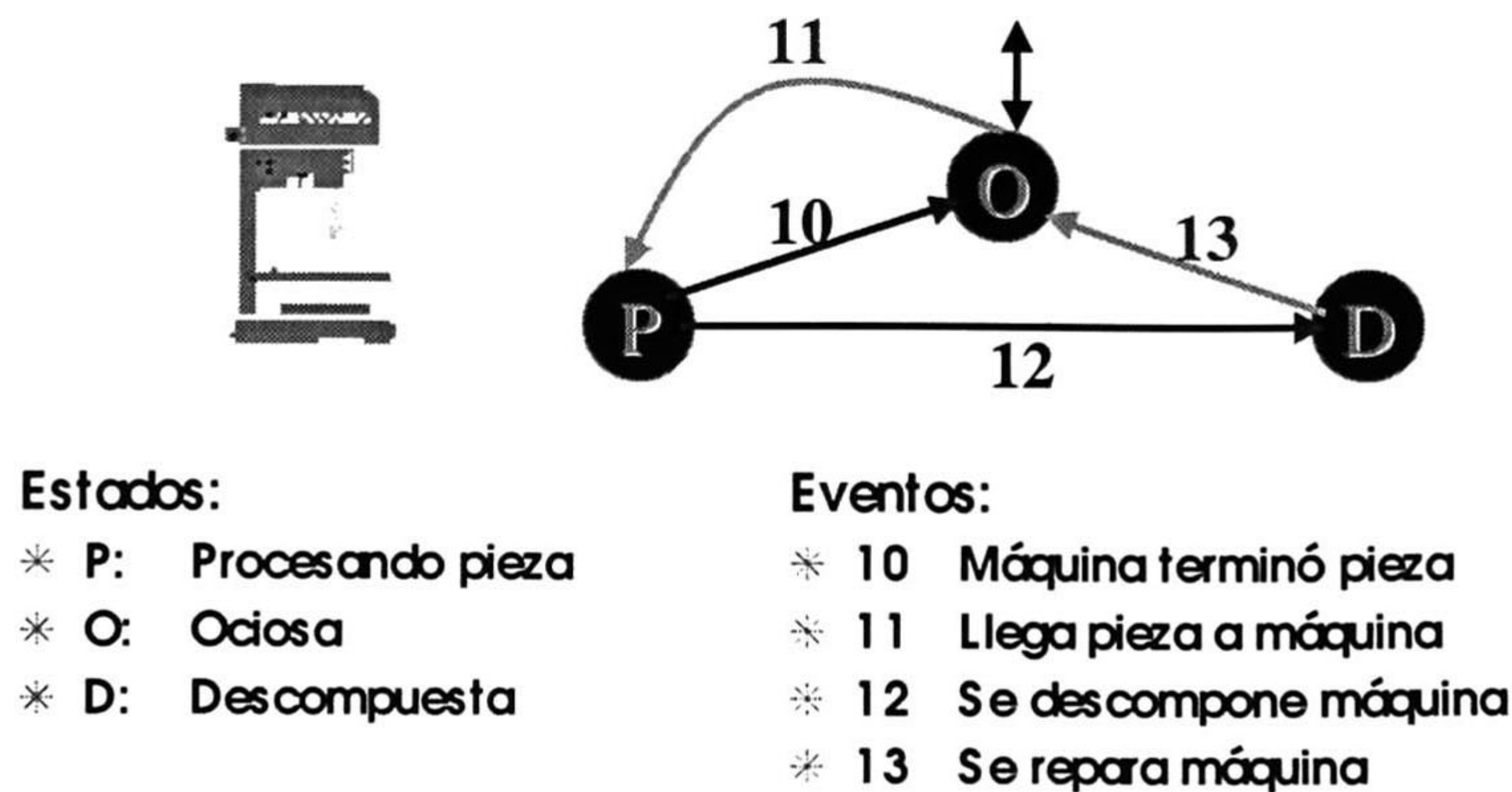


Figura 2.4: AF de máquina.

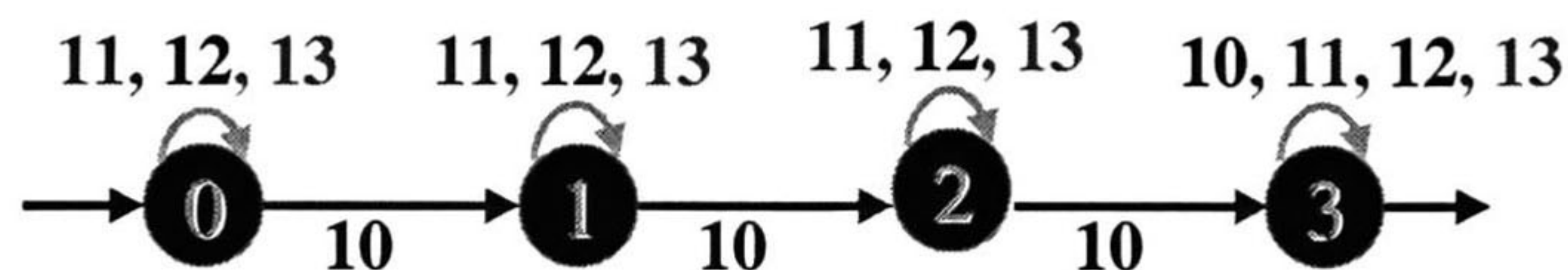


Figura 2.5: AF de la especificación.

Ejemplo 2.1 Considere el siguiente funcionamiento. Una máquina tiene tres estados: procesando pieza, ociosa y descompuesta; el modelo de esta máquina es representado mediante el autómata finito de la figura 2.4. En donde los eventos 10 (máquina terminó pieza) y 12 (se descompone máquina) son incontrolables y los eventos 11 (llega pieza a máquina) y 13 (se compone máquina) son controlables. La especificación deseada es: “elaborar por lo menos tres piezas” la cual es mostrada en la figura 2.5 constituye el supervisor; se prueba controlabilidad y se hace la composición² del sistema y del supervisor; la figura 2.6 muestra el sistema bajo supervisión.

Las características de supervisor diseñado por Wonham son:

- Al diseñar como supervisor un autómata finito, este supervisor es capaz de generar sólo lenguajes regulares.
- Debido a que los SED consisten de un gran número de componentes que operan concurrentemente, el número de estados crece exponencialmente y con ello la complejidad para realizar el controlador.
- Si la especificación cambia un nuevo supervisor debe calcularse.

2.4.2 Supervisor en redes de Petri

Giua hace una extensión a los trabajos de Ramadage y Wonham, ya que estudia algunos lenguajes sensitivos de contexto (un superconjunto de los lenguajes regulares). Para modelar

²La composición del sistema y del supervisor se forma haciendo el producto sincrónico total de éstos.

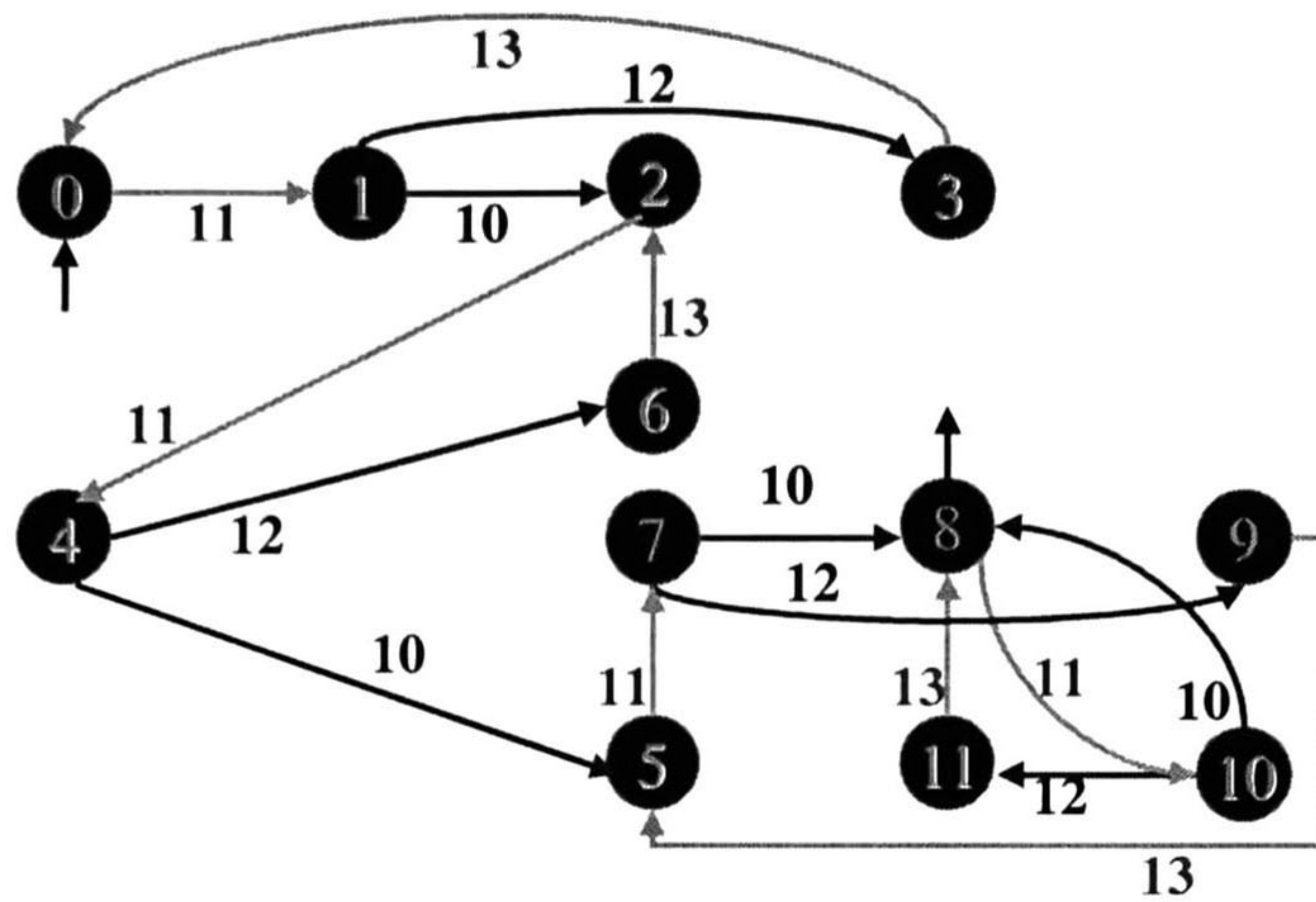


Figura 2.6: AF de la intersección del supervisor y el sistema.

el SED del sistema a controlar y el controlador, Giua usa como formalismo las Redes de Petri.

Para poder analizar el trabajo de Giua primero se presentan los lenguajes de las Redes de Petri, posteriormente las condiciones de existencia del controlador y por último el diseño de controladores usando esta metodología.

2.4.2.1 Lenguajes de las Redes de Petri

Una RP etiquetada (o una RP generadora) es una tripleta $G = (RP, l, M_f)$ donde:

- RP es una RP ordinaria;
- $l : T \rightarrow \Sigma \cup \{\varepsilon\}$ es una función de etiquetación que asigna a cada transición una etiqueta del alfabeto de eventos o asigna el símbolo ε como etiqueta;
- M_f es un conjunto finito de marcados finales.

2.4.2.2 Etiquetación

Existen tres tipos de funciones de etiquetación:

1. En una RP de *etiquetación libre* todas las transiciones tienen etiquetas distintas y ninguna se etiqueta con el símbolo de vacío ε , es decir:

$$(\forall t, t' \in T) [t \neq t' \rightarrow l(t) \neq l(t') \neq \varepsilon]$$

2. En una RP de *etiquetación libre de ε* ninguna transición se etiqueta con el símbolo de vacío ε .
3. En una RP de *etiquetación arbitraria* no existe restricción sobre l .

2.4.2.3 Tipos de lenguajes de la RP

Existen cuatro tipos de lenguajes asociados al SED (que se denominará G). Dependiendo de las diferentes nociones de las palabras terminales se tiene lo siguiente.

1. El *tipo L* o *lenguaje terminal* se define como el conjunto de palabras generadas por secuencias de disparos que alcanzan un marcado final³

$$L_L(G) = \left\{ l(\sigma) \mid M_0 \xrightarrow{\sigma} M_f \in \mathcal{M}_F \right\}$$

2. El *tipo G* o *lenguaje de cobertura* se define como el conjunto de palabras generadas por secuencias de disparos que alcanzan un marcado M que cubre un marcado final

$$L_G(G) = \left\{ l(\sigma) \mid M_0 \xrightarrow{\sigma} M \geq M_f \in \mathcal{M}_F \right\}$$

3. El *tipo D* o *lenguaje de bloqueo* se define como el conjunto de palabras generadas por secuencias de disparos, que alcanzan un marcado en el cual ninguna transición se habilita

$$L_D(G) = \left\{ l(\sigma) \mid \left(M_0 \xrightarrow{\sigma} M \right) \wedge \left(\nexists t \in T \mid M \xrightarrow{t} \right) \right\}$$

4. El *tipo P* o *lenguaje de prefijo* (lenguaje cerrado en teoría de Control Supervisor) se define como el conjunto de palabras generadas por cualquier secuencia de disparos

$$L_P(G) = \left\{ l(\sigma) \mid M_0 \xrightarrow{\sigma} \right\}$$

Dependiendo de la función de etiquetado, combinada con las clases de lenguajes previamente mencionadas, en las RP se tienen hasta doce lenguajes, los cuales se mencionan a continuación.

1. \mathcal{L}^f (respec. $\mathcal{G}^f, \mathcal{D}^f, \mathcal{P}^f$) denota la clase de lenguajes terminales (respec. de cobertura, de bloqueo, de prefijo) generados por una RP generadora de *libre etiquetación*.
2. \mathcal{L} (respec. $\mathcal{G}, \mathcal{D}, \mathcal{P}$) denota la clase de lenguajes terminales (respec. de cobertura, de bloqueo, de prefijo) generados por una RP generadora de *etiquetación libre de ϵ* .
3. \mathcal{L}^ϵ (respec. $\mathcal{G}^\epsilon, \mathcal{D}^\epsilon, \mathcal{P}^\epsilon$) denota la clase de lenguajes terminales (respec. de cobertura, de bloqueado, de prefijo) generados por una RP generadora de *etiquetación arbitraria*.

En la figura 2.7 se comparan los lenguajes que genera una RP con los que generan otros autómatas.

A continuación se presentan las condiciones necesarias y suficientes para la existencia de un Supervisor en RP [8].

³Un marcado final en RP es un marcado alcanzable desde el marcado inicial, es silimar a los estados maracdos en AF .

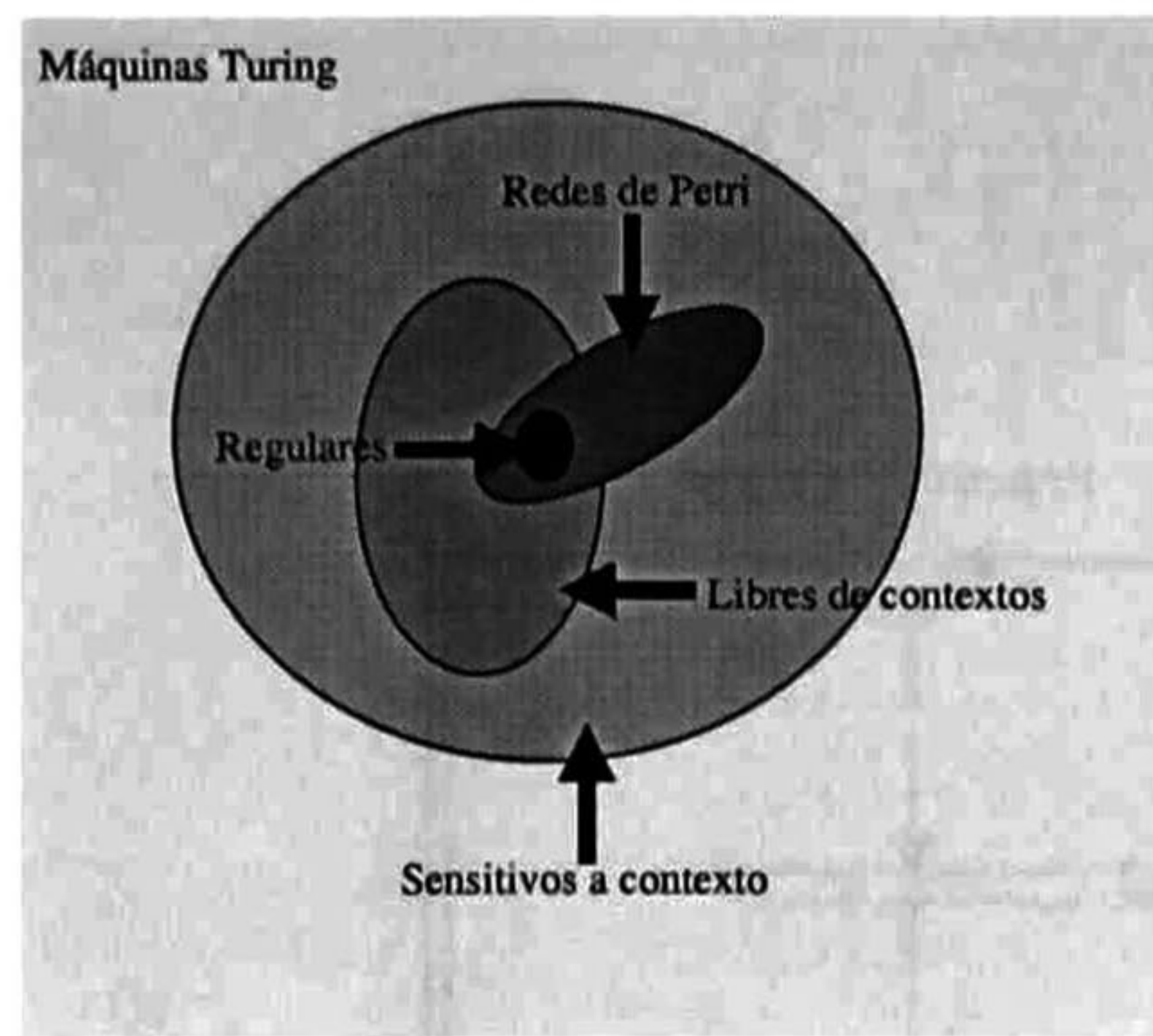


Figura 2.7: Tipos de lenguajes

2.4.2.4 Condiciones de existencia del supervisor

Teorema 2 Sea GRP una RP no bloqueada y $\mathcal{L} \subseteq \mathcal{L}(GRP)$ un lenguaje no vacío. Existe un Supervisor en RP , S tal que $\mathcal{L}(S/GRP) = \mathcal{L}$ si y solo si \mathcal{L} es un Lenguaje en RP tipo- P determinístico controlable, i.e., $\mathcal{L} \in P_d \cap \mathcal{C}(GRP)$.

Este teorema establece la existencia del supervisor S para GRP si el comportamiento a lazo cerrado de GRP bajo la supervisión de S es controlable.

Teorema 3 Sea GRP una RP no bloqueada y $\mathcal{L} \subseteq \mathcal{L}_m(GRP)$ un lenguaje no vacío. Existe un Supervisor en RP , S tal que $\mathcal{L}_m(S/GRP) = \mathcal{L}$ si y solo si $\mathcal{L} \in \mathcal{L}_{DP} \cap \mathcal{C}(GRP)$ y \mathcal{L} es $\mathcal{L}_m(GRP)$ – cerrado⁴.

Este teorema indica la existencia del supervisor S para GRP si llega a los estados finales de GRP .

2.4.2.5 Diseño del supervisor

El diseño del supervisor de Giua se basa en las siguientes ideas:

- Modelar las especificaciones y el sistema con RP .
- Realizar la composición concurrente del sistema con las especificaciones.
- Refinar la estructura para evitar marcados indeseables.

Ejemplos de supervisores

Ejemplo 2.2 Considere el siguiente enunciado. Una máquina que tiene tres estados: procesando pieza, ociosa y descompuesta; el modelo de esta máquina es representado mediante la red de Petri de la figura 2.8. En donde los eventos 10 (terminó pieza) y 12 (descompone) son incontrolables y los eventos 11 (llega pieza) y 13 (componen) son controlables. La especificación deseada es: “elaborar a lo más tres piezas” (vea la figura 2.9). El comportamiento del sistema bajo supervisión se muestra en la figura 2.10.

⁴La notación $\mathcal{L} \in \mathcal{L}_{DP}$ indica que \mathcal{L} es un lenguaje de bloqueo y prefijo.

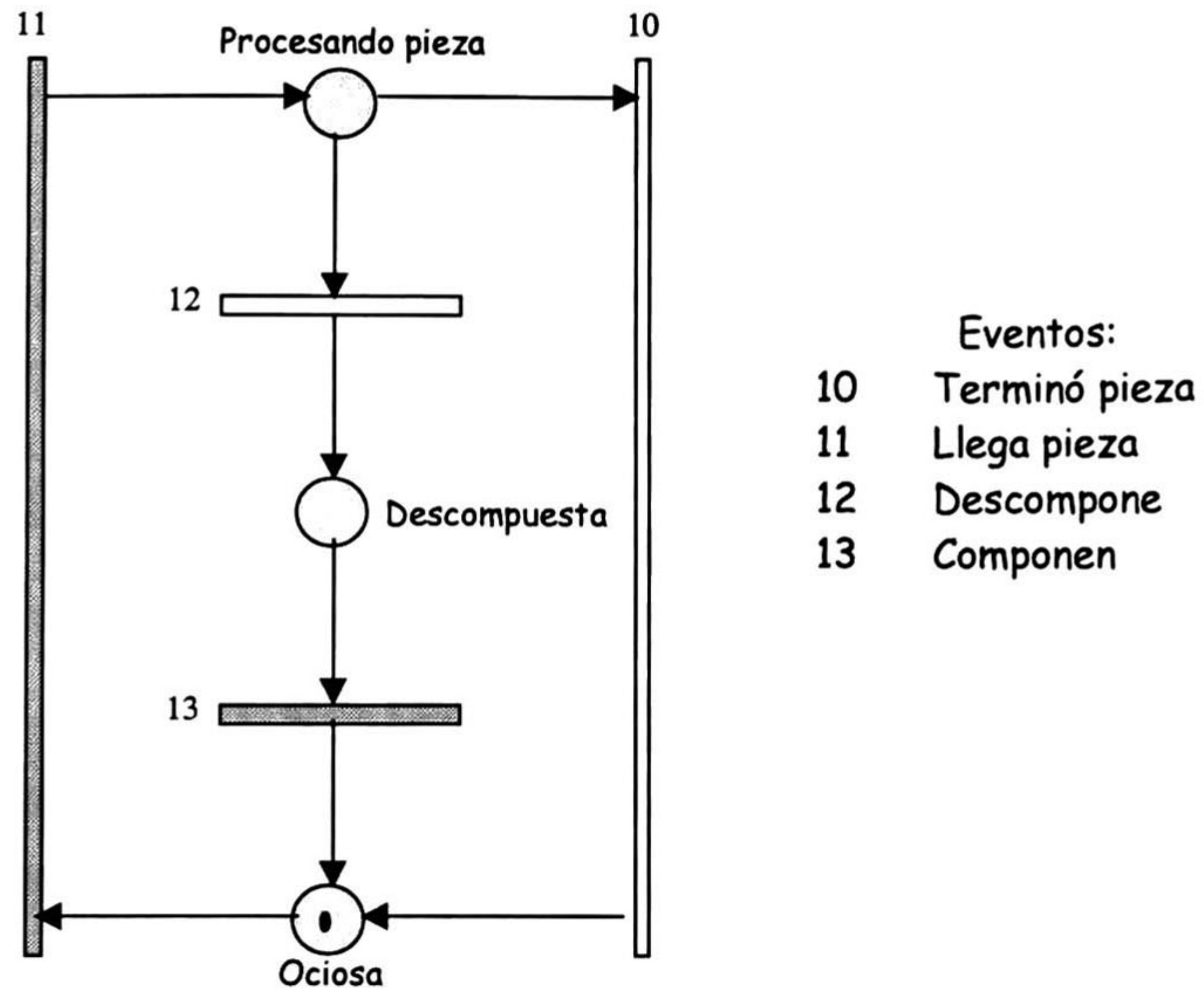


Figura 2.8: *RP* de máquina.

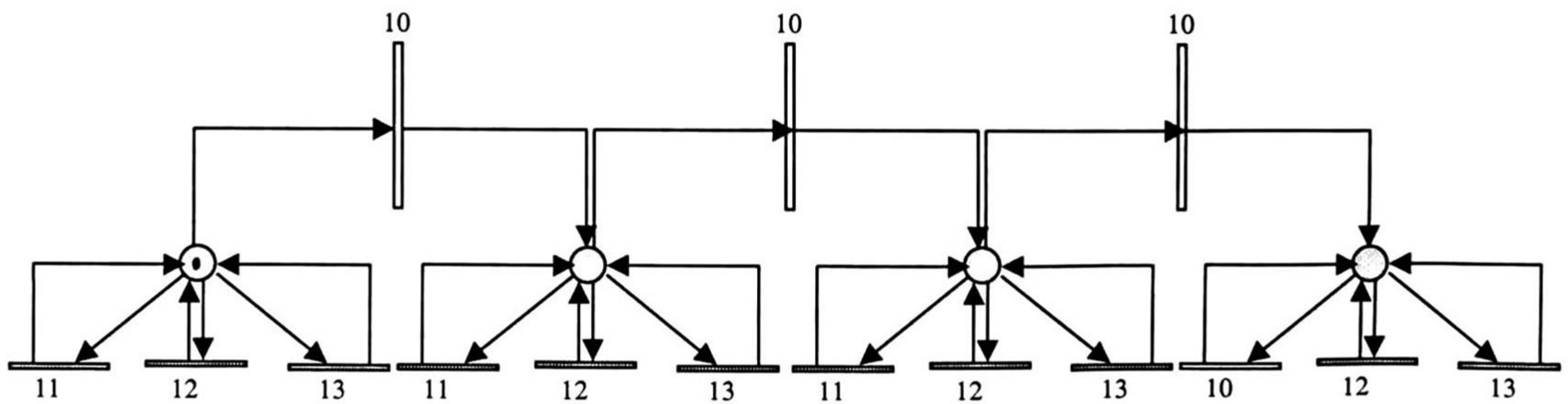


Figura 2.9: Especificación.

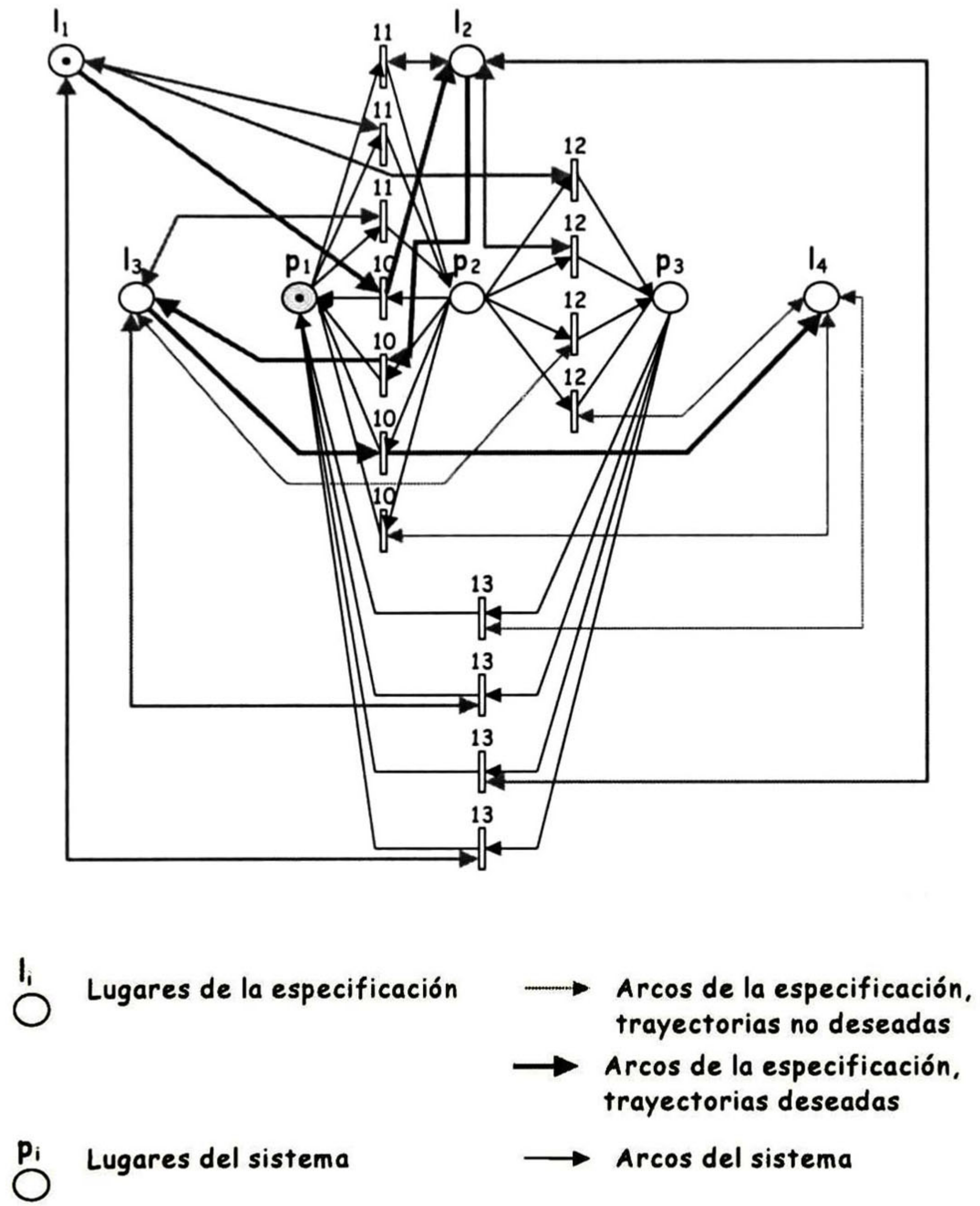


Figura 2.10: Comportamiento del sistema bajo supervisión.

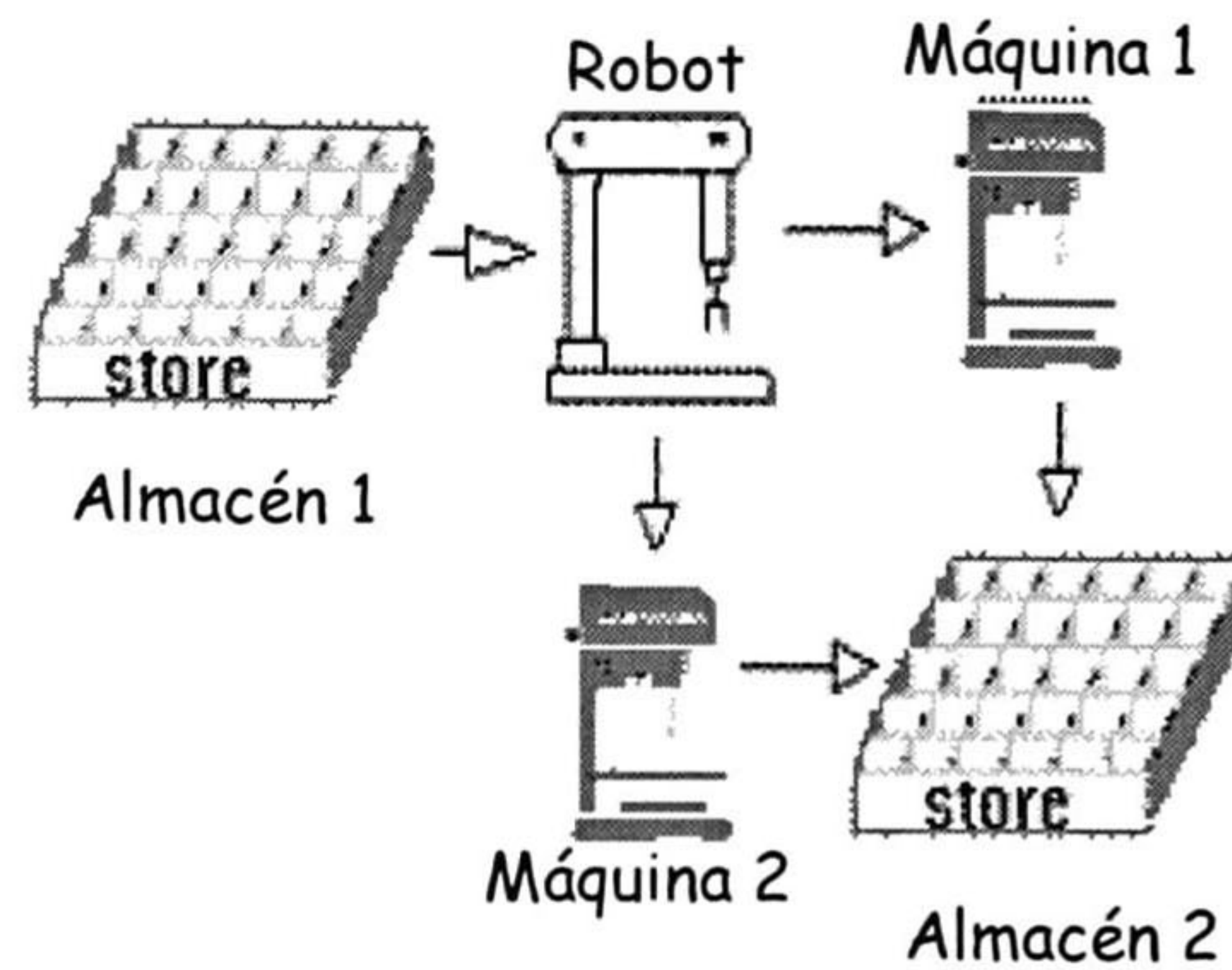


Figura 2.11: Sistema de Manufactura.

Ejemplo 2.3 El sistema de manufactura mostrado en la figura 2.11 consiste de dos máquinas, dos almacenes y un robot. El robot retira una pieza del almacén 1 y lo deposita ya sea en la máquina 1 o en la máquina 2. Una vez que la pieza ha sido procesada, pasa de la máquina al almacén 2 (vea la figura 2.12).

Suponga que la especificación es:

“si hay pieza en almacén 1, entonces el robot toma una pieza del almacén 1 y la deposita en la máquina 1, una vez que la máquina 1 termina, la pieza es pasada al almacén 2” (vea la figura 2.13).

La figura 2.14 muestra el supervisor. El supervisor es calculado usando la metodología propuesta en [8]. La composición concurrente⁵ de la figura 2.12 y la figura 2.14 daría como resultado al sistema bajo supervisión.

En este caso el sistema a lazo cerrado tendría 15 lugares y 31 transiciones.

Si la especificación cambia a:

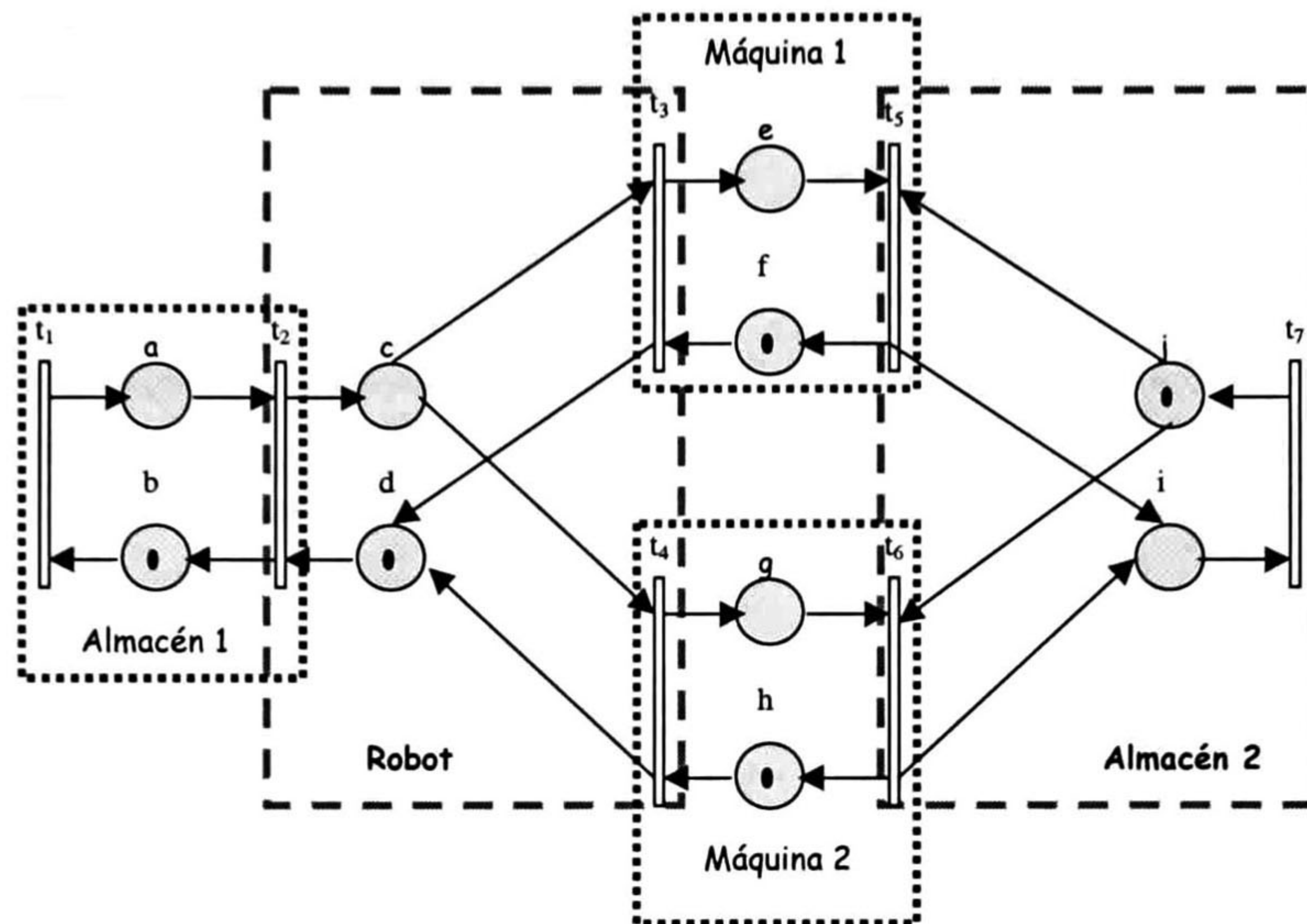
“si hay pieza en almacén 1, entonces el robot toma una pieza del almacén 1 y la deposita en la máquina 2, una vez que la máquina 2 termina, la pieza es pasada al almacén 2”,

cuando el estado de la máquina 1 indica que está procesando, entonces no existe supervisor para este caso, ya que no se partiría de la misma condición inicial. En este trabajo se muestra que utilizando la aproximación del seguimiento de modelo sí se puede controlar.

2.5 Conclusiones

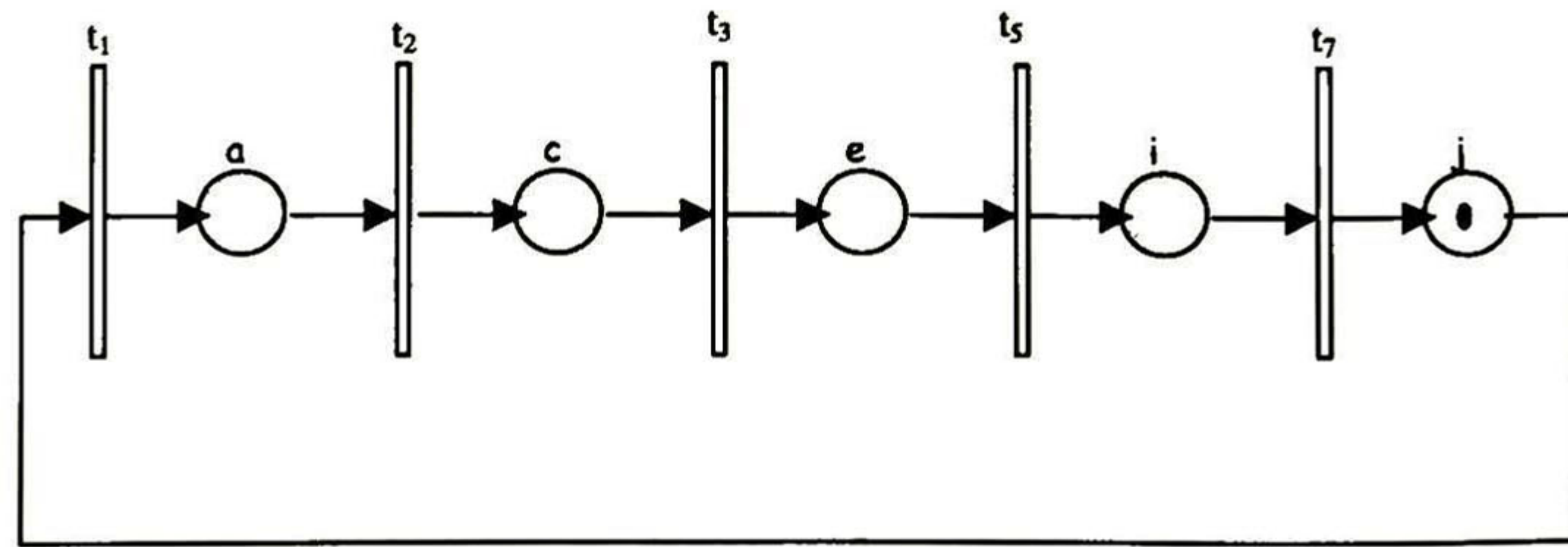
En este capítulo se presentaron los conceptos relacionados con lenguajes y RP que son la base para estudiar los problemas de control en SED . También presenta algunas otras

⁵La composición concurrente en RP es similar al producto síncrono total en AF , consiste de n lugares y r transiciones. Donde $n =$ número de lugares de la especificación y $r = \sum_{i=1}^m (\text{número de veces de } t_i) * (\text{número de veces de } t_i \text{ en el sistema})$, donde $m =$ número de transiciones t_i de la especificación.



<i>Lugares</i>		<i>Transiciones</i>	
a:	Pieza en almacén 1.	t ₁ :	Llega pieza al almacén 1.
b:	Almacén vacío.	t ₂ :	Robot toma pieza de almacén 1.
c:	Robot trabajando.	t ₃ :	Robot deposita pieza en máquina 1, o llega pieza a máquina 1.
d:	Robot ocioso.	t ₄ :	Robot deposita pieza en máquina 2, o llega pieza a máquina 2.
e:	Máquina 1 procesando pieza.	t ₅ :	Máquina 1 procesa pieza.
f:	Máquina 1 ociosa.	t ₆ :	Máquina 2 procesa pieza.
g:	Máquina 2 procesando pieza.	t ₇ :	Retirar pieza de almacén 2.
h:	Máquina 2 ociosa.		
j:	Almacén 2 vacío.		
i:	Pieza en almacén 2.		

Figura 2.12: Modelo del sistema de manufactura.



<i>Lugares</i>		<i>Transiciones</i>	
a:	Pieza en almacén 1.	t ₁ :	Llega pieza al almacén 1.
c:	Robot trabajando.	t ₂ :	Robot toma pieza de almacén 1.
e:	Máquina 1 procesando pieza.	t ₃ :	Robot deposita pieza en máquina 1, o llega pieza a máquina 1.
i:	Pieza en almacén 2.	t ₅ :	Máquina 1 procesa pieza.
j:	Almacén 2 vacío.	t ₇ :	Retirar pieza de almacén 2.

Figura 2.13: Modelo de referencia o especificación.

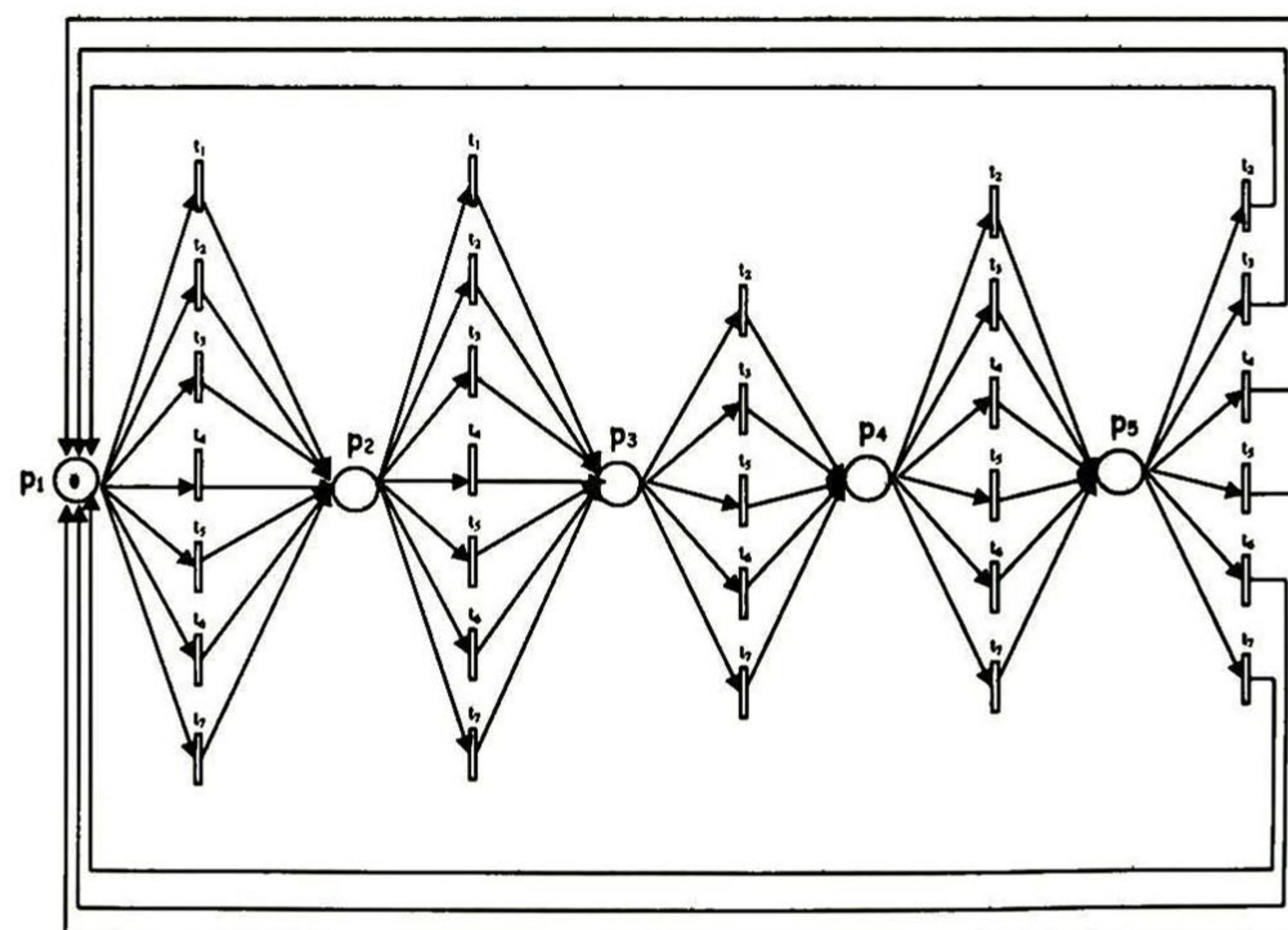


Figura 2.14: Supervisor para el sistema de manufactura.

definiciones como *RPI*, *observabilidad* y *controlabilidad* que serán utilizadas en los próximos capítulos.

También se mostraron los principales resultados que han sido presentados en la *teoría de control supervisor*, ya sea utilizando *AF* o *RP* acerca de los agentes llamados *supervisores*, mencionando sus características. En los *SMF*, los planes de proceso pueden cambiar continuamente y una nueva especificación puede darse sin importar el estado en que se encuentre el *SMF* en ese momento. Teniéndose que el lenguaje de la planta y de la especificación cambian, y con ello las restricciones de los trabajos de Wonham y Guia son violadas. Para abordar este caso, el presente trabajo plantea el estudio del problema de seguimiento de modelo en *SED*, que será presentado en el siguiente capítulo.

Capítulo 3

El problema de seguimiento de modelo

Resumen: Como se vió en el capítulo anterior, en el área de *SED* existen agentes llamados supervisores, los cuales logran restringir el comportamiento de un sistema a un comportamiento específico. Este tipo de controladores retroalimenta secuencias de eventos, lo cual hace que el controlador sea muy grande. Desafortunadamente, una retroalimentación de estados no siempre es posible; sin embargo, si se toma en cuenta el estado del sistema y de la especificación, es posible hacer retroalimentación de estado, resultando en controladores más simples y fáciles de implementar.

Este capítulo define el seguimiento de modelo e introduce nuevas definiciones como: *controlabilidad con respecto a un marcado o con respecto a un conjunto de salidas*, *lenguajes en RPI* y *modelo de referencia*. Finalmente, se dan las condiciones de existencia de un controlador y muestra cómo se debe diseñar éste.

3.1 Introducción

En este capítulo se introducen la definición del *problema de seguimiento de modelo en SED* y las condiciones que permiten a un sistema seguir una especificación. Para poder introducir el concepto problema de seguimiento de modelo, se dan nuevas definiciones; estas definiciones son utilizadas en el problema de seguimiento de modelo y en las condiciones bajo las cuales el problema de seguimiento tiene solución

3.2 Definiciones Básicas

En este trabajo se utilizarán sólo *RPI* binarias que tienen lugares marcados con diferentes símbolos de salida. A este tipo de redes se les llamará de etiquetado distinto. La siguiente definición formaliza esta idea.

Definición 3.1 *RPI de etiquetado distinto.*

Sea $S_f = (N, \Sigma, \Phi, \lambda, \varphi, \Delta)$ una *RPI*. Si $\forall M_i, M_j \in \mathcal{R}(S_f)$ y $\forall l_s, l_k \in L$ con ambos $\varphi(M_i(l_k)), \varphi(M_j(l_s)) \neq \varepsilon$ se tiene que $\varphi(M_i(l_k)) \neq \varphi(M_j(l_s))$, entonces S_f es llamada de etiquetado distinto.

3.2.1 La función φ

En el capítulo anterior aparece la función $\varphi_{RPI} : \mathcal{R}(N) \longrightarrow [\Phi \cup \{\varepsilon\}]^n$ que es una **función de etiquetado de los marcados** en la definición de *RPI*. Sin embargo esta función no será utilizada en el problema de **seguimiento de modelo**; en cambio, se utilizará una función φ que sólo será válida si la *RPI* es binaria y de etiquetado distinto. En trabajos futuros se tratará de abarcar más tipos de *RPI* y utilizar la función φ_{RPI} directamente. Por lo pronto se definirá la nueva **función φ** .

Primeramente se introduce un orden en el conjunto de los símbolos, después se definirá el vector de representación de símbolos de salida y finalmente la función φ .

Suponga que $S_f = (N_{S_f}, \Sigma_{S_f}, \Phi_{S_f}, \lambda_{S_f}, \varphi_{S_f}, \Delta)$ y $R_m = (N_{R_m}, \Sigma_{R_m}, \Phi_{R_m}, \lambda_{R_m}, \varphi_{R_m}, \Delta)$ son dos *RPI* donde $\Phi_{R_m} \subseteq \Phi_{S_f}$, y sean Φ' los símbolos que serán relevantes en S_f donde $\Phi' \subseteq \Phi_{R_m} \subseteq \Phi_{S_f}$, e $I = \{1, 2, \dots, u\}$ un conjunto de índices. Entonces $|\Phi'| = u \leq |\Phi_{S_f}|$. El isomorfismo $f : \Phi' \longrightarrow I$ induce un orden en Φ' de forma canónica: $\forall a, b \in \Phi', a < b$ si y sólo si $f(a) < f(b)$.

La función de orden dentro del conjunto Φ' sirve para definir una relación de orden entre los elementos de $\Phi' \subseteq \Phi_{R_m} \subseteq \Phi_{S_f}$, y con este orden establecer la comparación entre las salidas de ambos sistemas. El vector de representación de los símbolos de salida \mathbf{A}_j se forma de la siguiente manera.

$$\mathbf{A}_j = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_u \end{bmatrix}, \quad \alpha_i = \sum_1^n r_k$$

y

$$r_k = \begin{cases} 1 & \text{si } \varphi_{S_f}(M_j^{S_f}(l_k)) \in \Phi' \quad \wedge \quad f(\varphi_{S_f}(M_j^{S_f}(l_k))) = i \\ 0 & \text{en otro caso} \end{cases}$$

$$\begin{array}{ccc}
\mathcal{R}(S_f) & \xrightarrow{\varphi} & [\mathbb{N} \cup \{0\}]^u \\
& \searrow \varphi_{S_f} & \nearrow VR \\
& & [\Phi_{S_f} \cup \{\varepsilon\}]^n
\end{array}$$

Figura 3.1: Diagrama de funciones φ , φ_{S_f} y VR .

Es decir, α_i es el número de veces que el mismo símbolo b con $f(b) = i$, aparece en $\varphi_{S_f}(M_j^{S_f})$. Como ya se dijo \mathbf{A}_j es la representación del vector de $\varphi_{S_f}(M_j^{S_f})$ y su cálculo se denota como: $\mathbf{A}_j = VR(\varphi_{S_f}(M_j^{S_f}))$. Se sigue entonces que $VR : [\Phi_{S_f} \cup \{\varepsilon\}]^n \rightarrow [\mathbb{N} \cup \{0\}]^u$. Como se utilizarán *RPI* binarias de etiquetado distinto para modelar los sistemas se tiene que $\alpha_i \in \{0, 1\}$.

A continuación se presenta la definición de una **función** que relaciona a los marcados alcanzados con una representación del vector de símbolos de salida.

Definición 3.2 Sea S_f una *RPI* y $\varphi_{S_f} : \mathcal{R}(S_f) \rightarrow [\Phi \cup \{\varepsilon\}]^n$ su función de etiquetado sobre los marcados. Entonces, la **función** φ de representación vectorial de símbolos de salida es $\varphi : \mathcal{R}(S_f) \rightarrow [\mathbb{N} \cup \{0\}]^u$ puede definirse como: $\varphi(M_j^{S_f}) = \mathbf{A}_j$, si y sólo si $\mathbf{A}_j = VR(\varphi_{S_f}(M_j^{S_f}))$.

Suponga que:

$$\varphi(M_j^{S_f}) = \mathbf{A}_j = \begin{bmatrix} \alpha_{1j} \\ \vdots \\ \alpha_{uj} \end{bmatrix}$$

es la representación del vector de los símbolos de salida para el marcado $M_j^{S_f}$ en S_f . Si S_f es de etiquetado distinto, entonces para todo marcado alcanzable a lo más existe un lugar l_k que al estar marcado produce el símbolo b tal que $f(b) = i$. En este caso los α_{ij} no dependen del marcado $M_l^{S_f}$, es decir $\forall M_j^{S_f} \in \mathcal{R}(S_f)$, $\alpha_i = \alpha_{ij} = \vec{v}_k M_j^{S_f}$, donde \vec{v}_k es un vector renglón en el que existe sólo un 1 en la k -ésima posición y sus otros elementos son 0. Entonces en las *RPI* de etiquetado distinto φ puede representarse por la matriz:

$$\varphi(M_j^{S_f}) = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_u \end{bmatrix} = \begin{bmatrix} \varphi_{1,1} & \cdots & \varphi_{1,n} \\ \vdots & & \vdots \\ \varphi_{u,1} & \cdots & \varphi_{u,n} \end{bmatrix} \begin{bmatrix} M_j^{S_f}(l_1) \\ \vdots \\ M_j^{S_f}(l_n) \end{bmatrix} \quad (3.1)$$

donde $\alpha_i = [\varphi_{i1} \cdots \varphi_{in}]$. El diagrama 3.1 muestra las relaciones entre las funciones φ , φ_{S_f} , VR .

3.2.2 Ejemplo de modelado

En un *SED* existen actuadores, sensores y el sistema en sí. Para modelar un *SED* se utiliza como formalismo las Redes de Petri Interpretadas, cuya principal característica es que tiene dos funciones de etiquetado (λ , φ) para relacionar estados con señales de sensores y transiciones con señales de actuadores. Un modelo de un *SED* en *RPI* se obtiene de la siguiente forma:

1. Para cada estado posible de un actuador, existe un lugar en la *RPI*. Los estados de cada actuador se conectan mediante las transiciones que hacen posible que el actuador cambie de estado, y las transiciones se etiquetan con el evento (comando) que hacen que el actuador cambie de estado.
2. El comportamiento o dinámica del sistema es lo que se captura en el modelo interno y posee sólo transiciones incontrolables. Los lugares del modelo interno deben etiquetarse con las señales de los sensores. La dinámica interna puede modelarse como indican
3. Un autolazo debe añadirse desde un lugar del actuador hacia una transición del modelo interno, cuando el estado del actuador hace que el sistema evolucione del estado previo de la transición al posterior.

A continuación se muestran algunos ejemplos.

Ejemplo 3.1 Sistema “nivel de líquido”

Como se mencionó en el capítulo 1 el sistema “nivel de líquido” consiste en un tanque, dos válvulas (*válvula 1* y *válvula 2*) y dos sensores (*Bajo* y *Alto*) vea la figura 1.2. El sistema se basa en tres estados del nivel de líquido: *vacío* (cuando el agua no alcanza ninguno de los sensores); *bajo* (cuando el agua rebasa sólo al sensor *Bajo*); y *alto* (cuando el agua rebasa al sensor *Alto*); también se tienen dos estados de la *válvula 1*: *abierta* o *cerrada*. La *válvula 2* tiene un estado fijo y no se puede mover.

El comportamiento del sistema inicialmente es el siguiente: “El nivel de líquido en tanque es *vacío* y el estado de la *válvula 1* es *cerrada*. Cuando se da el evento *abrir válvula 1*, el nivel de líquido en el tanque pasará de *vacío* a *Bajo* o de *Bajo* a *Alto*; mientras el estado de la *válvula 1* sea *abierta* continuará subiendo el nivel de líquido hasta tocar el sensor *Alto*. El evento *cerrar válvula 1* hace que la *válvula 1* cambie de estado a *cerrada* y el nivel de líquido en el tanque disminuirá de la siguiente forma: si el nivel es *Alto* entonces pasará a *Bajo* y si es *Bajo*, pasará a *vacío*” En este momento el nivel de líquido en el tanque es *vacío* y el estado de la *válvula 1* es *cerrada*, que son las condiciones iniciales del sistema.

Para obtener el modelo del sistema se realizó lo siguiente:

1. Debido a que existe un actuador (*válvula 1*) que tiene dos estados, entonces dos lugares y dos transiciones deben usarse para representar el comportamiento de la válvula. Las transiciones son los eventos *abrir válvula 1* (representado por el símbolo *A*) y *cerrar válvula 1* (representado por el símbolo *C*). Vea la figura 3.2.
2. El modelo interno del sistema tiene tres estados que representan el nivel de líquido en el tanque (*vacío*, *bajo* y *alto*). Cuando el sistema está en estado *bajo* el sensor detecta

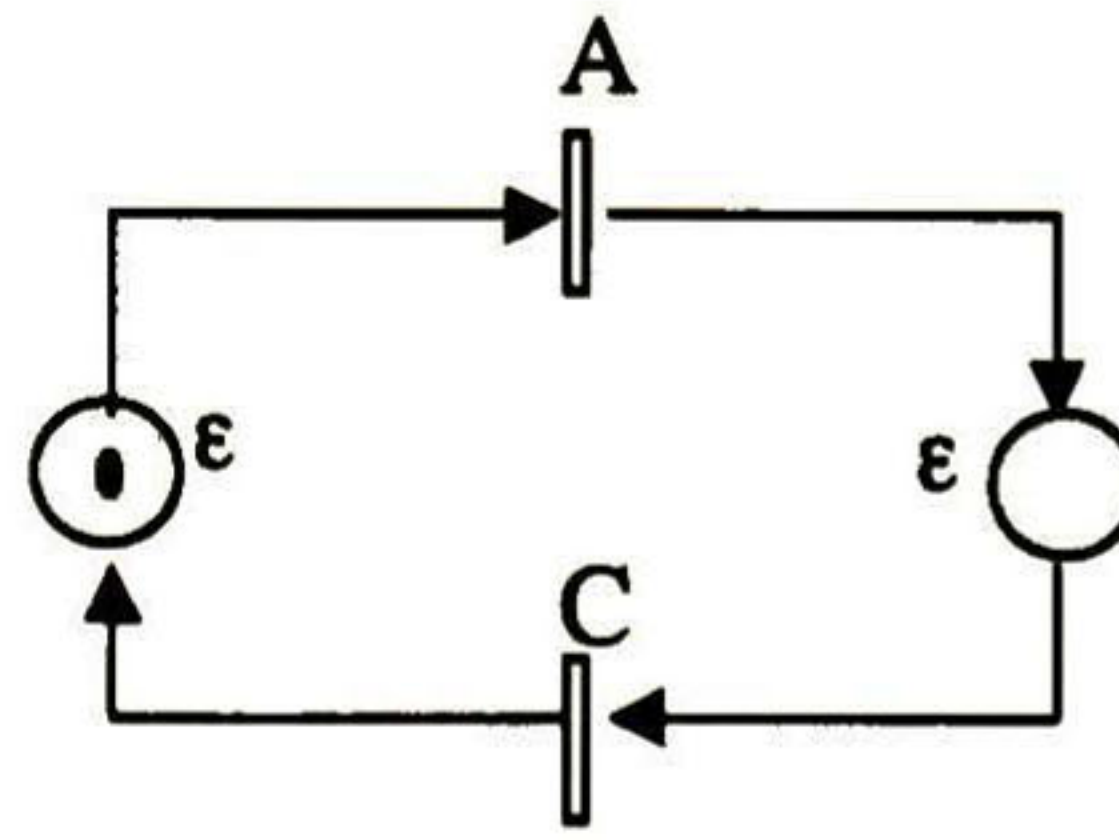


Figura 3.2: RPI de la válvula 1.

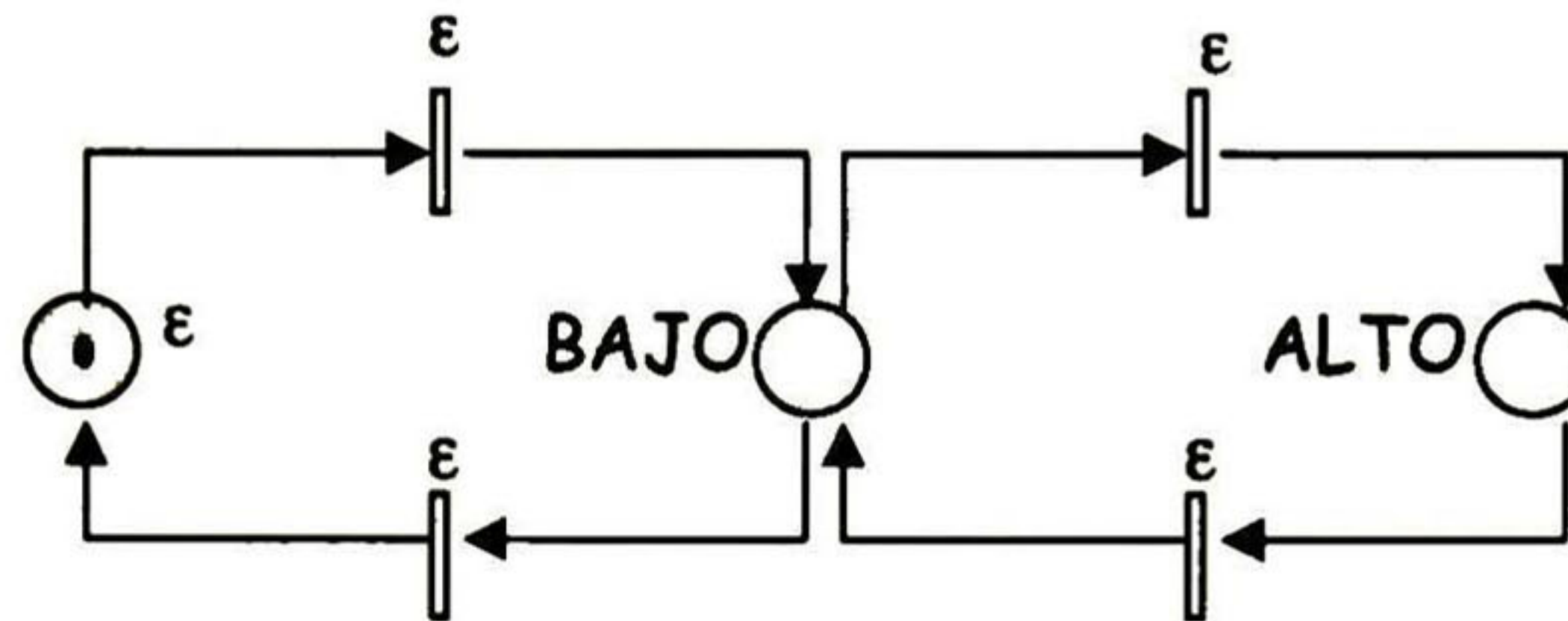


Figura 3.3: RPI del modelo interno del sistema "Nivel de líquido"

el nivel en **Bajo** entonces se etiqueta a este estado (lugar) con el símbolo asociado al sensor (en este caso **Bajo**), ocurre lo mismo cuando el sistema está en estado **alto**, sin embargo el estado del tanque vacío no tiene asociado ningún sensor, lo que se representa en una RPI como un lugar no medible. La figura 3.3 muestra el modelo interno del sistema.

3. Se agregan autolazos de la siguiente manera, analizando el comportamiento del sistema se observa que: "El nivel de líquido en tanque es vacío y el estado de la válvula 1 es cerrada, cuando se da el evento abrir válvula 1, el nivel de líquido en el tanque subirá hasta rebasar el sensor **Bajo** entonces se debe hacer un autolazo entre el estado de la válvula 1 abierta a la transición que lleve al tanque del estado vacío al estado bajo. La figura 3.4 muestra el sistema "nivel de líquido" modelada como una RPI.

4. La función φ es definida.

$$\varphi = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

Ejemplo 3.2 Sistema "Coche eléctrico"

El sistema "Coche eléctrico" consiste de un coche eléctrico, dos sensores (**Sensor 1** y **Sensor 2**) y dos botones (**Derecha** e **Izquierda**). El coche se mueve de izquierda a derecha activando los sensores (vea la figura). El sistema se basa en tres estados del coche eléctrico: **a** (cuando el coche está en la misma posición que el **Sensor 1**); **x** (cuando el coche no está en la posición de ninguno de los sensores); y **b** (cuando el coche está en la misma posición que el **Sensor 2**); también se tienen dos estados para los botones **Derecha** e **Izquierda**: **activado** o **desactivado**. Vea la figura 3.5.

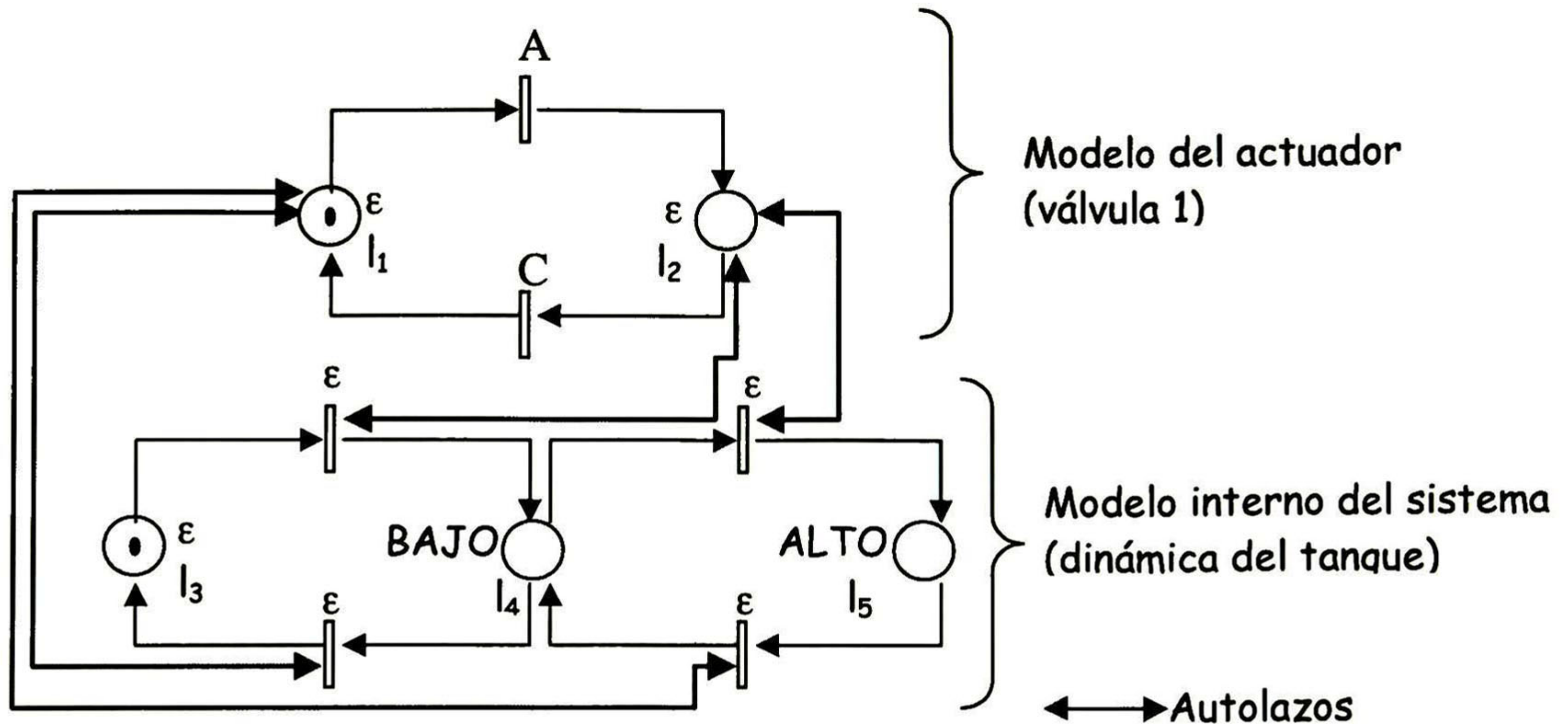


Figura 3.4: RPI del sistema "Nivel de líquido"

El comportamiento del sistema es el siguiente: "Cuando el coche está en su posición inicial el **Sensor 1** activa una señal llamada **a**; cuando el coche recibe el evento **mover a la derecha** se mueve a la derecha y cuando recibe el evento **mover a la izquierda** se mueve en este sentido. Cuando el coche llega a la posición del **Sensor 2** activa una señal llamada **b**. Si no se envía un evento de entrada (**mover a la derecha** o **mover a la izquierda**) el coche no se mueve"

Para obtener el sistema modelado se realizó lo siguiente:

1. Debido a que existen dos actuadores (**Derecha e Izquierda**) y cada uno tiene dos estados, entonces cuatro lugares y cuatro transiciones deben usarse para representar el comportamiento de los botones. Las transiciones son los eventos **mover a la derecha** (representado por el símbolo **D**), **desactivar botón derecha** (representado por el símbolo \bar{D}), **mover a la izquierda** (representado por el símbolo **I**) y , **desactivar botón izquierda** (representado por el símbolo \bar{I}). Vea la figura 3.6.
2. El modelo interno del sistema tiene tres estados (**a**, **x** y **b**) que representan la ubicación del coche con respecto a los sensores. Cuando el **Sensor 1** detecta que el coche se encuentra a la izquierda entonces se etiqueta a este estado (lugar) con el símbolo asociado al sensor (en este caso **a**), similarmente pasa con el **Sensor 2**, pero existe un estado (**x**) que representa la no activación de los sensores, este estado por lo tanto no tiene ningún sensor asignado y se representa en una RPI como un lugar **no medible**. La figura 3.7 muestra el modelo interno del sistema.
3. Se agregan autolazos de la siguiente manera, analizando el comportamiento del sistema se observa que: "El coche eléctrico está activando al **Sensor 1** y el estado del botón **Derecha** es **desactivado**, cuando se da el evento **mover a la derecha**, el coche eléctrico se mueve a la derecha (siempre y cuando el estado del botón **Izquierda** sea **desactivado**) hasta llegar a la posición del **Sensor 2** si el botón **Derecha** está en

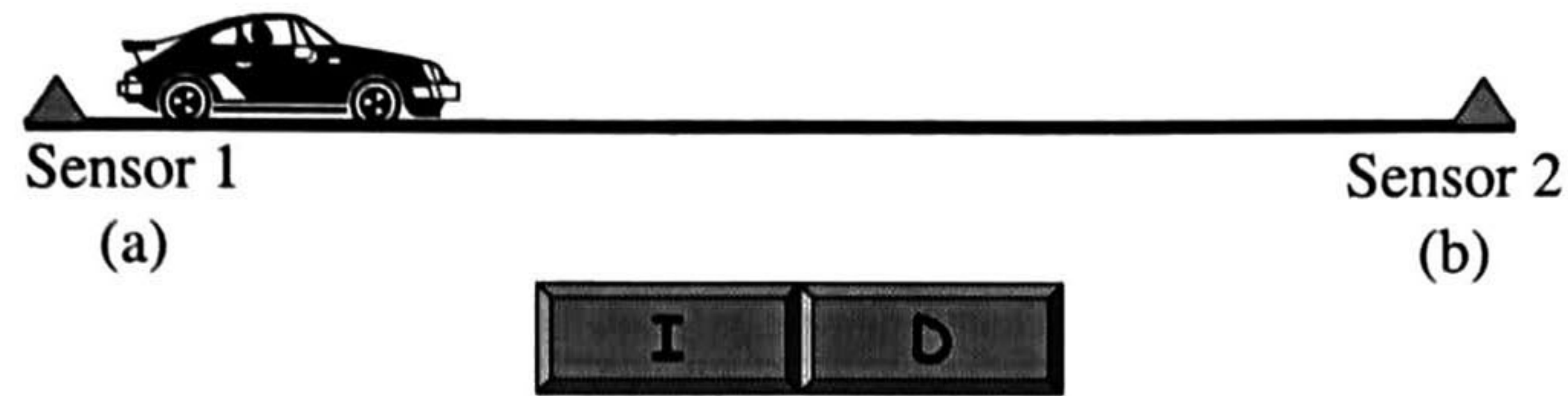


Figura 3.5: Sistema “Coche eléctrico”

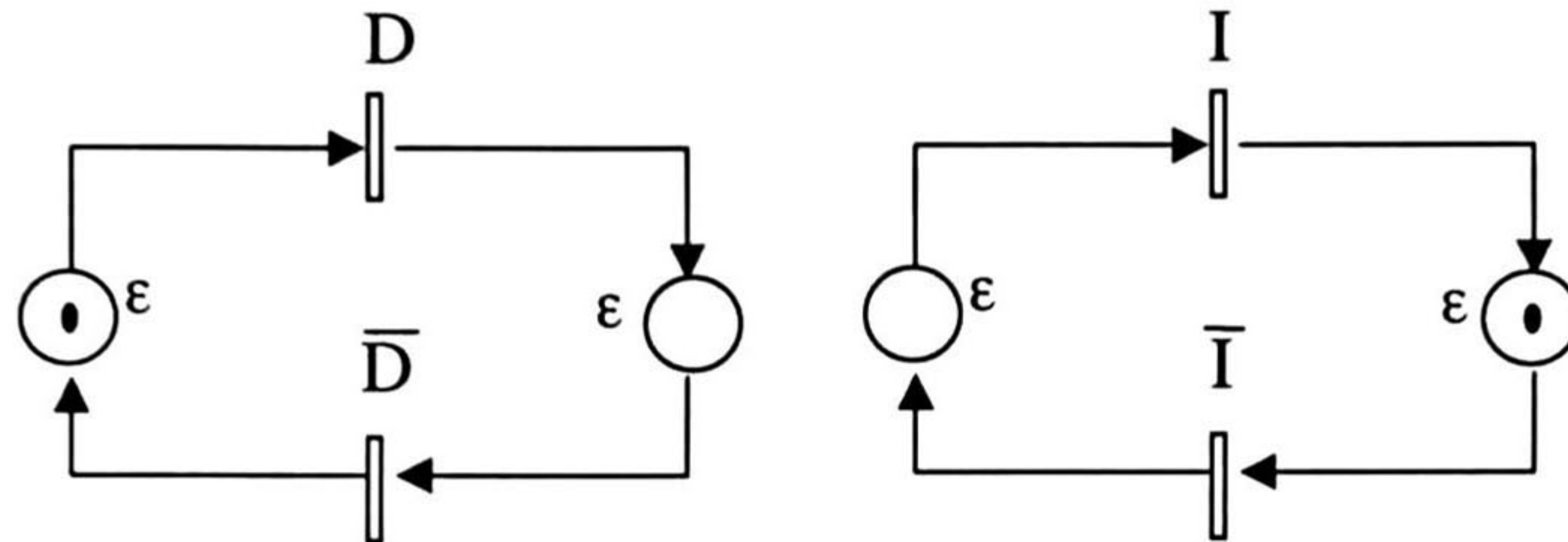


Figura 3.6: RPI de los actuadores Derecha e Izquierda.

estado *activado*” entonces se debe hacer un autolazo entre el estado del botón *Derecha activado* a la transición que lleve al coche del estado *a* al estado *x*. La figura 3.8 muestra el sistema “Coche eléctrico” modelada como una RPI.

4. La función φ es definida.

$$\varphi = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

3.2.3 Lenguajes de una RPI

Como una RPI contiene dos alfabetos y dos funciones de etiquetado, las cuales se relacionan con los actuadores, sensores y eventos internos del sistema, se pueden definir dos lenguajes: un lenguaje relacionado con la secuencia de disparo de transiciones (llamado lenguaje de entrada) y un lenguaje relacionado con la secuencia de marcados alcanzados (llamado lenguaje de salida) se obtienen las siguientes definiciones.

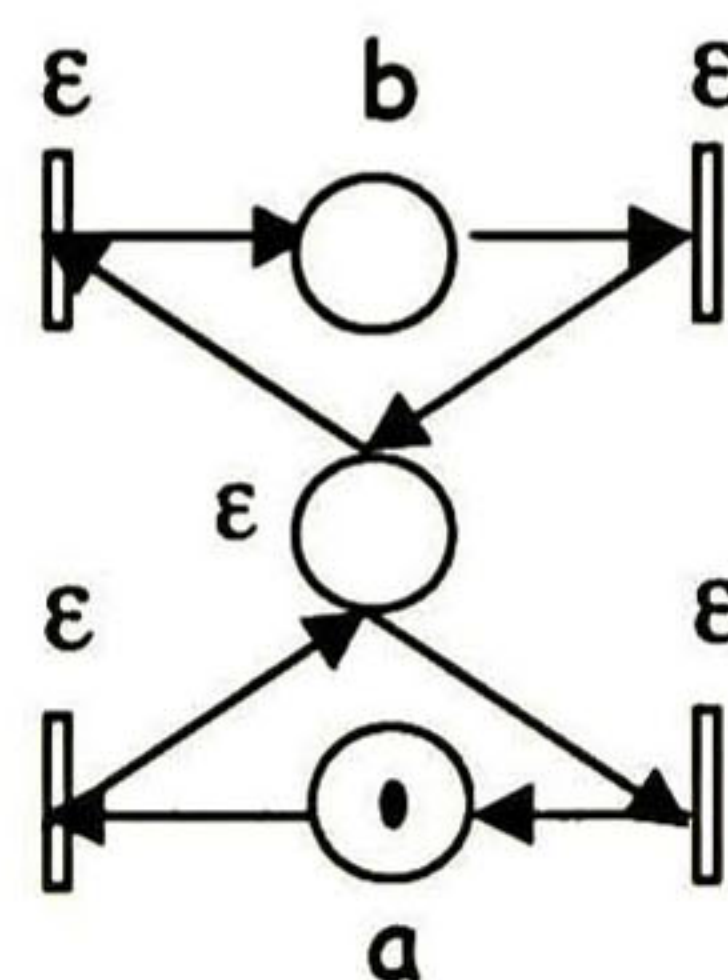


Figura 3.7: RPI del modelo interno del sistema “Coche eléctrico”.

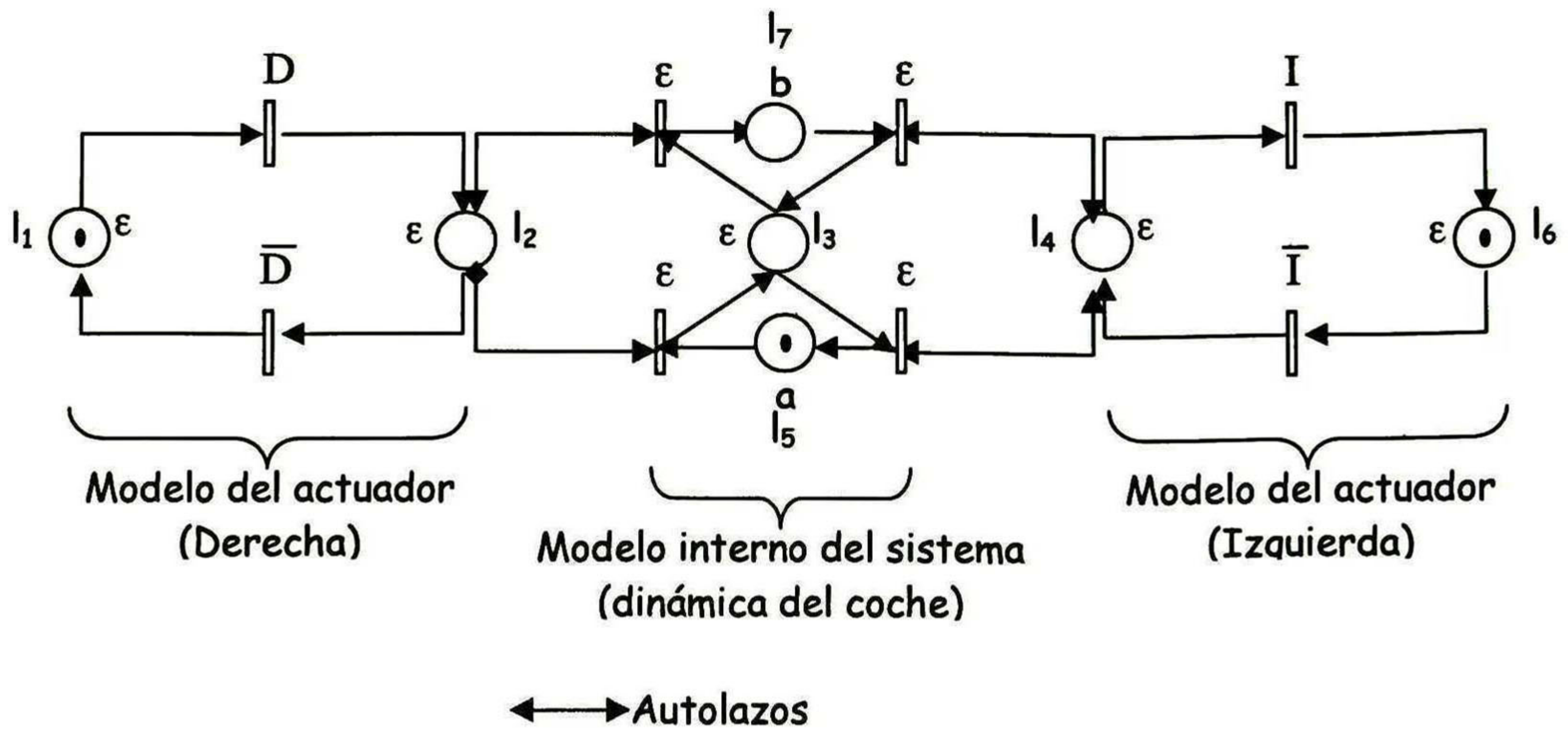


Figura 3.8: RPI del sistema "Coche eléctrico"

Definición 3.3 *Lenguaje de entrada (L_I)*. Sea $RPI(N, \Sigma, \Phi, \lambda, \varphi_{RPI}, \Delta)$ una red de Petri interpretada, M_0 el marcado de la RPI y $\gamma = t_1 t_2 \dots t_k$ una secuencia de disparos a partir de M_0 , esto es:

$$M_0 \xrightarrow{\gamma} M$$

es decir que γ pertenece al lenguaje de disparos de la RPI ($\gamma \in L(N, M_0^{RPI})$) entonces $\sigma_k = \lambda(t_1) \lambda(t_2) \dots \lambda(t_k)$ (o $\sigma_k = \lambda(\gamma)$ para simplificar), será una palabra de entrada. El conjunto de todas las palabras de entrada es llamado lenguaje de entrada, y es denotado por $L_I(M_0^{RPI})$

Definición 3.4 *Lenguaje de salida (L_O)*. Sea $RPI(N, \Sigma, \Phi, \lambda, \varphi_{RPI}, \Delta)$ una red de Petri interpretada, M_0 el marcado de la RPI y $\gamma = t_1 t_2 \dots t_k$ una secuencia de disparos a partir de M_0 , esto es:

$$M_0 \xrightarrow{\gamma} M$$

es decir que γ pertenece al lenguaje de disparos de la RPI ($\gamma \in L(N, M_0^{RPI})$) entonces

$$M_0^{RPI} \xrightarrow{t_1} M_{0+1}^{RPI} \xrightarrow{t_2} M_{0+2}^{RPI} \dots \xrightarrow{t_k} M_{0+k}^{RPI}$$

La palabra $w_k = \varphi_{RPI}(M_0^{RPI}) \varphi_{RPI}(M_{0+1}^{RPI}) \dots \varphi_{RPI}(M_{0+k}^{RPI})$ será llamada una palabra de salida. El conjunto de todas las palabras de salida es el lenguaje de salida, y es denotado por $L_O(M_0^{RPI})$.

Definición 3.5 *Lenguaje de salida de la representación vectorial (L_{O_Φ})*. Sea $RPI(N, \Sigma, \Phi)$ una red de Petri interpretada, $L_O(M_0^{RPI})$ el lenguaje de salida de la RPI y sea $W = w_1 w_2 \dots w_q \in L_O(M_0^{RPI})$ una palabra de salida. La palabra $v_k = VR(w_1) VR(w_2) \dots VR(w_q)$ es llamada la palabra de salida de la representación vectorial de la palabra W . El conjunto

$$L_{O_\Phi}(M_0^{RPI}) = \{v_k \mid v_k = VR(w_1) VR(w_2) \dots VR(w_q) \forall W \in L_O(M_0^{RPI})\}$$

es llamado el lenguaje de salida de la representación vectorial. Donde el vector cero es el símbolo vacío de este lenguaje, cuya longitud es cero.

Definición 3.6 *Lenguaje de entrada de la representación vectorial para la palabra V .* Sean $S_f = (N_{S_f}, \Sigma_{S_f}, \Phi_{S_f}, \lambda_{S_f}, \varphi_{S_f}, \Delta)$ y $R_m = (N_{R_m}, \Sigma_{R_m}, \Phi_{R_m}, \lambda_{R_m}, \varphi_{R_m}, \Delta)$ dos RPI binarias, vivas y de etiquetado distinto; Φ' el conjunto de símbolos relevantes en S_f y R_m ; los marcados actuales de las RPI son $M_0^{S_f}$ y $M_0^{R_m}$ respectivamente y $V = v_1 v_2 \dots v_k \in L_{O_{\Phi'}}(M_0^{R_m})$, entonces, el lenguaje de entrada de la representación vectorial para la palabra V en S_f es el lenguaje:

$$L_{I_{\Phi'}}(V^{R_m}) = \left\{ \gamma = t_1 t_2 \dots t_q \in L(N_{S_f}, M_0^{S_f}) \left| \begin{array}{l} M_0^{S_f} \xrightarrow{t_1} M_{0+1}^{S_f} \xrightarrow{t_2} \dots \xrightarrow{t_q} M_{0+k}^{S_f} \quad \wedge \\ V = \varphi(M_0^{S_f}) \dots \varphi(M_{0+k}^{S_f}) \end{array} \right. \right\}$$

Note que algunas $\varphi(M_i^{S_f})$ pueden ser cero (o el símbolo vacío) y se sigue obteniendo la secuencia de salida V

Básicamente el lenguaje de entrada describe una ley de control para el *SED* mientras que el lenguaje de salida describe el comportamiento del sistema visto a través de los sensores, por lo que pueden definirse propiedades como: *controlabilidad, observabilidad, identificabilidad, estabilidad*, etc. A continuación se mencionan algunas de ellas.

3.2.4 Controlabilidad

Un *SED* es controlable si los estados especificados a priori pueden alcanzarse usando secuencias de transiciones controlables o incontrolables, es decir, que exista una ley de control que mueva al sistema de un estado inicial a un estado final. Esta idea se presenta a continuación.

Definición 3.7 *Controlabilidad con respecto a un marcado inicial.*

Sea $S_f = (N_1, \Sigma, \Phi, \lambda, \varphi, \Delta)$ una RPI, $\mathcal{M}_F = \{M_{f_1}, M_{f_2}, \dots, M_{f_k}\}$ el conjunto de marcados finales y M_0 el marcado inicial de S_f . El sistema S_f es **controlable con respecto a \mathcal{M}_F** si para cada $M_{f_i} \in \mathcal{M}_F$ existe por lo menos una $\gamma = t_1 t_2 \dots t_k$ secuencia de disparos en S_f a partir de M_0 , tal que:

$$M_0 \xrightarrow{\gamma} M_{f_i}$$

y para $\sigma_k = \lambda(t_1) \lambda(t_2) \dots \lambda(t_k)$ (o $\sigma_k = \lambda(\gamma)$ para simplificar) se cumple una de las siguientes condiciones:

1. $\forall \alpha \in \overline{\sigma_k}$ tal que $M_0 \xrightarrow{\alpha} M_x$ entonces M_x no habilita transiciones incontrolables en S_f .
2. Si $\exists \alpha \in \overline{\sigma_k}$ tal que $M_0 \xrightarrow{\alpha} M_x$, y M_x habilita transiciones incontrolables en S_f , entonces $\forall M_y$ alcanzado desde M_x por el disparo de una transición incontrolable $\exists \gamma'$ tal que $M_y \xrightarrow{\gamma'} M_{f_i}$, con $\sigma'_k = \lambda(\gamma')$ que satisface una de las condiciones anteriores.

Un caso especial de controlabilidad surge cuando se desea mover desde M_0 a M_i mientras que se generan sólo algunas palabras de salida especificadas apriori. La siguiente definición formaliza este caso.

Definición 3.8 *Controlabilidad con respecto a un conjunto de secuencias \mathbf{k}*

Sean $S_f = (N_{S_f}, \Sigma_{S_f}, \Phi_{S_f}, \lambda_{S_f}, \varphi_{S_f}, \Delta)$ y $R_m = (N_{R_m}, \Sigma_{R_m}, \Phi_{R_m}, \lambda_{R_m}, \varphi_{R_m}, \Delta)$ dos RPI binarias, vivas y de etiquetado distinto; Φ' el conjunto de símbolos relevantes en S_f y R_m tal que $\mathbf{k} = L_{O_{\Phi'}}(M_0^{R_m}) \subseteq L_{O_{\Phi'}}(M_0^{S_f})$; los marcados iniciales son $M_0^{S_f}$ y $M_0^{R_m}$ respectivamente y $\Sigma_i = \{t_i \mid t_i \in \Sigma_{S_f} \wedge \lambda_{S_f}(t_i) = \varepsilon\}$. Se dice que R_m es controlable con respecto a S_f si $\forall V \in \mathbf{k} \exists \gamma \in L_{I_{\Phi'}}(V^{R_m})$ que cumple con:

$$\bar{\gamma}t_i \cap L(N_{S_f}, M_0^{S_f}) \subseteq \bar{\gamma}$$

para $\forall t_i \in \Sigma_i$. Entonces γ es llamada *secuencia controlable*.

El problema de seguimiento de modelo consta de un sistema a controlar y una especificación o modelo a seguir; a continuación se presenta la definición de este último bajo nuestro contexto.

Definición 3.9 *Modelo de referencia (R_m)*.

Un modelo de referencia R_m es una RPI binaria y viva que representa el modelo a ser seguido. El modelo de referencia es totalmente controlable y medible (es decir, no tiene transiciones incontrolables ni lugares no medibles) y se representa como:

$$R_m = (N_{R_m}, \Sigma_{R_m}, \Phi_{R_m}, \lambda_{R_m}, \varphi_{R_m}, \Delta)$$

3.3 El problema de seguimiento de modelo

Antes de definir el *problema de seguimiento de modelo* es necesario definir una *norma* para comparar la salida del sistema con la del modelo de referencia.

Definición 3.10 *Error entre las salidas de S_f y R_m*

Sean $R_m = (N_{R_m}, \Sigma_{R_m}, \Phi_{R_m}, \lambda_{R_m}, \varphi_{R_m}, \Delta)$ y $S_f = (N_{S_f}, \Sigma_{S_f}, \Phi_{S_f}, \lambda_{S_f}, \varphi_{S_f}, \Delta)$ dos RPI vivas, binarias y de etiquetado distinto; $M_l^{S_f}$ el l -ésimo marcado de S_f y $M_i^{R_m}$ el i -ésimo marcado¹ de R ; Φ' el conjunto de símbolos relevantes en S_f y R_m ; φ y φ' las funciones de representación vectorial de los símbolos de salida con respecto a Φ' . Entonces el error se define como:

$$\mathcal{E}_i^l = \left\| \varphi(M_l^{S_f}) - \varphi'(M_i^{R_m}) \right\| = \|\mathbf{A}_l - \mathbf{A}_i\| = \left\| \begin{bmatrix} e_1 \\ \vdots \\ e_q \end{bmatrix} \right\| = |e_1| + |e_2| + \cdots + |e_q|$$

A continuación se define el problema de seguimiento de modelo cuando se tiene información completa del estado.

¹Note que el índice de los marcados de S_f y R_m son diferentes, ya que S_f y R_m tienen diferentes entradas y por ello han evolucionado con longitudes de palabras distintas.

Definición 3.11 *Problema de seguimiento de modelo con información total del estado.*

Sean $R_m = (N_{R_m}, \Sigma_{R_m}, \Phi_{R_m}, \lambda_{R_m}, \varphi_{R_m}, \Delta)$ y $S_f = (N_{S_f}, \Sigma_{S_f}, \Phi_{S_f}, \lambda_{S_f}, \varphi_{S_f}, \Delta)$ el modelo de referencia y la planta respectivamente; $M_l^{S_f}$ el l -ésimo marcado de S_f ; $M_i^{R_m}$ el i -ésimo marcado de R_m y $\mathbf{k} = L_{O_{\Phi}}(M_0^{R_m}) \subseteq L_{O_{\Phi}}(M_{l+N}^{S_f})$ controlable con respecto a S_f . Si la ocurrencia de eventos en S_f es mucho mayor que la ocurrencia de eventos en R_m para todo horizonte de observación, entonces el problema de seguimiento de modelo consiste en encontrar un controlador H :

$$\begin{aligned} H : \mathcal{R}(M_0^{S_f}) \times \mathcal{R}(M_0^{R_m}) &\longrightarrow \Sigma_{S_f} \\ u_i^i &:= H(M_l^{S_f}, M_i^{R_m}) \end{aligned} \quad (3.3)$$

que hace que:

$$\forall i \exists N < \infty \quad \lim_{k_i \rightarrow N} \left\| \varphi(M_{l+k_i}^{S_f}) - \varphi'(M_i^{R_m}) \right\| = 0$$

■

El problema de seguimiento de modelo con información total del estado puede representarse por el esquema de la figura 3.9, donde:

- R_m representa el modelo a seguir.
- S_f representa el sistema a ser controlado.
- H es el controlador.

3.4 Condiciones para el seguimiento de modelo

En esta sección se establecen las condiciones de existencia que permiten determinar cuándo tiene solución el problema de seguimiento de modelo con información total del estado. A continuación se presentan las hipótesis de trabajo:

Hipótesis 3.1 *El sistema es representado por una RPI binaria y viva, y se dispone de toda la información de su estado.*

Hipótesis 3.2 *Todos los lugares y transiciones están etiquetados con símbolos diferentes, esto es:*

$$\begin{aligned} \forall l_i, l_j \in L_{S_f} \quad \varphi_{S_f}(M_k^{S_f}(l_i)) &\neq \varphi_{S_f}(M_k^{S_f}(l_j)) \\ \forall t_i, t_j \in T_{S_f} \quad \lambda(t_i) &\neq \lambda(t_j) \end{aligned}$$

Teorema 4 *Sean $S_f = (N_{S_f}, \Sigma_{S_f}, \Phi, \lambda_{S_f}, \varphi_{S_f}, \Delta)$ y $R_m = (N_{R_m}, \Sigma_{R_m}, \Phi, \lambda_{R_m}, \varphi_{R_m}, \Delta)$ dos RPI vivas, binarias y de etiquetado distinto donde S_f es el modelo de la planta y R_m es*

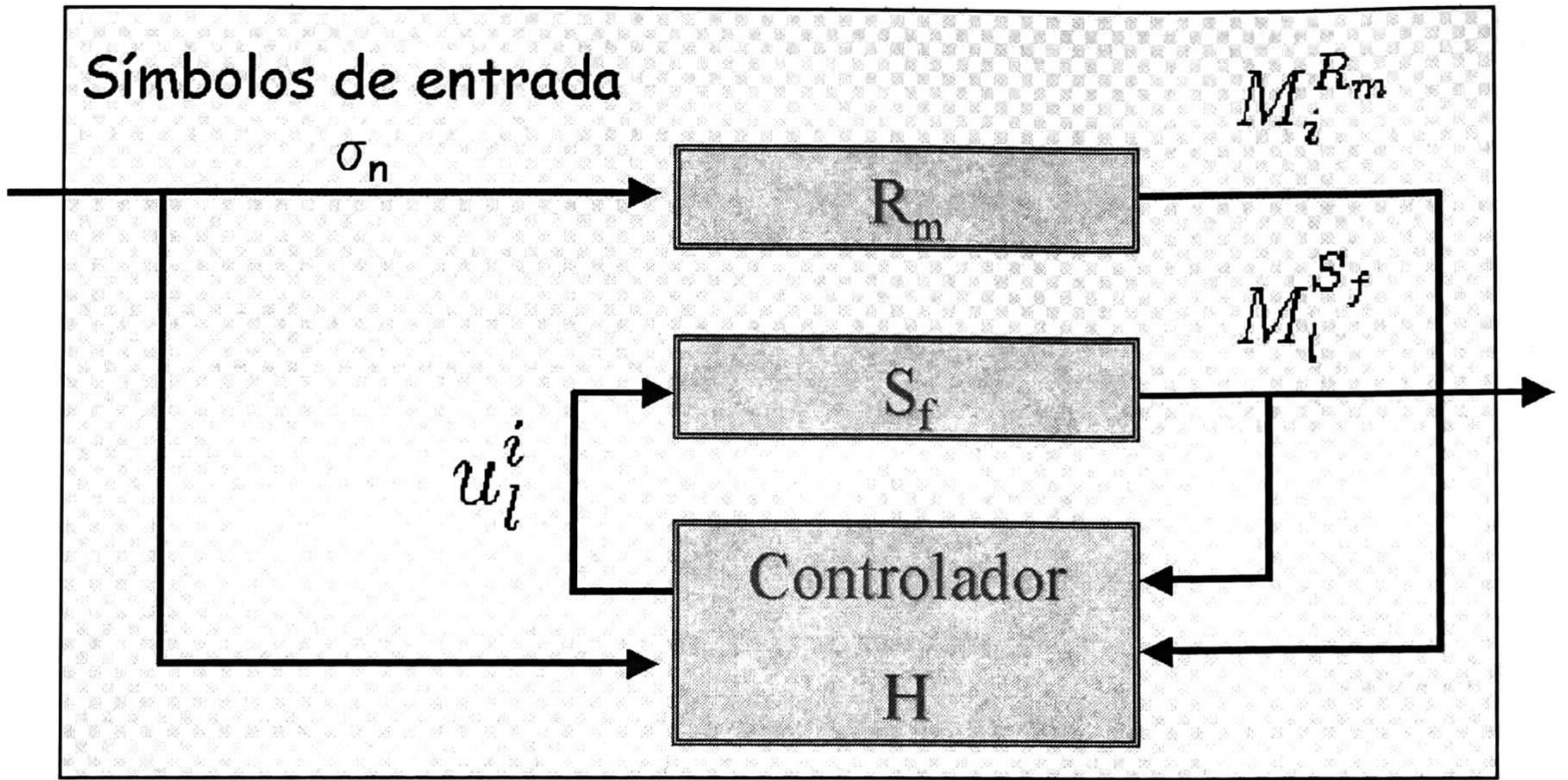


Figura 3.9: Esquema del problema de “seguimiento de modelo con información total del estado”.

el modelo de referencia; $\Phi' \subseteq \Phi$ el conjunto de símbolos relevantes, donde la ocurrencia de eventos en S_f es mucho mayor que en R_m y las ecuaciones de estados de S_f y R_m son:

$$S_f = \begin{cases} M_k^{S_f} = M_0^{S_f} + C \cdot \overrightarrow{v_{k-1}} \\ b_{k_i} = \varphi_{S_f}(M_k^{S_f}) \end{cases} \quad \text{y} \quad R_m = \begin{cases} M_i^{R_m} = M_0^{R_m} + Q \cdot \overrightarrow{z_{i-1}} \\ r_i = \varphi_{R_m}(M_i^{R_m}) \end{cases} \quad (3.4)$$

Si existe una función Π tal que satisface las siguientes condiciones:

$$1) \Pi \cdot M_0^{R_m} = M_s^{S_f}$$

$$2) \varphi \circ \Pi = \varphi'$$

y además existen las secuencias W_I, W_{S_i} tales que:

a) $M_s^{S_f}$ es un marcado alcanzable por la secuencia $W_I = \sigma_1 \sigma_2 \cdots \sigma_s \in L_I(S_f)$ tal que:

$$M_0^{S_f} \xrightarrow{W_I} M_s^{S_f}$$

b) $\forall i$ existe $W_{S_i} = \alpha_1 \alpha_2 \cdots \alpha_k \in L_I(S_f)$ secuencia controlable tal que:

$$C \cdot \overrightarrow{W_{S_i}} = \Pi \cdot Q \cdot \overrightarrow{z_{i-1}}$$

entonces, para $N < \infty$

$$\forall i \quad \lim_{k_i \rightarrow N} \left\| \varphi(M_{l+k_i}^{S_f}) - \varphi'(M_i^{R_m}) \right\| = 0$$

Demostración. Se tiene que el sistema y modelo de referencia son:

$$S_f = \begin{cases} M_i^{S_f} = M_0^{S_f} + C\overline{v_{k_i-1}} \\ b_{k_i} = \varphi(M_{k_i}^{S_f}) \end{cases} \quad \text{and} \quad R_m = \begin{cases} Y_i^{R_m} = Y_0^{R_m} + Q\overline{z_{i-1}} \\ r_i = \varphi'(Y_i^{R_m}) \end{cases} \quad (3.5)$$

se supone que existe una función Π y dos secuencias W_I, W_{S_i} tales que satisfacen los incisos del teorema 4. La demostración se realizará en dos partes.

- Primeramente se muestra que la norma del error en $i = 0$ converge a cero:

Usando el hecho de la existencia de la secuencia W_I tal que:

$$M_0^{S_f} \xrightarrow{W_I} M_s^{S_f}$$

y $\Pi \cdot M_0^{R_m} = M_s^{S_f}$, en la definición del error, se tiene

$$\mathcal{E}_0 = \left\| \varphi(M_s^{S_f}) - \varphi'(M_0^{R_m}) \right\| \quad (3.6)$$

$$\mathcal{E}_0 = \left\| \varphi(\Pi \cdot M_0^{R_m}) - \varphi'(M_0^{R_m}) \right\|$$

ya que $\varphi \circ \Pi = \varphi'$, se concluye

$$\mathcal{E}_0 = \left\| \varphi'(M_0^{R_m}) - \varphi'(M_0^{R_m}) \right\| = 0$$

o equivalentemente:

$$\mathcal{E}_0 = \left\| \varphi(M_s^{S_f}) - \varphi'(M_0^{R_m}) \right\| = 0$$

por otro lado, como $M_s^{S_f} = M_0^{S_f} + C\overline{W_I}$, con $N = \text{longitud}(W_I)$, entonces:

$$\mathcal{E}_0 = \lim_{k \rightarrow N} \left\| \varphi(M_{0+k}^{S_f}) - \varphi'(M_0^{R_m}) \right\| = 0$$

- Una vez que el error es cero para $i = 0$ se mostrará que la norma del error para cada evento i posterior volverá a converger. Note que si R_m acepta un símbolo σ_{i+1} de entrada, entonces R_m alcanza un nuevo estado $M_{i+1}^{R_m}$, es decir $M_i^{R_m} \xrightarrow{\sigma_{i+1}} M_{i+1}^{R_m}$; y esto ocurre a partir del estado $M_N^{S_f} = M_s^{S_f}$. Por tanto, para cumplir con la parte $\forall i$, se debe probar que dada una nueva secuencia de entrada W_{S_i} controlable para S_f tal que $M_s^{S_f} \xrightarrow{W_{S_i}} M_{s+N'}^{S_f}$, el error entre $M_{s+N'}^{S_f}$ y $M_{i+1}^{R_m}$ es cero para el evento i . La demostración se realizará por inducción.

- Considere que $i = 1$, sea $M_1^{R_m}$ el estado que alcanza el modelo de referencia; tal que:

$$M_1^{R_m} = M_0^{R_m} + Q \overline{z_0} \quad (3.7)$$

Se tiene que el error para $i = 1$ es:

$$\mathcal{E}_1 = \left\| \varphi \left(M_s^{S_f} \right) - \varphi' \left(M_1^{R_m} \right) \right\| \quad (3.8)$$

por hipótesis existe $W_{S_0} \in L_I(S_f)$ **secuencia controlable** de longitud N' tal que:

$$C \cdot \overrightarrow{W_{S_0}} = \Pi \cdot Q \cdot \overrightarrow{z_0} \quad (3.9)$$

al aplicar W_{S_0} al sistema S_f se alcanza el estado

$$M_{s+N'}^{S_f} = M_s^{S_f} + C\overrightarrow{W_{S_0}} \quad (3.10)$$

la ecuación del error para el nuevo estado del sistema S_f es:

$$\mathcal{E}_1 = \left\| \varphi \left(M_{s+N'}^{S_f} \right) - \varphi' \left(M_1^{R_m} \right) \right\| \quad (3.11)$$

substituyendo (3.10) y (3.7) en (3.11):

$$\mathcal{E}_1 = \left\| \varphi \left(M_s^{S_f} + C\overrightarrow{W_{S_0}} \right) - \varphi' \left(M_0^{R_m} + Q \overrightarrow{z_0} \right) \right\| \quad (3.12)$$

además, por hipótesis se tiene que $\varphi \circ \Pi = \varphi'$, entonces

$$\begin{aligned} \mathcal{E}_1 &= \left\| \varphi \left(M_s^{S_f} + C\overrightarrow{W_{S_0}} \right) - \varphi \circ \Pi \left(M_0^{R_m} + Q \overrightarrow{z_0} \right) \right\| \\ \mathcal{E}_1 &= \left\| \varphi \left(M_s^{S_f} + C\overrightarrow{W_{S_0}} \right) - \varphi \left(\Pi M_0^{R_m} + \Pi \cdot Q \cdot \overrightarrow{z_0} \right) \right\| \end{aligned}$$

haciendo algunas manipulaciones,

$$\mathcal{E}_1 = \left\| \varphi \left(M_s^{S_f} \right) + \varphi \left(C\overrightarrow{W_{S_0}} \right) - \varphi \left(\Pi M_0^{R_m} \right) - \varphi \left(\Pi \cdot Q \overrightarrow{z_0} \right) \right\|$$

de (3.9) y de la hipótesis $\Pi M_0^{R_m} = M_s^{S_f}$ se concluye que:

$$\mathcal{E}_1 = \left\| \left[\varphi \left(M_s^{S_f} \right) - \varphi \left(\Pi M_0^{R_m} \right) \right] + \left[\varphi \left(C \overrightarrow{W_{S_0}} - \Pi \cdot Q \cdot \overrightarrow{z_0} \right) \right] \right\| = 0$$

por otro lado como $M_{s+N'}^{S_f} = M_s^{S_f} + C\overrightarrow{W_{S_0}}$, y si $N' = \text{longitud}|W_{S_0}|$, entonces:

$$\mathcal{E}_1 = \lim_{k \rightarrow N'} \left\| \varphi \left(M_{s+k}^{S_f} \right) - \varphi' \left(M_1^{R_m} \right) \right\| = 0$$

– Suponiendo que

$$\mathcal{E}_n = \lim_{k \rightarrow N} \left\| \varphi \left(M_{l+k}^{S_f} \right) - \varphi' \left(M_n^{R_m} \right) \right\| = 0$$

es cierto para $i = n$, donde $M_{l+N}^{S_f} = M_l^{S_f} + C\overrightarrow{W_{S_n}} \doteq M_q^{S_f}$, con $N = \text{longitud}(W_{S_n})$. Se mostrará que la norma del error también converge a cero para $i = n + 1$.

Sea $M_{n+1}^{R_m}$ el estado que alcanza el modelo de referencia; tal que:

$$M_{n+1}^{R_m} = M_n^{R_m} + Q \cdot \vec{z}_n \quad (3.13)$$

El error en este caso es:

$$\mathcal{E}_{n+1} = \left\| \varphi \left(M_q^{S_f} \right) - \varphi' \left(M_{n+1}^{R_m} \right) \right\| \quad (3.14)$$

por hipótesis existe $W_{S_{n+1}} \in L_I(S_f)$ **secuencia controlable** de longitud N' tal que:

$$C \cdot \overrightarrow{W_{S_{n+1}}} = \Pi \cdot Q \cdot \vec{z}_n \quad (3.15)$$

al aplicar $W_{S_{n+1}}$ al sistema S_f se alcanza el estado

$$M_{q+N'}^{S_f} = M_q^{S_f} + C \overrightarrow{W_{S_{n+1}}} \quad (3.16)$$

la ecuación del error para el nuevo estado del sistema es:

$$\mathcal{E}_{n+1} = \left\| \varphi \left(M_{q+N'}^{S_f} \right) - \varphi' \left(M_{n+1}^{R_m} \right) \right\| \quad (3.17)$$

substituyendo (3.16) y (3.13) en (3.17):

$$\mathcal{E}_{n+1} = \left\| \varphi \left(M_q^{S_f} + C \overrightarrow{W_{S_{n+1}}} \right) - \varphi' \left(M_n^{R_m} + Q \cdot \vec{z}_n \right) \right\| \quad (3.18)$$

además, por hipótesis se tiene que $\varphi \circ \Pi = \varphi'$, y que $\Pi M_n^{R_m} = M_q^{S_f}$ entonces

$$\begin{aligned} \mathcal{E}_{n+1} &= \left\| \varphi \left(M_q^{S_f} + C \overrightarrow{W_{S_{n+1}}} \right) - \varphi \circ \Pi \left(M_n^{R_m} + Q \cdot \vec{z}_n \right) \right\| \\ \mathcal{E}_{n+1} &= \left\| \varphi \left(M_q^{S_f} + C \overrightarrow{W_{S_{n+1}}} \right) - \varphi \left(\Pi M_n^{R_m} + \Pi \cdot Q \cdot \vec{z}_n \right) \right\| \end{aligned}$$

haciendo algunas manipulaciones,

$$\mathcal{E}_{n+1} = \left\| \varphi \left(M_q^{S_f} \right) + \varphi \left(C \overrightarrow{W_{S_{n+1}}} \right) - \varphi \left(\Pi M_n^{R_m} \right) - \varphi \left(\Pi \cdot Q \cdot \vec{z}_n \right) \right\|$$

introduciendo (3.15) se concluye que:

$$\mathcal{E}_{n+1} = \left\| \left[\varphi \left(M_q^{S_f} \right) - \varphi \left(\Pi M_n^{R_m} \right) \right] + \left[\varphi \left(C \cdot \overrightarrow{W_{S_{n+1}}} - \Pi \cdot Q \cdot \vec{z}_n \right) \right] \right\| = 0$$

por otro lado como $M_{q+N'}^{S_f} = M_q^{S_f} + C \overrightarrow{W_{S_{n+1}}}$, y si $N' = \text{longitud}(W_{S_{n+1}})$, entonces:

$$\mathcal{E}_{n+1} = \lim_{k \rightarrow N'} \left\| \varphi \left(M_{q+k}^{S_f} \right) - \varphi' \left(M_{n+1}^{R_m} \right) \right\| = 0$$

Entonces

$$\forall i, \quad \mathcal{E}_i = \lim_{k \rightarrow N} \left\| \varphi \left(M_{l+k}^{S_f} \right) - \varphi' \left(M_i^{R_m} \right) \right\| = 0$$

■ Es importante notar que N es finita porque el número de estados de S_f y R_m es finito, también si el tamaño de W_I o W_S es mayor que el número de estados, entonces por el lema de bombeo, una palabra más corta es alcanzable desde algún otro estado.

Pueden resaltarse algunas observaciones adicionales:

Nota 3.1 Cada columna de $\Pi \cdot Q$ debe estar en la imagen de C , más aún cada columna de $\Pi \cdot Q$ debe ser una combinación lineal de enteros positivos de C . En realidad:

$$\begin{aligned} C\overrightarrow{W_{S_i}} - \Pi \cdot Q\overrightarrow{z_{i-1}} &= 0 \\ C\overrightarrow{W_{S_i}} &= \Pi \cdot Q\overrightarrow{z_{i-1}} \end{aligned}$$

Como $\overrightarrow{z_{i-1}}$ es el disparo de una transición, entonces la parte derecha de la ecuación representa las diferentes columnas de $\Pi \cdot Q$.

Nota 3.2 Si el teorema 4 se mantiene, entonces se dice que $L_{O_{\Phi'}}(R_m)$ es controlable con respecto a $L_{O_{\Phi'}}(S_f)$. Esto se debe a que cualquier palabra en $L_{O_{\Phi'}}(R_m)$ debe ser controlable con respecto a $L_{O_{\Phi'}}(S_f)$.

El siguiente corolario del teorema 4, indica cómo se construye la secuencia que genera el controlador para confinar el comportamiento de un SED dentro del comportamiento de otro SED .

Corolario 3.1 Si el teorema 4 se mantiene, entonces un controlador para confinar el comportamiento de S_f dentro del comportamiento de R_m es la función H dada en (3.3).

Demostración. Como $\exists W_I$ tal que:

$$M_0^{S_f} \xrightarrow{W_I} M_s^{S_f} \text{ y } \lim_{k_i \rightarrow N} \left\| \varphi \left(M_{0+k_i}^{S_f} \right) - \varphi' \left(M_i^{R_m} \right) \right\| = 0 \quad (3.19)$$

entonces la función H puede calcularse como:

$$\begin{aligned} H \left(M_0^{S_f}, M_0^{R_m} \right) &= u_0^0 \\ H \left(M_1^{S_f}, M_0^{R_m} \right) &= u_1^0 \\ &\vdots \\ H \left(M_{s-1}^{S_f}, M_0^{R_m} \right) &= u_{s-1}^0 \end{aligned}$$

Donde $W_I = u_0^0 u_1^0 \cdots u_{s-1}^0$, además ocurre para cualquier $M_i^{R_m}$

$$\begin{aligned} H \left(M_q^{S_f}, M_i^{R_m} \right) &= u_q^i \\ H \left(M_{q+1}^{S_f}, M_i^{R_m} \right) &= u_{q+1}^i \\ H \left(M_{q+2}^{S_f}, M_i^{R_m} \right) &= u_{q+2}^i \\ &\vdots \\ H \left(M_{q+N-1}^{S_f}, M_i^{R_m} \right) &= u_{q+N-1}^i \end{aligned}$$

donde $W_{S_i} = u_q^i u_{q+1}^i u_{q+2}^i \cdots u_{q+N-1}^i$. Ahora H es una función. Suponga que no es así, entonces existen dos secuencias de control u_i^i, u_x^k tales que $H(M_i^{Sf}, M_i^{Rm}) = u_i^i \neq H(M_x^{Sf}, M_k^{Rm}) = u_x^k$ con $M_i^{Sf} = M_x^{Sf}, M_i^{Rm} = M_k^{Rm}$. Como $\varphi'(M_i^{Rm}) = \varphi'(M_k^{Rm})$ entonces el sistema debe alcanzar la misma salida en ambos casos. Por lo que el mismo símbolo de salida puede elegirse, y $u_i^i = u_x^k$, lo que es una contradicción. Por lo tanto H es una función y es el controlador deseado. ■

3.5 Conclusiones

En este capítulo se introdujeron los conceptos necesarios para establecer relaciones de orden en las salidas de una *RPI*. Mediante la aplicación de la relación de orden, se define un nuevo lenguaje de entrada y salida. Con este resultado se establece el concepto de error entre las salidas de dos sistemas. Con la introducción de Π se logra relacionar las funciones de orden de ambos sistemas, al aplicarla a los marcados del modelo de referencia se logra aumentar las dimensiones de éste modelo a la dimensión del sistema.

Las condiciones para el seguimiento de modelo se reducen a encontrar una secuencia de eventos que lleve del marcado actual del sistema al marcado (aumentado) del modelo de referencia. A partir de este marcado se establece una secuencia para cada evolución de la especificación que permita que el error sea siempre cero. Este resultado permite definir un controlador mediante la función H ; esta función establece la ley de control que no depende únicamente de las palabras de entrada, sino que también depende del marcado de la red y de la especificación. Esta característica es la que permite hacer que el sistema siga a éste modelo.

Capítulo 4

Algoritmo para el seguimiento de modelo

Resumen: En este capítulo se presenta un algoritmo que construye la función H . Aunque esta función parece ser muy grande, en realidad no es así, ya que se aprovecha el hecho de que sólo hay que analizar las columnas de $\Pi \cdot Q$.

4.1 Introducción

En este capítulo se presenta un algoritmo para encontrar una ley de control que permita que un sistema pueda seguir a una especificación.

Este algoritmo consta de tres fases. En la fase de inicialización se piden las funciones de ordenación φ y φ' de ambos sistemas con el fin de hacer compatibles las salidas de los sistemas, y la función Π que mantiene las condiciones del teorema 4, si no existe esta función entonces en el sistema a controlar no existe la secuencia de disparos para seguir el comportamiento del modelo de referencia. Una vez terminada la fase de inicialización, la segunda fase del algoritmo inicia calculando la primera secuencia de disparos que hace el error igual a cero para el instante inicial en el modelo de referencia.

La fase tres comienza cuando la especificación cambia de estado y suministra una secuencia controlable W_{S_i} al sistema que permite nuevamente que el error a la salida entre ambos sistemas sea cero.

4.2 El algoritmo

Algoritmo 4.1 *Controlador para el seguimiento de modelo usando la función H^1*
/Fase de inicialización/

1. *Entradas:*

(a) *Las funciones φ , φ' y Π ;*

$Y_i^{R_m} \leftarrow Y_0^{R_m}$; *el marcado actual del modelo de referencia*

$M_i^{S_f} \leftarrow M_0^{S_f}$; *el marcado actual del sistema a controlar*

Salidas:

(a) *secuencia* $\leftarrow \emptyset$

/Fase dos/

Paso 1. $M_s^{S_f} \leftarrow \Pi \cdot Y_i^{R_m}$

Paso 2. *Tome la secuencia W_I controlable con respecto a $M_s^{S_f}$*

Paso 3. *Dispare W_I en S_f .*

/Fase tres/

a. *Mientras ($\mathcal{E}_i = 0$) hacer;*

continue ciclado en este mientras

fin mientras

b. $M_s^{S_f} \leftarrow \Pi \cdot Y_i^{R_m}$;

¹Note que la especificación puede ser cambiada en cualquier estado del sistema, también que el modelo de referencia no cambiara de estado hasta que el error entre las salidas de ambos sistemas sea 0.

- c. Tome la secuencia W_{S_i} controlable con respecto a S_f tal que $C \overline{W}_{S_i} = \Pi \cdot Q \overline{z}_{i-1}$
- d. Dispense la secuencia W_{S_i} en S_f .
- e. Ir a paso a.

4.3 Ejemplo

Ejemplo 4.1 Considere el ejemplo del sistema de manufactura mostrado en la figura 2.11, y modelado con la RPI mostrada en la figura 2.12.

Suponga que la especificación que está realizando el sistema de manufactura es:

“si hay pieza en almacén 1, entonces el robot toma una pieza del almacén 1 y lo deposita en la máquina 1, una vez que la máquina 1 termina, la pieza es pasada al almacén 2”.

Suponga que el estado del sistema es el siguiente:

“no hay pieza en almacén 1, el robot está disponible, la máquina 1 está procesando pieza, la máquina 2 esta ociosa, en almacén 2 no hay pieza”.

El sistema tiene como marcado actual el mostrado en la figura 4.2 y se denomina como $M_i^{S_f}$

En estas condiciones está el sistema cuando la especificación cambia a:

“si hay pieza en almacén 1, entonces el robot toma una pieza del almacén 1 y lo deposita en la máquina 2, una vez que la máquina 2 termina, la pieza es pasada al almacén 2” (vea la figura 4.1)

Entonces, las piezas deben ser procesadas por la máquina 2.

Los modelos del sistema y de la especificación son RPI vivas, binarias y de etiquetado distinto.

El modelo para el sistema es S_f :

$$M_{k+1}^{S_f} = M_0^{S_f} + C \overline{v}_k$$

$$M_{k+1}^{S_f} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 1 & 1 \end{bmatrix} \overline{v}_k$$

$$b_k = \varphi \left(M_k^{S_f} \right)$$

$$b_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \left(M_k^{S_f} \right)$$

El modelo de la especificación es R_m :

$$Y_i^{R_m} = Y_0^{R_m} + Q \overrightarrow{z_{i-1}}$$

$$Y_i^{R_m} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 1 \end{bmatrix} \overrightarrow{z_{i-1}}$$

$$r_i = \varphi' (Y_i^{R_m})$$

$$r_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} (Y_i^{R_m})$$

$$r_i = I (Y_i^{R_m}) = Y_i^{R_m}$$

La función Π que satisface el teorema 4 es:

$$\Pi = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

El algoritmo se comporta de la siguiente forma²:

Inicialización

1. La salida de ambos sistemas es:

$$M_l^{S_f} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}, Y_0^{R_m} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

²Recuerde que existe la función de igualdad.

2. Las funciones φ , φ' y Π son:

$$\varphi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad \varphi' = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \Pi = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

donde:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \circ \begin{matrix} \varphi \circ \Pi = \varphi' \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \end{matrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Fin de Inicialización

Fase dos

Paso 1. Calcular a qué mercado se quiere llevar al sistema de manufactura, esto es, obtener

$$M_s^{Sf} = \Pi \cdot Y_0^{Rm}$$

$$M_s^{Sf} = \Pi \cdot Y_0^{Rm} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

Paso 2. Para este caso $W_I = t_1$

Paso 3. Disparar la secuencia W_I . Al hacer esto se obtiene que:

$$\begin{aligned}
 \mathcal{E}_0 &= \left\| \varphi \left(M_s^{S_f} \right) - \varphi' \left(Y_0^{R_m} \right) \right\| \\
 &= \left\| \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right\| \\
 &= \left\| \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right\| \\
 &= 0
 \end{aligned}$$

Fin de la Fase dos

Fase tres

a. **Mientras** (el error sea cero) hacer ;

continúe ciclado en este mientras

fin mientras

Como la salida de ambos sistemas usando la función de igualación es la misma, se quedaría en este paso.

Si en este momento es introducido al modelo de referencia la entrada y , entonces ejecutaría **b** del algoritmo, dando como resultado:

b. Para este caso se obtuvo que $W_{S_i} = t_2$

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 \\ 0 & 0 & 0 & 0 & -1 & -1 & 1 \end{bmatrix} C\overline{W}_{S_i} = \Pi Q\overline{z_{i-1}}$$

$$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \\ -1 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

c. Dispense \overline{W}_{S_i} ; entonces los marcados de los sistemas son:

$$M_{s+1}^{S_f} = M_s^{S_f} + C\overline{W}_{S_i}$$

$$M_{s+1}^{S_f} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$M_{s+1}^{S_f} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

y

$$\begin{aligned}
 Y_1^{Rm} &= Y_0^{Rm} + Q \vec{z}_0 \\
 Y_1^{Rm} &= \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 0 \\ -1 & & & \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \\
 Y_1^{Rm} &= \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \\
 Y_1^{Rm} &= \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}
 \end{aligned}$$

El error es:

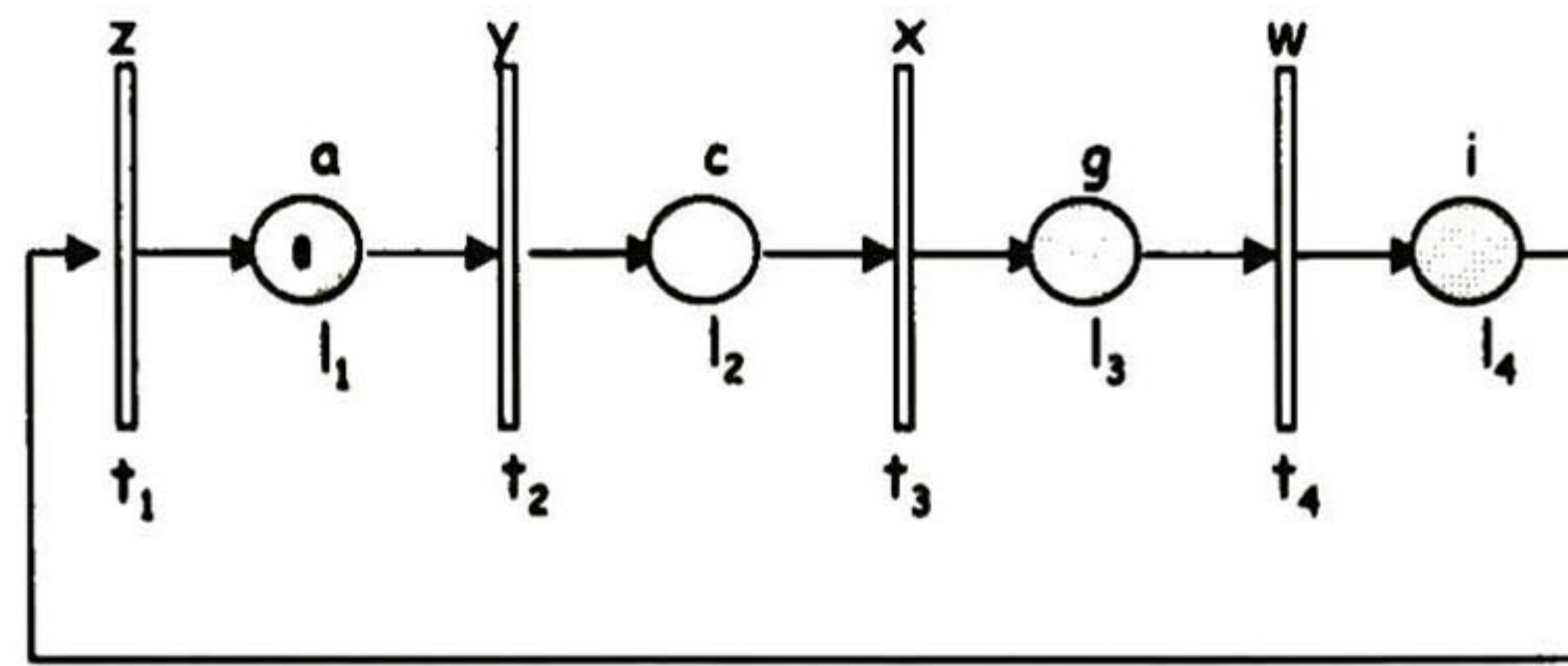
$$\begin{aligned}
 \left\| \varphi \left(M_{s+1}^{Sf} \right) - \varphi' \left(Y_1^{Rm} \right) \right\| &= \mathcal{E}_1 \\
 \left\| \begin{array}{c} \varphi \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} - \varphi' \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \end{array} \right\| &= \mathcal{E}_1 \\
 \left\| \begin{array}{c} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \end{array} \right\| &= 0
 \end{aligned}$$

1. Ir a **a**;**fin si***Fin de Fase tres*

Al volver a **a** se tiene que la salida de ambos sistemas usando las funciones φ y φ' es la misma, se quedaría en este paso. Así continuaría funcionando hasta que una nueva especificación sea introducida.

La figura 4.3 muestra el esquema del seguimiento de modelo para este ejemplo.

Ejemplo 4.2 Retomemos el sistema "Coche eléctrico" que se muestra en la figura 4.4. Este sistema consiste de un coche eléctrico, dos sensores y dos botones. El coche se mueve de izquierda a derecha activando los sensores (vea la figura 4.5). Como puede observar en la figura 4.6 el sistema "Coche eléctrico" cuenta con lugares no medibles (l_1, l_2, l_3, l_4, l_6) y cuatro transiciones incontrolables.



En donde:

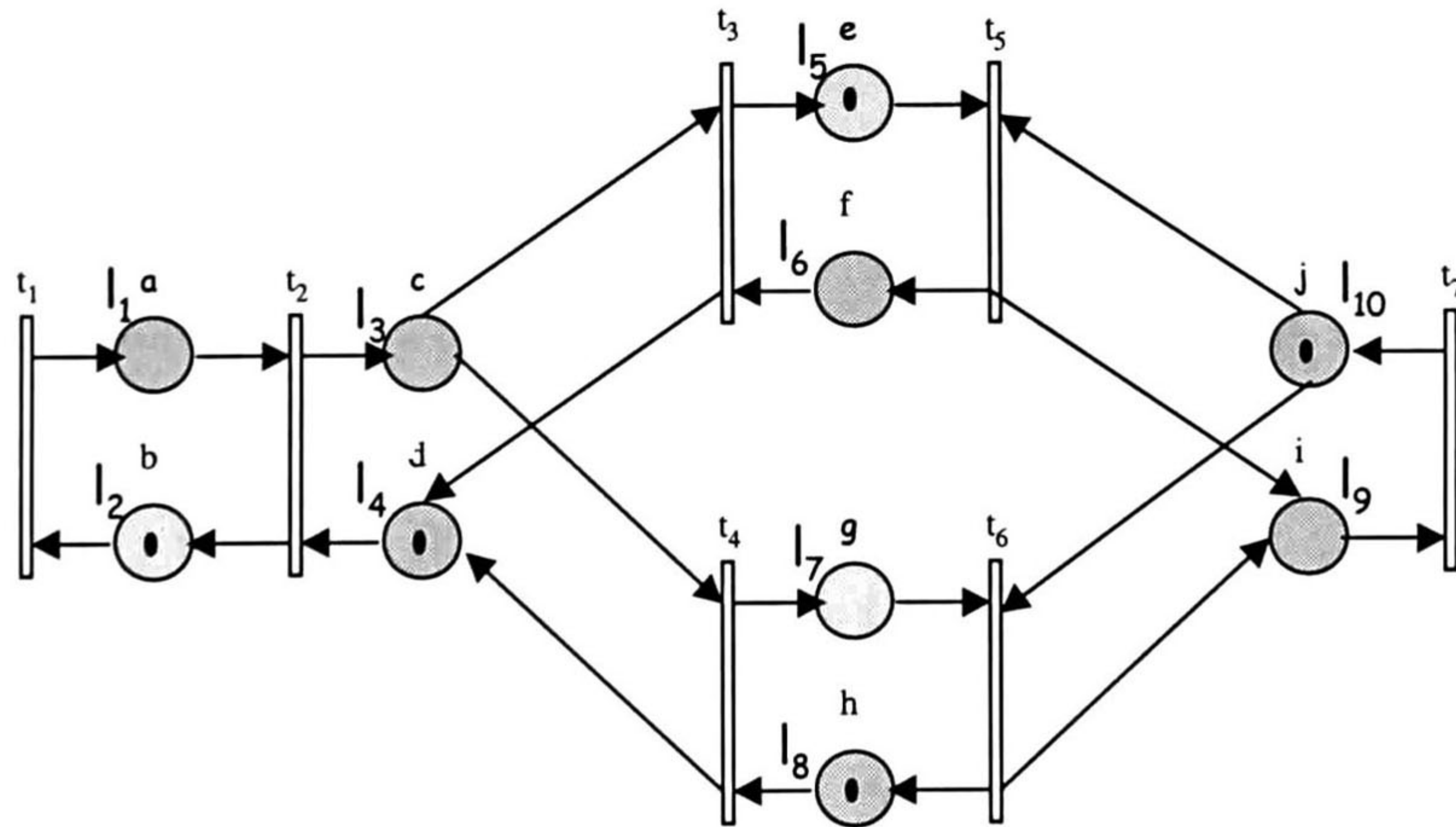
Señales de los sensores

- a señal que indica que hay pieza en almacén.
- c señal que indica que el robot tiene pieza.
- g señal que indica que la máquina 2 está procesando la pieza.
- i señal que indica que se almacenó una pieza en el almacén 2.

Eventos controlables

- z colocar pieza en almacén 1.
- y robot toma pieza.
- x robot coloca pieza en máquina 1.
- w almacenar pieza en almacén 2.

Figura 4.1: Especificación.



$$M_f^{S_f} = [0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 0 \ 0 \ 1 \ 0 \ 1]^T$$

Figura 4.2: Marcado actual del sistema de manufactura.

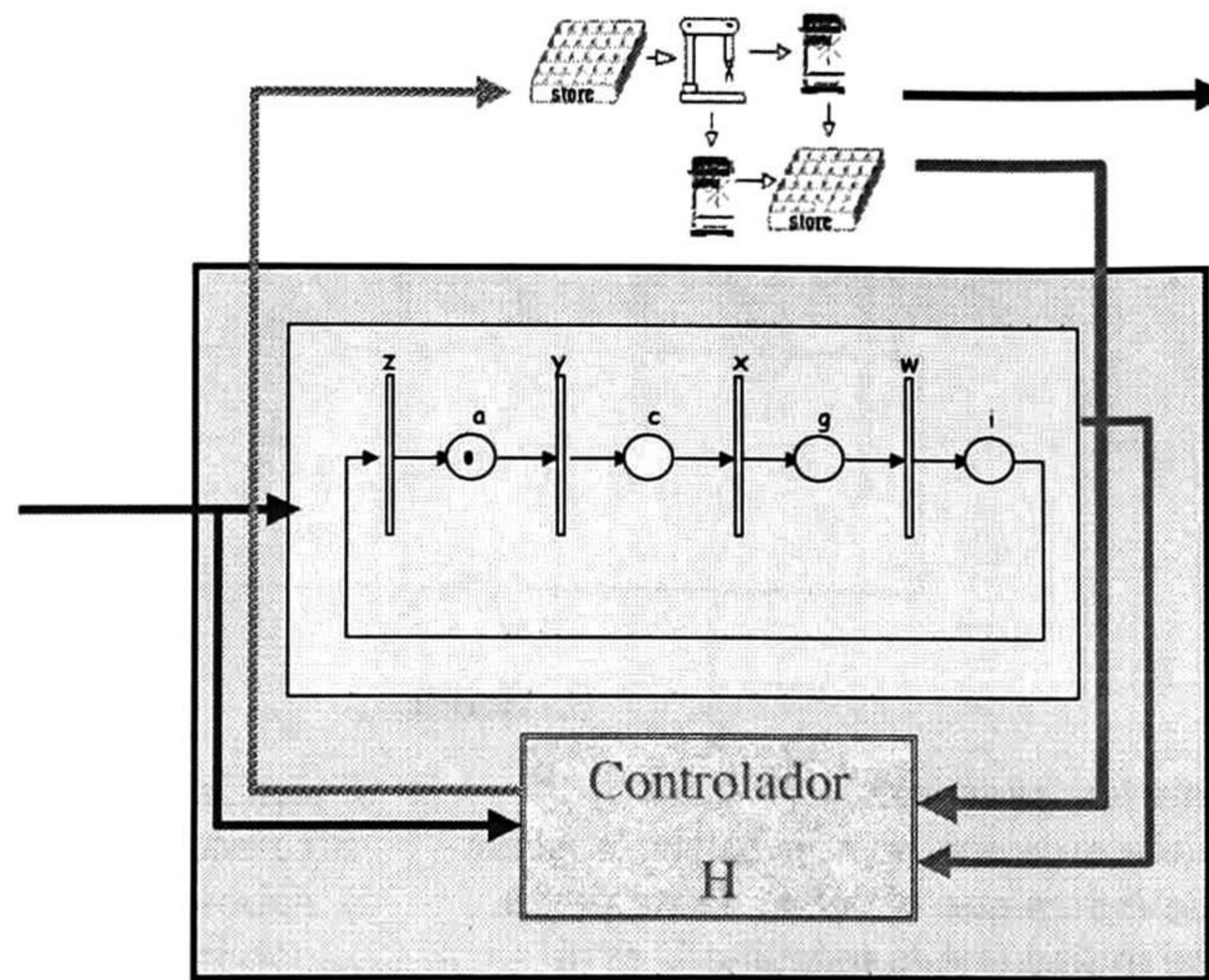


Figura 4.3: Esquema para el problema de seguimiento de modelo para el “*Sistema de manufactura*”.

Suponga que la especificación es:

“mover el coche a la derecha, una vez que el coche activa el sensor 2 mover coche a la izquierda, cuando el sensor 1 es activado mover el coche a la izquierda” (vea la figura 4.7).

El sistema tiene como marcado actual el mostrado en la figura 4.6 y se denomina como $M_l^{S_f}$.

Los modelos del sistema y de la especificación son *RPI* vivas, binarias y de etiquetado distinto.

El modelo para el sistema es S_f :

$$M_{k+1}^{S_f} = M_0^{S_f} + C \vec{v}_k$$

$$M_{k+1}^{S_f} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \vec{v}_k$$

$$b_k = \varphi(M_k^{S_f})$$

$$b_k = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} (M_k^{S_f})$$

El modelo de la especificación es R_m :

$$Y_i^{R_m} = Y_0^{R_m} + Q \overrightarrow{z_{i-1}}$$

$$Y_i^{R_m} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \overrightarrow{z_{i-1}}$$

$$r_i = \varphi' (Y_i^{R_m})$$

$$r_i = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} (Y_i^{R_m})$$

$$r_i = I (Y_i^{R_m})$$

$$r_i = Y_i^{R_m}$$

La función Π que satisface el teorema 4 es:

$$\Pi = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 0 \\ 1 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

El algoritmo se comporta de la siguiente forma:

Inicialización

1. La salida de ambos sistemas es:

$$M_l^{S_f} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, Y_0^{R_m} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

2. Las funciones φ , φ' y Π son:

$$\varphi = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \varphi' = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \Pi = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 0 \\ 1 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

donde:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \circ \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 0 \\ 1 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Fin de Inicialización

Fase dos

Paso 1. Calcular a qué marcado se quiere llevar al sistema de manufactura, esto es, obtener

$$M_s^{S_f} = \Pi Y_0^{R_m}$$

$$M_s^{S_f} = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 0 \\ 1 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Para este caso $W_I = t_1 t_5 t_7 t_2$ y la palabra de entrada es $D\bar{D}$

Paso 3. Disparar la secuencia W_I . Al hacer esto se obtiene que:

$$\begin{aligned} \mathcal{E}_0 &= \left\| \varphi \left(M_s^{S_f} \right) - \varphi' \left(Y_0^{R_m} \right) \right\| \\ &= \left\| \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right\| \\ &= \left\| \begin{bmatrix} 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right\| \\ &= 0 \end{aligned}$$

Fin de la Fase dos

Fase tres

- a. **Mientras** (el error sea cero) hacer ;
 continúe ciclado en este mientras
fin mientras

Como la salida de ambos sistemas usando la función de igualación es la misma, se quedaría en este paso.

Si en este momento es introducido al modelo de referencia la entrada e_2 , entonces ejecutaría **b** del algoritmo, dando como resultado:

- b. Para este caso se obtuvo que $W_{S_i} = t_3 t_8 t_6 t_4$ y la palabra de entrada es $\bar{I}\bar{I}$

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ -1 \end{bmatrix} = \Pi \cdot Q \bar{z}_{i-1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -1 & 1 \\ 0 & 0 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ -1 \end{bmatrix}$$

- c. Dispare \bar{W}_{S_i} ; entonces los marcados de los sistemas son:

$$M_{s+1}^{S_f} = M_s^{S_f} + C\bar{W}_{S_i}$$

$$M_{s+1}^{S_f} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ -1 \end{bmatrix}$$

$$M_{s+1}^{S_f} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

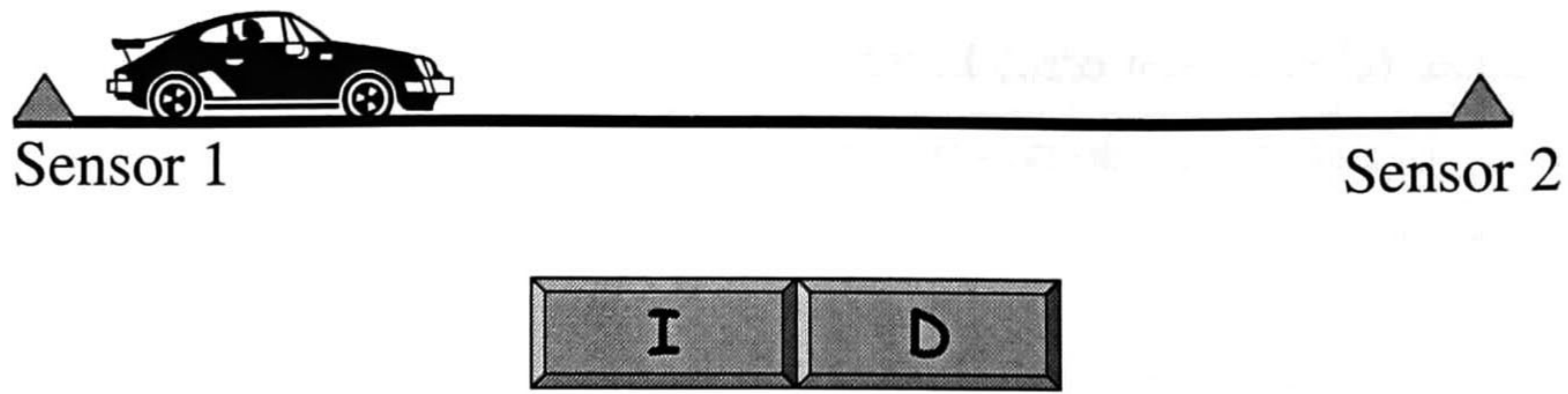


Figura 4.4: Sistema "Coche eléctrico".

y

$$\begin{aligned}
 Y_1^{R_m} &= Y_0^{R_m} + Q \vec{z}_0 \\
 Y_1^{R_m} &= \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
 Y_1^{R_m} &= \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ -1 \end{bmatrix} \\
 Y_1^{R_m} &= \begin{bmatrix} 1 \\ 0 \end{bmatrix}
 \end{aligned}$$

El error es:

$$\left\| \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right\| = \mathcal{E}_1$$

$$\left\| \begin{bmatrix} 1 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right\| = 0$$

Ir a a;

fin si

Fin de Fase tres

Al volver a a se tiene que la salida de ambos sistemas usando las funciones φ y φ' es la misma, se quedaría en este paso. Así continuaría funcionando hasta que una nueva especificación sea introducida.

La figura 4.8 muestra el esquema del seguimiento de modelo para este ejemplo.

4.4 Conclusiones

En este capítulo es dado un algoritmo que permite encontrar una ley de control para que un sistema pueda seguir una especificación. Éste necesita las funciones de ordenamiento y la

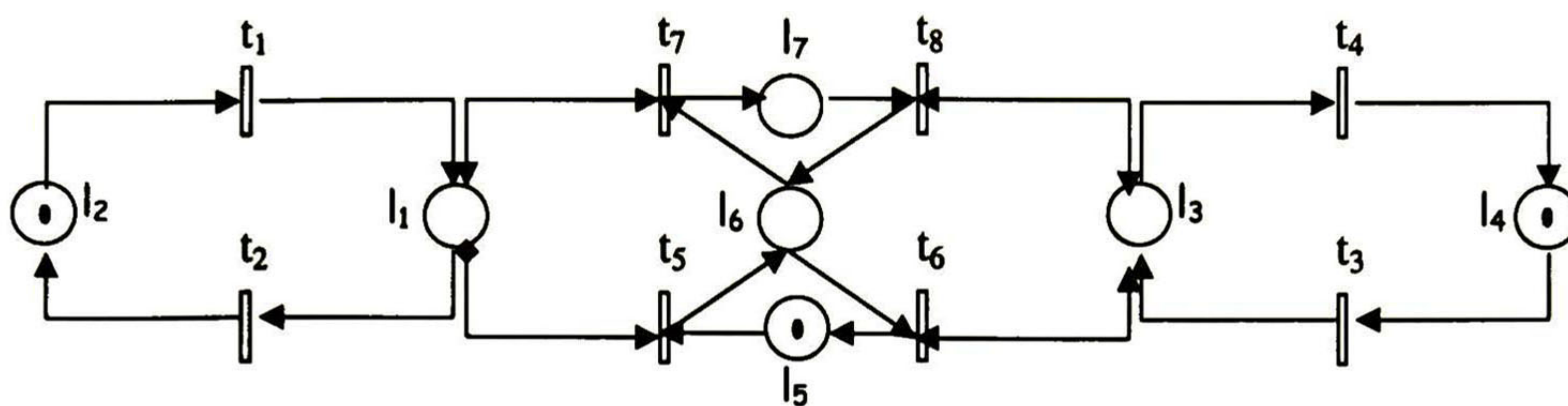


Figura 4.5: RPI del sistema "Coche eléctrico".

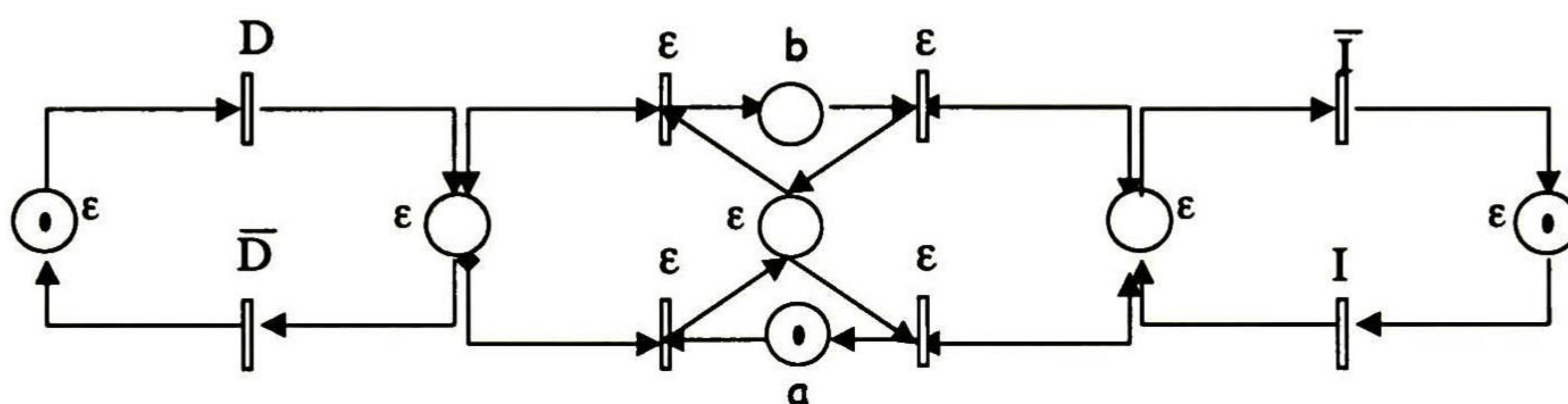


Figura 4.6: RPI del sistema "Coche eléctrico".

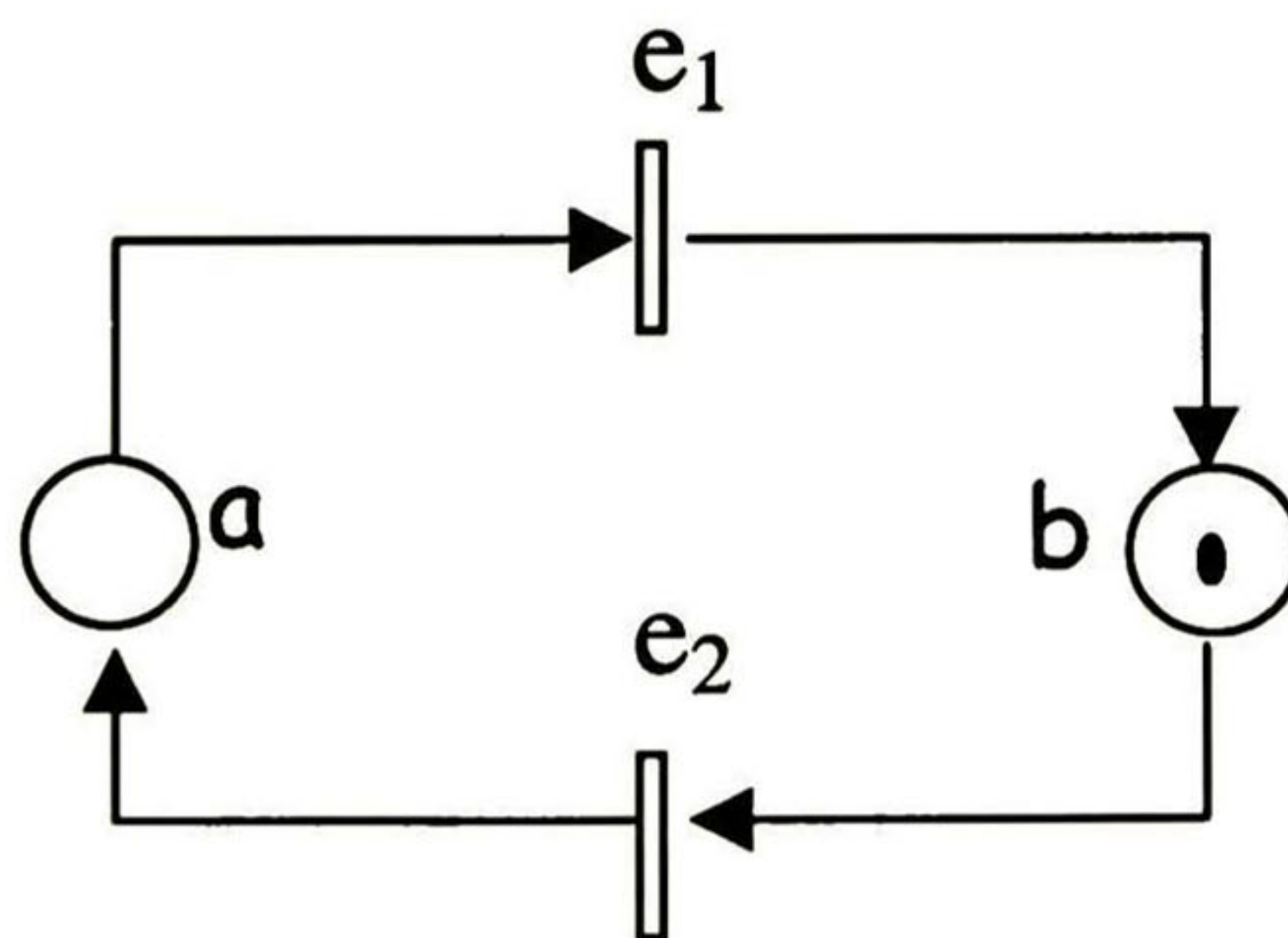


Figura 4.7: Especificación

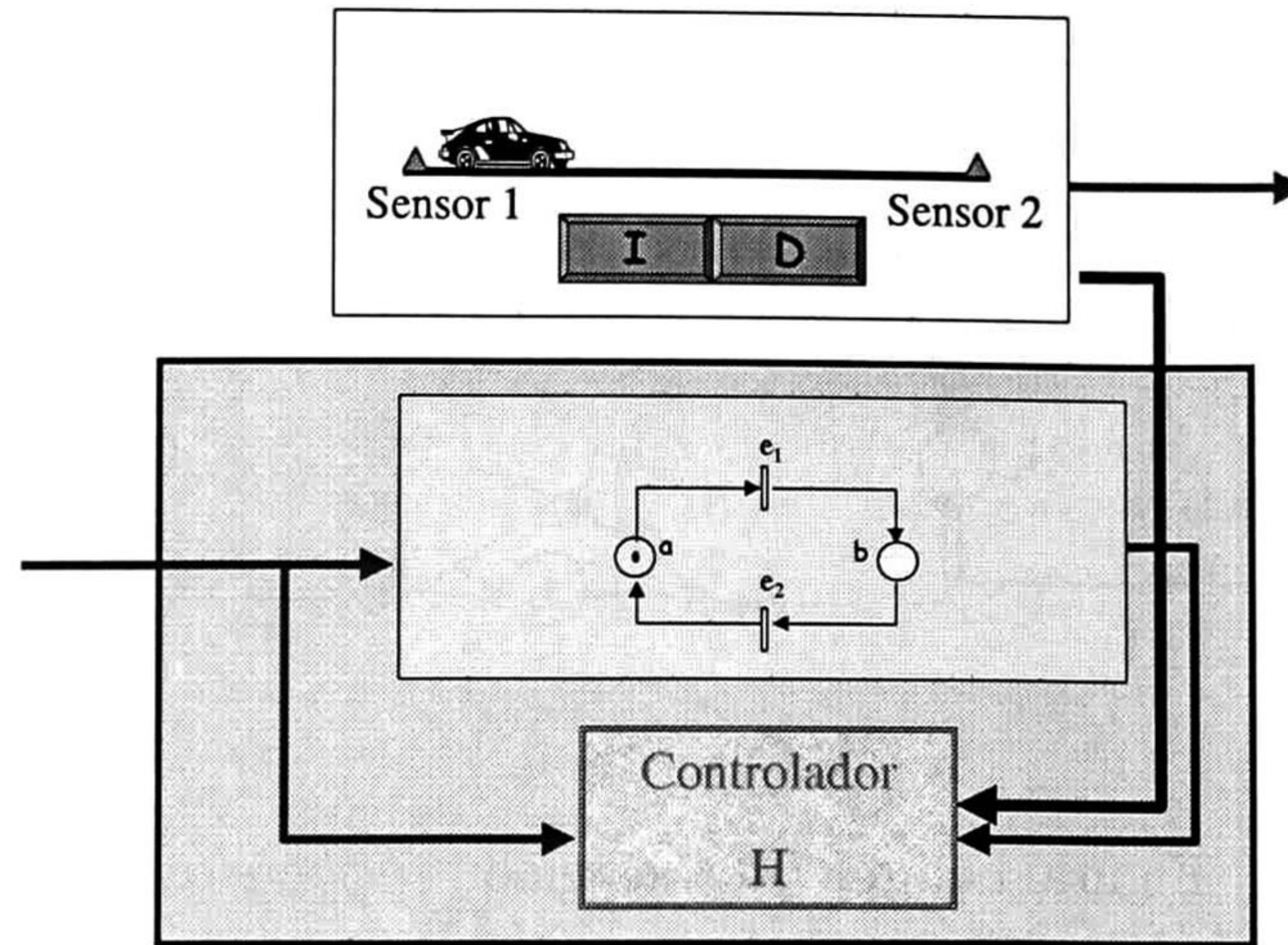


Figura 4.8: Esquema para el problema de seguimiento de modelo para el sistema “Coche eléctrico”.

función Π ; con esta información se logra calcular un marcado deseado en el sistema a partir del marcado actual de la especificación. Este algoritmo está dividido en tres partes. En la última fase (fase tres) se usa el hecho de que $\Pi \cdot Q$ es una combinación lineal de C (vea la nota 3.1), se obtiene que cada disparo de transiciones en la especificación tiene una secuencia de disparos equivalente en el sistema, por lo que sólo se debe disparar esta secuencia para que el error sea cero. Ya que esta secuencia puede contener transiciones incontrolables (las cuales se pueden disparar en cualquier momento) no es posible saber cuánto tiempo se tardará el sistema en alcanzar al modelo de la especificación, sin embargo, se sabe que esta secuencia es finita y la complejidad del algoritmo es de orden lineal.

Capítulo 5

Conclusiones y trabajo futuro

5.1 Conclusiones

El trabajo desarrollado en la presente memoria se centra en el estudio del problema de seguimiento de modelo en los Sistemas de Eventos Discretos. Los sistemas han sido modelados como *RPI* para dotar al modelo de la propiedad de interpretación, también se supone que poseen las propiedades de acotamiento, vivacidad y observabilidad.

Como primer resultado se definió del problema de seguimiento de modelo en Sistemas de Eventos Discretos, ya que hasta ahora las teorías de control para Sistemas de Eventos Discretos no abarcan este caso.

Otro resultado importante son las definiciones de controlabilidad con respecto a un marcado y a un conjunto de secuencias, ya que se hace una extensión a los trabajos desarrollados sobre esta propiedad.

También, se dan condiciones suficientes para determinar cuándo un sistema puede seguir a un modelo de referencia y se presenta el algoritmo controlador que elimina el cálculo del supervisor.

Estos resultados respresentan una aportación a la teoría de control en el campo de los Sistemas de Eventos Discretos modelados en Redes de Petri interpretadas.

Las ventajas de este control son:

- No se necesita que el sistema y la especificación partan del mismo estado.
- Existe sólo un controlador para todas las posibles especificaciones.
- Permite que la especificación evolucione independientemente al sistema.
- Ya que cada disparo de transiciones de la especificación tiene una secuencia equivalente en el sistema, se obtiene entonces que la especificación podría sumergirse en el sistema.

Las limitaciones de este control son:

- No se tiene aún un algoritmo para el cálculo de la función Π .
- El disparo de secuencias con eventos incontrolables no puede determinarse cuanto tiempo tardará en llevarse a cabo, sólo se sabe que ocurrirá eventualmente.
- En este momento no es importante restringir la secuencia, sólo interesa que el sistema llegue a marcados específicos.

5.2 Trabajo Futuro

Las actividades que pueden realizarse a partir de los resultados obtenidos en la propuesta presentada en esta memoria, son las siguientes :

- Estudiar el problema de seguimiento de modelo cuando el sistema físico no es observable.
- Realizar un controlador basado en seguimiento de modelo que pueda ser comercializado. Y que incluya ayuda visual para introducir fácilmente las especificaciones.
- Extender las soluciones a *RPI* no binarias (para tener modelos más compactos y abarcar otro tipo de sistemas).
- Estudiar el caso en que el algoritmo controlador se pueda representar como una Red de Petri, con el fin de poder obtener propiedades del controlador; el lenguaje que genera, entre otros; ya que el controlador presentado sólo es un algoritmo que emite al sistema físico la ley de control adecuada.

Bibliografía

- [1] M. Abellanas y D. Lodaes, "*Matemática Discreta*", Publicada en México por: Macrobit Editores 1991.
- [2] Luis Isidro Aguirre Salas, Antonio Ramírez y Ofelia Begovich. "*Asymptotic Observers for Discrete Event Systems using Interpreted Petri nets*". International Symposium on Robotics and Automation. Saltillo, Coahuila, Cap. Discrete Event Systems. Noviembre 1998, pp. 365-369.
- [3] Cnet M. (tm)zveren y Alan S. Willsky. "*Observability of Discrete Event Dynamic Systems*". IEEE Transactions of Automatic Control, Vol. 35, No. 7, Julio 1990, pp. 797-806.
- [4] Jörg Desel and Javier Esparza, "*Free Choice Petri Nets*", Cambridge University Press 1995.
- [5] D. Y. Lee y F. DiCesare. "*FMS scheduling using Petri Nets and heuristic search*". En Proceedings, International Conference on Robotics and Automation, pp. 1057-1062, Nice, France, May 1992.
- [6] F. DiCesare, G. Harhalakis, J. M. Proth, M. Silva and F. B. Vernadat, "*Practice of Petri Nets in Manufacturing*", Chapman&Hall 1993.
- [7] Javier Esparza and Mogens Nielsen, "*Decidability Issues for Petri Nets*", BRICS Report Series (RS-94-8).
- [8] Alessandro Giua, "*Petri Nets as Discrete Event Models for Supervisory Control*", Tesis Doctoral, Rensselaer Polytechnic Institute. Troy, New York, 1992
- [9] John E. Hopcroft and Jeffrey D. Ullman, "*Introduction to Automata Theory, Languages, and Computation*", Addison-Wesley 1979.
- [10] Matthias Jantzen, "*Language Theory of Petri Nets*", Fachbereich Informatik University of Hamburg Rothenbaumchaussee 67 D-2000 Hamburg 13, Reporte Interno páginas 397-412.
- [11] Yong Li y W.M. Wonham. "*Controllability and Observability in the state-feedback Control of Discrete Event Systems*". Proceeding of the 27th Conference on Decision and Control. Austin, Texas, December 1998, pp. 203-208.
- [12] Luis Ernesto López Mellado, "*Introducción a las Redes de Petri*", Notas de la Facultad de Ciencias Físico- Matemáticas UANL. Octubre 1997

- [13] M.E. Meda & A. Ramírez, "*Identification in Discrete Event Systems*", IEEE-SMA 1998.
- [14] M.E. Meda Campaña, "*Identificación de Sistemas Dinámicos de Eventos Discretos utilizando Redes de Petri*", Tesis de Maestría, Laboratorio de Ingeniería Eléctrica y Ciencias de la Computación-Guadalajara. Guadalajara, Jalisco, México, Septiembre 1998.
- [15] T. Murata, "*Petri nets: Properties, analysis and applications*", Proceedings of the IEEE, vol. 77, p. 541-580, april 1989.
- [16] Ogata Katsuhiko, "*Ingeniería de Control Moderna*", Prentice-Hall, 1998.
- [17] P. J. Ramadge and W. M. Wonham, "*Supervisory control of a class of discrete event process*", SIAM J: Cont. Optimiz., vol. 25, p. 206-230, Jan. 1987.
- [18] Manuel Silva, "*Las Redes de Petri en la Automática y la Informática*", Editorial AC 1985.
- [19] W. M. Wonham and P. J. Ramadge, "*Modular Supervisory Control of Discrete-Event Systems*", Math. Control Signals Systems, 1: 13-30, 1988.
- [20] W. M. Wonham, "*Notes on control of DES for Universidad de Toronto*", 1996-1997.
- [21] Antonio Ramírez Treviño, "*Scheduling en redes de Petri*", Tesis doctoral, Departamento de Ingeniería Eléctrica e Informática, Centro Superior de Ingenieros, Universidad de Zaragoza. Zaragoza, España, Diciembre 1993.



**CENTRO DE INVESTIGACION Y DE ESTUDIOS AVANZADOS DEL IPN
UNIDAD GUADALAJARA**

El Jurado designado por la Unidad Guadalajara del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, aprobó la tesis: **“Seguimiento de Modelo en SED usando RPI”** de la Srita. **Alejandra Santoyo Sanchez**, el día 30 de Julio de 1999.

EL JURADO

Dr. Luis Ernesto López Mellado
Investigador Cinvestav 3A
CINVESTAV DEL IPN
Guadalajara.

Dr. Antonio Ramírez Treviño
Investigador Cinvestav 2A
CINVESTAV DEL IPN
Guadalajara

Dr. Ricardo Raúl Jacinto Montes
Investigador Cinvestav 2B
CINVESTAV DEL IPN
Guadalajara.

Dr. Raúl Ernesto González Torres
Investigador Cinvestav 2C
CINVESTAV DEL IPN
Guadalajara



CINVESTAV
BIBLIOTECA CENTRAL



SSIT000003849