HERRAMIENTA BASADA EN AUTÓMATAS PARA DISEÑO E IMPLEMENTACIÓN CON PLC, DE SUPERVISORES DE SISTEMAS DE EVENTOS DISCRETOS.

MONOGRAFíA

Leydy Viviana Muñoz Robles
Lenin Galindez Trochez

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE ELECTRÓNICA, INSTRUMENTACIÓN Y CONTROL
POPAYÁN
2010

HERRAMIENTA BASADA EN AUTÓMATAS PARA DISEÑO E IMPLEMENTACIÓN CON PLC, DE SUPERVISORES DE SISTEMAS DE EVENTOS DISCRETOS.

Monografía

Leydy Viviana Muñoz Robles
Lenin Galindez Trochez

Director
PhD. Carlos Alberto Gaviria López

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE ELECTRÓNICA, INSTRUMENTACIÓN Y CONTROL
POPAYÁN
2010

Nota de Aceptación

_____

_____

_____

Director _____
PhD. Carlos Alberto Gaviria López

Jurado _____

Jurado _____

Fecha de sustentación: Popayán, 12 de octubre de 2010

CONTENIDO

1 Tabla de contenido

RESUMEN *1*

INTRODUCCIÓN *2*

1 SISTEMAS A EVENTOS DISCRETOS *5*

1.1 SISTEMAS Y MODELOS **5**

1.1.1 Concepto de sistema 6
1.1.2 Clasificación de los sistemas 6
1.1.3 Concepto de modelo 8
1.1.4 Concepto de estado 9
1.1.5 Concepto de evento 9
1.1.6 Sistemas a eventos discretos 10
1.1.7 Ejemplos de sistemas de eventos discretos 10

1.2 Lenguajes y formalismos. **12**

1.2.1 Concepto de lenguaje. 12
1.2.2 Lenguajes formales para modelado de sistemas a eventos discretos. 13
1.2.3 Lenguajes representados mediante autómatas 15
1.2.4 Operaciones con autómatas 20

1.3 Control supervisor y herramientas de simulación **22**

1.3.1 Concepto de supervisor 22
1.3.2 Herramienta para diseño de supervisores. 23
1.3.3 Herramienta para simulación de supervisores. 23

2 METODOLOGIA PARA EL DISEÑO DE SUPERVISORES *26*

2.1 DESCRIPCION GENERAL DE UNA METODOLOGIA **26**

2.2 PRESENTACIÓN DE LA METODOLOGIA EXISTENTE **28**

2.2.1 Enumeración de los eventos significativos 28
2.2.2 Identificación y modelado de todos los componentes 29
2.2.3 Obtención del modelo aumentado de la planta en lazo abierto 29
2.2.4 Identificación y modelado de todas las restricciones físicas 29
2.2.5 Obtención del modelo de la planta en lazo abierto 30
2.2.6 Identificación y modelado de todas las restricciones de control 31
2.2.7 Obtención del modelo de la planta en lazo cerrado 31

2.3 APLICACIÓN DE LA METODOLOGIA [1] EN LA CONSTRUCCIÓN DE AUTOMATAS SUPERVISORES DE PROCESOS QUE USEN CONTADORES Y CONDICIONES DE TIEMPO. . 32

2.4 COMPONENTES DE LA METODOLOGIA PROPUESTA Y SOLUCIÓN PARA QUE LA METODOLOGIA [26] PUEDA SER USADA PARA CONSTRUIR AUTOMATAS SUPERVISORES, EN PROCESOS EN LOS QUE EL TIEMPO JUEGA UN PAPEL IMPORTANTE. 33

2.5 METODOLOGÍA PROPUESTA **34**

2.5.1 Enumeración de los eventos significativos y clasificación en controlables y observables 35
2.5.2 Identificación y modelado de todos los componentes 36
2.5.3 Obtención del modelo aumentado de la planta en lazo abierto 36
2.5.4 Identificación y modelado de todas las restricciones físicas 37
2.5.5 Obtención del modelo de la planta en lazo abierto 38
2.5.6 Identificación y modelado de todas las restricciones de control 38

2.5.7 Obtención del modelo de la planta en lazo cerrado 39

2.6 USO DE LA HERRAMIENTA DESUMA PARA LA CONSTRUCCIÓN DEL AUTOMATA SUPERVISOR SIGUIENDO UNA SINTAXIS DISEÑADA PARA LA APLICACIÓN DE LA METODOLOGIA. 40

2.6.1 Sintaxis para la definición de los eventos significativos. 40
2.6.2 Sintaxis eventos significativos asociados a una variable, propuestos como ampliación a la metodología existente. 42
2.6.3 Temporizadores propuestos como ampliación a la metodología existente. 43
2.6.4 Contadores 46
2.6.5 Comparadores 49
2.6.6 Conjuntos de condiciones. 49

2.7 SIMULACIÓN Y VERIFICACIÓN MEDIANTE LA HERRAMIENTA UPPAAL DEL AUTOMATA SUPERVISOR OBTENIDO 50

2.8 GENERACIÓN DEL CÓDIGO LADDER MEDIANTE EL SOFTWARE FSMLADDER 53

3 APLICACIÓN DE LA METODOLOGIA A CASOS DE ESTUDIO 55

3.1 PROCESO DE BEBIDAS CARBONATADAS DEL LABORATORIO DE CONTROL DE PROCESOS DE LA UNIVERSIDAD DEL CAUCA, EN EL CUAL SE HACE USO DE TEMPORIZADORES. 56

3.1.1 EL PROCESO 56
3.1.2 LOS EVENTOS SIGNIFICATIVOS 59
3.1.3 Los componentes 60
3.1.4 Modelo aumentado de la planta en lazo abierto 61
3.1.5 Restricciones físicas y de control 62
3.1.6 El modelo de la planta en lazo cerrado 67
3.1.7 Simulación y verificación usando la herramienta UPPAAL 69
3.1.8 Generación automática del código LADDER utilizando la herramienta FSMLADDER ... 70
3.1.9 Supervisión del proceso. 73

3.2 AUTOMATA SUPERVISOR PARA UNA LÍNEA DE PRODUCCIÓN FORMADA POR UN ROBOT, DOS BANDAS TRANSPORTADORAS Y DOS UNIDADES DE ALMACENAMIENTO. 75

3.2.1 LOS EVENTOS SIGNIFICATIVOS 75
3.2.2 Los componentes 76
3.2.3 Modelo aumentado de la planta en lazo abierto 77
3.2.4 Restricciones físicas y de control 77
3.2.5 El modelo de la planta en lazo cerrado 78
3.2.6 Simulación y verificación usando la herramienta UPPAAL 80
3.2.7 Generación automática del código LADDER utilizando la herramienta FSMLADDER ... 80
3.2.8 Supervisión del proceso. 80

4 CONCLUSIONES, TRABAJOS FUTUROS Y LIMITACIONES. 81

4.1 CONCLUSIONES 81

4.2 TRABAJOS FUTUROS 84

4.3 LIMITACIONES 85

BIBLIOGRAFÍA 86

LISTA DE FIGURAS

Figura 1.1.Clasificacion de los sistemas [3]. *7*
Figura 1.2. Tipos de modelos [4]. *8*
Figura 1.3. Representación de un sistema a eventos discretos [28]. *10*
Figura 1.4Diagrama de estados civiles de una persona [8]. *11*
Figura 1.5. Intersección controlada por un semáforo [9]. *12*
Figura 1.6. DTE de una maquina contestadora [28]. *13*
Figura 1.7. Componentes de una red de Petri [28]. *14*
Figura 1.8 Composición paralela [23]. *17*
Figura 1.9. Autómata usando variables [28]. *19*
Figura 1.10. Bloqueo vivo y muerto en un autómata [21]. *20*
Figura 1.11. Autómata bien comportado [22]. *21*
Figura 1.12. Composición paralela [23]. *22*
Figura 1.13. System Editor de UPPAAL [28]. *24*
Figura 1.14. Simulator de UPPAL [28]. *24*
Figura 1.15. Verifier de UPPAL [28]. *25*
Figura 2.1 Ejemplo de un autómata con eventos controlables. *36*
Figura 2.2 Ejemplo de un autómata con eventos observables. *36*
Figura 2.3. Ejemplo restricción física. *38*
Figura 2.4 Ejemplo restricción de control mediante la eliminación de eventos. *39*
Figura 2.5. Ejemplo restricción de control como un autómata. *39*
Figura 2.6 Ejemplo de definición de un evento significativo en DESUMA [28]. *41*
Figura 2.7 Clasificación de los eventos significativos en DESUMA [28] *43*
Figura 2.8 Ejemplo de temporizador asociado a un evento controlable [28]. *45*
Figura 2.9 Construcción de un temporizador [28]. *46*
Figura 2.10 Construcción de un contador [28]. *48*
Figura 2.11 Construcción de un contador [28]. *48*
Figura 2.12. Construcción de un reset [28]. *48*
Figura 3.1Proceso bebidas carbonatadas [28]. *58*
Figura 3.2. Autómatas observables correspondientes a los sensores de cada tanque. *60*
Figura 3.3. Autómatas controlables correspondientes a los mixer en los que se hace uso de temporizadores [28]. *61*
Figura 3.4. Autómata controlable correspondiente a la motobomba [28]. *61*
Figura 3.5. Autómata controlable correspondiente a las electroválvulas [28]. *61*
Figura 3.6. Componentes que interactúan en el sistema del tanque dos. [28]. *63*
Figura 3.7. Componentes que interactúan en el sistema del tanque tres. [28]. *63*
Figura 3.8. Componentes que interactúan en el sistema del tanque cinco. [28]. *64*
Figura 3.9. Componentes que interactúan en el sistema del tanque cuatro. [28]. *64*
Figura 3.10. Restricciones de los componentes del proceso (tanques 2, 3, 4 y 5). [28]. *65*
Figura 3.11. Generadores de las restricciones de control [28] *66*
Figura 3.12. Eventos del proceso de tanques interactuantes [28]. *67*
Figura 3.13. Modelo en lazo cerrado. *68*
Figura 3.14. Verificación de la no existencia de bloqueos [28]. *69*
Figura 3.15. Verificación de la no existencia de estados inalcanzables [28]. *70*
Figura 3.16. Configuración de los eventos controlables en la herramienta FSMLADDER [28]. *71*
Figura 3.17. Configuración de los eventos observables en la herramienta FSMLADDER [28]. *71*
Figura 3.18 Fracción de código LADDER del proceso de bebidas carbonatadas [28]. *72*
Figura 3.19. Supervisorio SCADA para el proceso de bebidas carbonatadas [28]. *74*

LISTA DE ANEXOS

Anexo A. Manual de usuario Software DESUMA.

Anexo B. Manual de usuario Software FSMLADDER.

Anexo C. Metodología de supervisión basada en un modelo comportamental del Proceso.

RESUMEN

Este proyecto plantea una metodología para la el diseño de supervisores mediante el formalismo de autómatas extendidos de sistemas a eventos discretos, capaz de detectar todos los posibles estados de funcionamiento del proceso a evaluar.

Se utiliza en primera instancia una metodología existente que utiliza el concepto de autómatas para el diseño de supervisores, sin embargo esta metodología no es aplicable en procesos donde existe la presencia de contadores y temporizadores, por lo que se hace necesario hacer una ampliación a la misma. Esta ampliación involucra que además de contar con una herramienta para el diseño de los autómatas finitos se deba usar una herramienta para la verificación y simulación del autómata, con el fin de determinar que no existen bloqueos o estados inalcanzables en el supervisor. Para cumplir este objetivo se usa la herramienta UPPAAL.

Finalmente se desarrolla una herramienta software llamada FSMLADDER, que como su nombre lo indica nos permite a partir de una Máquina de estado finito (FSM) o autómata obtener un supervisor del proceso, para posteriormente ser traducido de manera automática al lenguaje LADDER . De esta manera el usuario deberá crear los autómatas a partir de la metodología propuesta en DESUMA, cargarlos al software FSMLADDER y este se encargara de realizar la composición paralela (autómata supervisor), convertirlo de manera automática al archivo XML usado por UPPAL para realizar la verificación y simulación y finalmente generar el archivo con el código LADDER del proceso.

INTRODUCCIÓN

En procesos industriales es común que existan restricciones de tiempo cuyo cumplimiento es esencial para su funcionamiento. En general, están formados por varios componentes que interactúan entre si, como sensores, actuadores y condiciones sobre los mismos
En las últimas dos décadas han surgido diferentes formalismos para representarlos, así como para describir propiedades sobre su comportamiento. Uno de los más exitosos ha sido el de los autómatas temporizados [2], nacido como extensión a la teoría de autómatas, agregando la posibilidad de incorporar restricciones de tiempo.

Este proyecto se basa en una metodología descrita en el capítulo 2, denominada SUPERVISIÓN BASADA EN UN MODELO COMPORTAMENTAL DEL PROCESO [26], diseñada en conjunto por la Universidad Politécnica de Cataluña ubicada en Barcelona – España, y el Laboratorio de Análisis y Arquitectura de Sistemas (LAAS) de Toulouse – Francia, esta metodología propone utilizar el conocimiento heurístico que el experto posee para modelar un proceso, mediante la construcción de varios autómatas finitos y a partir de ellos obtener un único autómata supervisor que determinara el comportamiento de un proceso.

En el capítulo 3 será aplicada la metodología a casos típicos en secuencias de procesos industriales, en los cuales puede o no, existir el uso de temporizadores, contadores y variables; demostrando que la metodología actual [26] no es aplicable en procesos en los que se necesita hacer uso de condiciones de tiempo, variables, o contadores, siendo estos utilizados en entornos industriales. De esta manera, se propondrá una ampliación a la metodología [26], para que sea aplicable a estos casos.

La metodología ampliada se basará en el formalismo de autómatas extendidos en vez de en el formalismo de autómatas finitos en que se basa la metodología descrita en [26] y buscara obtener un modelo a eventos discretos del proceso, con el cual es posible monitorizar el estado de funcionamiento de procesos que utilicen condiciones de tiempo o contadores para su funcionamiento.

Una segunda fase del proyecto se detalla en el capítulo 3 y describe el uso de herramientas software que permitan implementar la metodología propuesta en el capítulo 2, validar y verificar el autómata supervisor obtenido y traducir el autómata supervisor al código LADDER utilizado para la programación de un PLC Micrologix 1500 serie C.

Para lograrlo, se propone que el usuario diseñe el autómata supervisor del proceso siguiendo tanto la metodología ampliada, como la sintaxis propuesta en este proyecto de grado, se propone además la construcción del autómata supervisor utilizando la herramienta DESUMA, una herramienta para diseño de autómatas desarrollada en la Universidad de Mount Allison y que permite realizar operaciones entre autómatas finitos.
Aunque en este punto, la metodología se ha ampliado al uso de autómatas extendidos, para cumplir con los objetivos de este proyecto se necesita una herramienta que permita implementar la metodología, además que permita realizar la validación y simulación del autómata supervisor obtenido y finalmente que permita traducir este autómata al código LADDER, un lenguaje de programación gráfico basado en los esquemas eléctricos de control clásicos, usado para la programación de los autómatas programables o PLC de la empresa Allen Bradley.

La herramienta construida, denominada FSMLADDER, usa las librerías UMDES de DESUMA, que permiten las operaciones entre autómatas, para realizar la composición paralela, además utiliza los model checkers temporizados implementados en la herramienta UPPAAL, consistentes en programas que dada una especificación de un sistema de tiempo real y propiedades sobre este último, incluyendo condiciones de temporizadores, determinan de manera automática si estas se cumplen o no. UPPAAL es un entorno integrado de herramientas para modelar, simular y verificar sistemas de tiempo real, desarrollado por la Universidad de Aalborg en Dinamarca y la Universidad de Uppsala en Suecia. Pero usar estos dos sistemas, DESUMA y UPPAAL tiene un costo, es necesario que la herramienta FSMLADDER interprete la composición paralela obtenida en DESUMA, correspondiente en un archivos con extensión *.FSM y los convierta a los archivos con extensión *.XML usados por UPPAAL para realizar la verificación y simulación del autómata supervisor y sus condiciones de temporizadores.

Ahora bien, de la misma manera como cuando un usuario diseña un GRAFCET para finalmente llevarlo a código LADDER, con la metodología se busca diseñar un autómata supervisor utilizando el formalismo de autómatas finitos para traducirlo también al código LADDER, con la diferencia de que antes de realizar la conversión se verifican las propiedades de alcanzabilidad, accesibilidad y bloqueo de el autómata supervisor. Esta conversión al código LADDER también se realizará de forma automática mediante la utilización de la herramienta FSMLADDER, con solo seleccionar el archivo XML al que se le ha realizado la verificación y simulación.

Actualmente no existe ninguna herramienta que permita generar la composición paralela utilizando temporizadores, verificar y simular el autómata supervisor obtenido y generar el código LADDER para la

programación de un PLC MICROLOGIX 1500, por lo que este proyecto de grado llena este vacío tecnológico existente.

En el capítulo 3 se puede encontrar una descripción más detallada de la metodología presentada. Se debe aclarar que esta es una metodología general para la obtención de un modelo a eventos discretos de un proceso industrial, con el cual es posible monitorear el estado de funcionamiento del proceso.
Para que la metodología propuesta sea validada, en el capítulo tres se aplica la metodología a dos ejercicios de simulación y un caso experimental y se muestran los resultados, con el fin de comprobar que resuelve los limitantes de la metodología en [26] y permite generar código correcto para la programación del PLC que se mencionó con anterioridad.

Para comprender mejor la información planteada en este documento se ha dedicado el capitulo dos para la definición aspectos teóricos básicos de sistemas a eventos discretos y ejemplos de uso. Introduciendo conceptos como autómata finito, evento, estado, composición paralela, estados muertos, y estados inalcanzables.

Por ningún motivo el lector deberá confundir el término autómatas finitos con el término autómatas programables o PLC (Controlador Lógico Programable). Un autómata finito hace referencia a una maquina de estado finito o a una secuencia de eventos y estados dentro de un proceso, mientras el concepto de autómatas programables hace referencia a un equipo comúnmente utilizado en la industria, diseñado para programar y controlar procesos secuenciales en tiempo real; en este proyecto se diseñara una metodología para a través de autómatas finitos obtener el código de programación de un autómata programable o PLC. De igual manera es importante que se diferencie el concepto supervisor del concepto supervisorio – en este proyecto se hará uso de ambos - el autómata supervisor es un autómata finito que monitorea el estado de funcionamiento de un proceso obtenido al aplicar la metodología, mientras el supervisorio es una aplicación de software que proporciona comunicación con los dispositivos de campo y permite monitorear y controlar el proceso de forma automática desde la pantalla del ordenador.

1 SISTEMAS A EVENTOS DISCRETOS

En este capítulo introduciremos conceptos relacionados con sistemas a eventos discretos y la representación de lenguajes formales mediante autómatas.
En las últimas décadas, gracias a la evolución de la tecnología, se ha ampliado el uso de los sistemas que se definirán formalmente más adelante como conducidos por eventos discretos, o simplemente de eventos discretos. En un sistema de eventos discretos, las actividades son dirigidas por la ocurrencia asincrónica de eventos bien sea controlados (por ejemplo el pulso de un botón) como no controlados (como la falla espontanea de un equipo). Tal tipo de sistemas puede encontrarse en múltiples áreas del conocimiento tales como en sistemas de producción automatizados, sistemas de comunicaciones o sistemas de información.
La complejidad de un sistema no debería medirse en función del número de componentes o subsistemas que lo integran (existen sistemas electrónicos con un considerable número de componentes que se encuentran lejos de ser clasificados como sistemas complejos, por ejemplo, un televisor), ni tampoco en función del número de ecuaciones necesarias para describir su comportamiento. En el ámbito de este tema, la complejidad de un sistema no se entiende como una propiedad inherente al comportamiento del mismo, sino más bien como una falta de metodología y de herramientas que permitan especificar y formalizar el conocimiento que se tiene del sistema con el objetivo de desarrollar un modelo que presente un comportamiento similar al del sistema real[1].Disponer de metodologías que permitan formalizar la dinámica de un proceso, así como una buena base en el uso de las herramientas matemáticas, hace que algunos sistemas que en el pasado habían sido considerados complejos, ya no lo sean en la actualidad.
Entre los formalismos más conocidos para representación de sistemas de eventos discretos (SED) se encuentran las Redes de Petri, Grafcet, y los Autómatas, objeto de estudio de este trabajo.

1.1 SISTEMAS Y MODELOS

Los sistemas hacen parte de la vida diaria, la definición de estos es cada vez más compleja y se tiene que tener una gran cantidad de información específica a la hora de definirlos, un sistema se clasifica con base a su tipo, o a su aplicación. Entendiendo esta idea, el término sistema es aplicable a cada uno de estos ejemplos:

- Una planta de fabricación con maquinas, personal, dispositivos de transporte y almacén.
- El servicio de emergencias de un hospital, incluyendo al personal, las salas, el equipamiento y el transporte de los pacientes.
- Una red de ordenadores con servidores, clientes, dispositivos de disco, impresoras, etc.
- Un supermercado con control de inventario, cajeros y atencional cliente.
- Un parque temático con atracciones, tiendas, restaurantes, trabajadores, clientes y aparcamientos.

1.1.1 Concepto de sistema

El concepto de sistema es una idea generalizada alrededor de nuestro medio, se habla de ello en medios cotidianos, ciencia, tecnología, incluso en el campo deportivo es común escuchar el termino, se han creado en la actualidad diferentes campos que llevan por nombre proyectos de sistemas, análisis de sistemas e incluso ingeniería de sistemas.

Un sistema se puede definir como un conjunto de partes o elementos organizados y relacionados que interactúan entre sí para lograr un objetivo. Los sistemas reciben (entrada) datos, energía o materia del ambiente y proveen (salida) información, energía o materia, también se pueden definir como una combinación de elementos que actúan conjuntamente y cumplen un determinado objetivo [2], esta interacción se puede dar tanto a nivel software como a nivel hardware.

A nivel hardware es común hablar de sistemas mecánicos, que se componen de elementos que pueden comportarse como masas, amortiguadores o muelles, los sistemas eléctricos se componen de tres elementos fundamentales como son las resistencias, los condensadores y las bobinas y los sistemas electromecánicos que son la combinación de los anteriores y a nivel software uno de los sistemas más conocido es el de los sistemas operativos.

1.1.2 Clasificación de los sistemas

Los sistemas pueden ser explicados de una manera muy general, exponiendo solamente cuáles son sus entradas y salidas, pero también

puedes ser descritos de manera profunda, explicando el funcionamiento y los componentes y la interacción entre cada uno de sus elementos

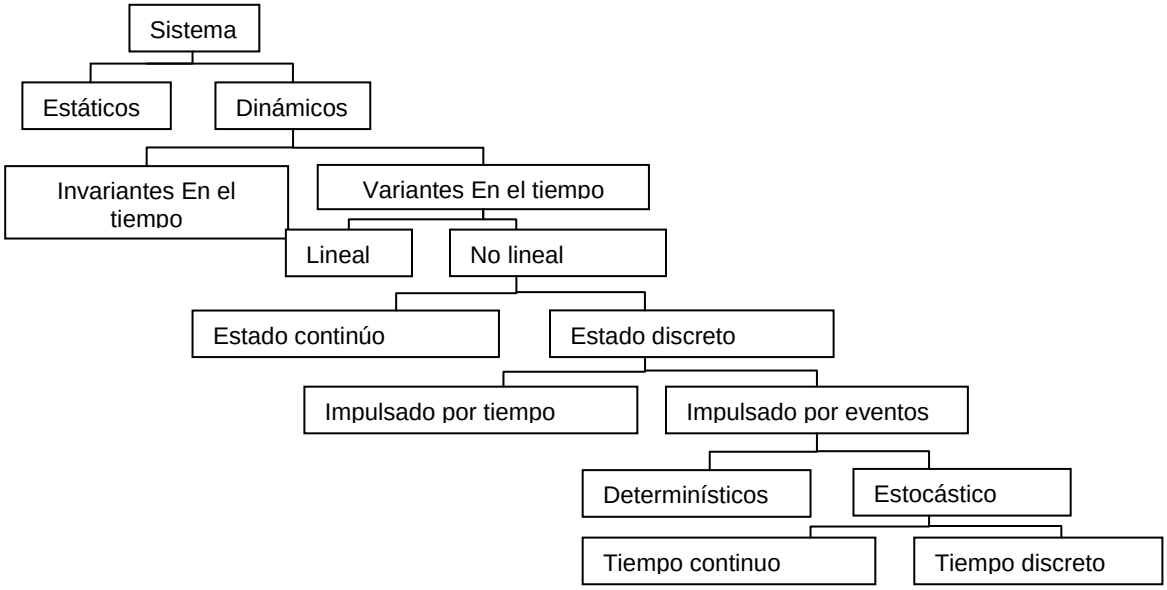


Figura 1.1.Clasificacion de los sistemas [3].

Siguiendo la rama de la taxonomía en la Figura [3] que lleva hasta los sistemas impulsados por eventos, se describe la siguiente clasificación de sistemas:

- **Sistemas dinámicos y estáticos:** Los sistemas dinámicos son sistemas que evolucionan con el tiempo, por ejemplo una cinta transportadora en una fábrica. Los sistemas estáticos representan el sistema en un instante determinado, donde el tiempo no juega ningún papel.

- **Sistemas variantes en el tiempo e invariantes en el tiempo:** El comportamiento de los sistemas invariantes en el tiempo no cambia con el tiempo. Esta propiedad también llamada Estacionaria, implica que nosotros podemos aplicar una entrada especifica a un sistema y esperar que responda siempre de la misma forma.

- **Sistemas lineales y no lineales:** Un sistema lineal satisface la condición $g(a_1u_1 + a_2u_2) = a_1g(u_1) + a_2g(u_2)$ donde u_1, u_2 son dos vectroes de entrada, a_1, a_2, son dos números reales y g(.) es la salida resaltante. Los sistemas dinámicos lineales invariantes en el tiempo, son descritos por modelos en espacio de estados.

- **Sistemas de estados discretos y continuos:** En los sistemas de estados continuos el estado de las variables puede generalmente tomar un valor real o complejo. En Sistemas de estado discreto los estados de las variables pertenecen a un conjunto discreto de elementos y donde la transición entre estados es consecuencia únicamente de la ocurrencia de algún evento *e* perteneciente a un conjunto discreto de elementos *E*.

- **Sistemas conducidos por el tiempo y sistemas conducidos por eventos:** En sistemas de estado continuo, las variables de estado continuo cambian continuamente con el tiempo. Tales sistemas se denominan conducidos por el tiempo, y las transiciones están sincronizadas por el reloj, siendo el reloj por sí solo el responsable de cualquier transición de estado.

  En los sistemas conducidos por eventos, la ocurrencia de eventos generados asíncronamente, fuerzan instantáneamente una transición de estados. Entre ocurrencias de eventos el estado permanece inalterable [3]. En varios instantes de tiempo, no necesariamente conocidos de antemano y no necesariamente que coincidan con períodos de reloj, algún evento e anuncia que está ocurriendo. Estos son el tipo de sistemas son los que se estudian en este proyecto de grado.

1.1.3 Concepto de modelo

Un modelo definido de la manera más general es una representación de la realidad, y que es desarrollado para un fin específico, mediante cualquier tipo de lenguaje común a un grupo de personas. Para interactuar con un sistema se puede interactuar directamente con él, o usar modelos. Existen diferentes tipos de modelos como se muestra en [4]: mental, verbal, físico y matemático, como lo muestra la figura 1.1.

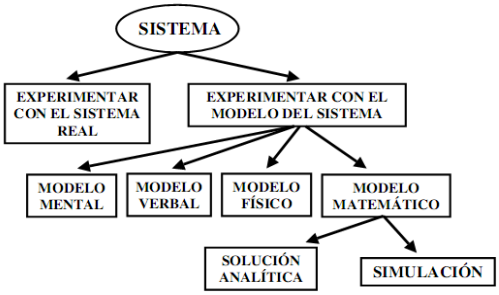


Figura 1.2. Tipos de modelos [4].

Los modelos más usados en ingeniería son los modelos matemáticos, pero obtener un modelo matemático que caracterice de forma adecuada el comportamiento de un determinado sistema no es sencillo, y es uno de los grandes problemas a la hora de implementar la solución a un problema.

Ningún modelo matemático puede abarcar toda la realidad del sistema, sin embargo, para que un modelo sea útil no es necesario que sea excesivamente complicado. Basta con que represente los aspectos esenciales del mismo y que las predicciones sobre el comportamiento del sistema, basadas en dicho modelo, sean lo suficientemente precisas.

1.1.4 Concepto de estado

El concepto de estado según el diccionario de la real Academia Española es "La situación en que se encuentra alguien o algo, y en especial cada uno de sus sucesivos modos de ser o estar", aplicado a un sistema es la situación o la condición en que se hallan los elementos del sistema en algún instante.
Un elemento no puede estar en más de un estado al mismo tiempo [5]. Por ejemplo la bomba de un sistema hidráulico, puede tener dos estados, está encendida o apagada, pero no puede estar encendida y apagada al mismo tiempo.

En un sistema, el estado resume la información concerniente a entradas anteriores y que es necesaria para determinar el comportamiento del sistema para entradas posteriores.

1.1.5 Concepto de evento

Un evento es una acción, y debe entenderse como la representación de la ocurrencia de un cambio instantáneo en alguna parte de un sistema. Puede caracterizarse por un valor y un instante en el que ocurre. El valor puede ser un número, un vector, una palabra o, en general, un elemento cualquiera de un conjunto determinado [6].

Un evento debe ser pensado como algo que ocurre instantáneamente y provoca transiciones desde un valor del estado a otro. Puede ocurrir de formas diversas: Por una acción específica que tiene lugar (Ej.: alguien que presiona un botón), por una ocurrencia espontánea dictada por la naturaleza (Ej.: un computador que se bloquea por cualquier razón demasiado compleja de entender).o como resultado de varias condiciones que se cumplen de repente (Ej.: el sobrepaso del nivel de un tanque por un nivel peligroso) [7].

Un evento desencadena cambios de estado en partes del sistema o del sistema en general, por ejemplo el interruptor de encendido en un vehículo, es capaz de iniciar la combustión interna en el motor y así permitir el arranque de este, una alarma en un silo de almacenamiento por sobrepaso de capacidad, son acciones que al final le permiten al sistema evolucionar a un siguiente estado dentro del conjunto de estados posibles.

1.1.6 Sistemas a eventos discretos

Los sistemas, como se expuso anteriormente, pueden estar condicionados en su evolución por el tiempo o por eventos. Cuando el espacio de estados de un sistema está naturalmente descrito por un conjunto discreto tal como {0, 1, 2, 3….}, y las transiciones de estados se observan únicamente en puntos discretos en el tiempo, se asocian esas transiciones de estado con eventos y se está ante un sistema a eventos discretos [7].

Su comportamiento se caracteriza por una secuencia finita o infinita de estados delimitados por eventos que ocurren de manera asíncrona. La figura 1.2, muestra la evolución de un sistema de eventos discretos, donde en el eje *y* se representa los estados {x1, x2, x3…, x6}, y en el eje *x* el tiempo, pero nótese que el sistema solo cambia de estado cuando se produce en el tiempo un evento {e1, e2, e3…, e7}, aunque esos eventos no son periódicos.

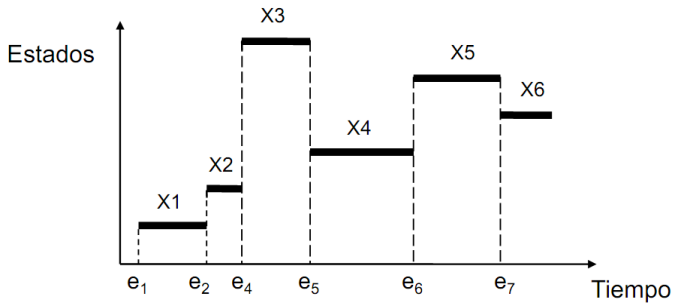


Figura 1.3. Representación de un sistema a eventos discretos [28].

1.1.7 Ejemplos de sistemas de eventos discretos

- **Ejemplo del estado civil de una persona**

Los “estados civiles” en que puede estar una persona: *soltera, casada, viuda, divorciada*, etc. De uno de estos estados se puede pasar a otro al ocurrir un evento o acción, que es el segundo concepto básico de la modelación discreta. Así, por ejemplo, del estado “soltero” se puede pasar al estado

“casado” al ocurrir el evento “boda”. Similarmente, se puede pasar de “casado” a “divorciado” mediante el evento “divorcio”. En estos modelos se supone que se permanece en los estados un cierto tiempo, hasta la ocurrencia de otro evento, pero por el contrario los eventos son instantáneos. Esto puede ser más o menos realista dependiendo de la situación que se esté modelando, por ejemplo, en el medio rural hay bodas que duran una semana, pero desde el punto de vista de la duración de una vida humana, este tiempo puede considerarse despreciable. En el caso del evento “divorcio”, pudiera ser inadecuado considerarlo como instantáneo, pues hay divorcios que duran años. En este caso, el modelo puede refinarse definiendo un nuevo estado “divorciándose”, al que se llega desde “casado” mediante el evento “inicio divorcio”. La figura 1.3, ilustra el modelo referido al ejemplo planteado [8].

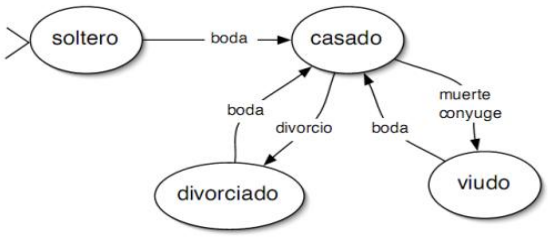


Figura 1.4Diagrama de estados civiles de una persona [8].

- **Ejemplo de un Semáforo:**

Los vehículos para moverse en las ciudades hacen uso de los semáforos, considérese como ejemplo una intersección en T como se muestra en la Figura 1.4., que será visto como un sistema a eventos discretos.
Hay cuatro tipos de vehículos:

- (1,2) Vehículos que vienen desde el punto 1 y gira a la derecha hacia el punto 2.
- (1,3) Vehículos que vienen desde 1 y giran a la izquierda hacia el punto 3.
- (2,3) Vehículos que siguen derecho desde el punto 2 hasta 3.
- (3,2) Vehículos que siguen derecho desde el punto 3 hasta 2.

Al semáforo se le asigna que encienda rojo para los vehículos (1,2) y (1,3) (verde para los vehículos (2,3) y (3,2)) o encienda verde para los vehículos (1,2) y (1,3), (rojo para los vehículos (2,3) y (3,2)). En este caso el conjunto de eventos es:

$$\mathrm{E} = \{\mathrm{a}_{12}, \mathrm{a}_{13}, \mathrm{a}_{23}, \mathrm{a}_{32}, \mathrm{d}_{12}, \mathrm{d}_{13}, \mathrm{d}_{23}, \mathrm{d}_{32}, \mathrm{g}, \mathrm{r}\}$$

Donde

$a_{12}, a_{13}, a_{23}, a_{32}$	Llegada de vehículos de cada uno de los cuatro tipos.
$d_{12}, d_{13}, d_{23}, d_{32}$	Es la salida de uno de los 4 tipos de vehículos dejando la intersección.
g	Indica que la luz verde está encendida para los vehículos (1,2) y (1,3).
r	Indica que la luz roja esta encendido para los vehículos (1,2) y (1,3).

Un posible espacio de estados se define por una palabra formada por los cuatro tipos de vehículos .y el estado del semáforo, esto es:

$$X = \{x_{12}, x_{13}, x_{23}, x_{32}, y\} : x_{12}, x_{13}, x_{23}, x_{32} \geq 0, y \in \{G, R\}$$

Donde $x_{12}, x_{13}, x_{23}, x_{32}$representan el número de vehículos de cada tipo, y y es el estado del semáforo, G denota verde para los vehículos (1,2) y (1,3) y R denota rojo para los vehículos (1,2) y (1,3) [9].

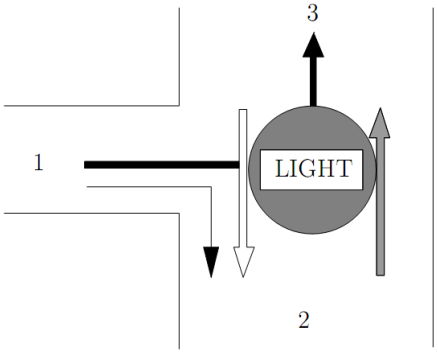


Figura 1.5. Intersección controlada por un semáforo [9].

1.2 Lenguajes y formalismos.

1.2.1 Concepto de lenguaje.

Para llegar a este concepto es necesario antes definir algunos conceptos previos.

- **Símbolo:** La noción más primitiva es la de símbolo, un símbolo es una representación distinguible de cualquier información. Los símbolos pueden ser cualquiera, como w, 9, #.Un símbolo es una entidad indivisible.

- **Palabras**: Con la unión de símbolos es posible formar cadenas, de caracteres tales como {fmdskf, casa, rrrr}, estas cadenas de caracteres son denominadas palabras [12].

Partiendo de los anteriores conceptos, un lenguaje, se define como un conjunto de palabras, que a su vez están formadas por símbolos. Así {abracadabra}, es un lenguaje de una sola palabra, {1100, 0000, 1010}, es otro lenguaje pero con otro tipo palabras y de símbolos. También un lenguaje, tal como el español o un lenguaje de programación como java, puede ser considerado como un conjunto de oraciones, esto es, cadenas finitas de elementos de algún vocabulario básico.

1.2.2 Lenguajes formales para modelado de sistemas a eventos discretos.

Existen diferentes formalismos para el modelado de de sistemas a eventos discretos, entre ellos se tiene:

- **Gráfico de transición de estados**

También conocido como DTE, es una herramienta de modelado que enfatiza el comportamiento dependiente del tiempo del sistema. Los principales componentes del diagrama son estados, y flechas que representan los cambios de estado. La figura 1.5, muestra un DTE, que describe el comportamiento de un maquina contestadora [16].

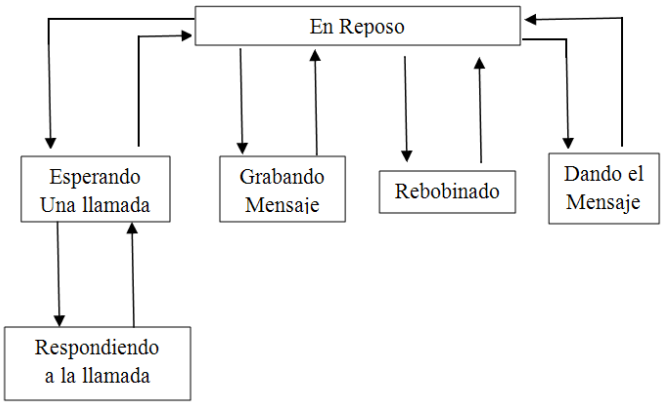


Figura 1.6. DTE de una maquina contestadora [28].

- **Redes de Petri**

Las redes de Petri se utilizan para modelar sistemas cuando sus procesos pueden actuar de forma concurrente. Establecen un marco de referencia que

permite resolver el problema que se presenta cuando la concurrencia conduce a un callejón sin salida (bloqueo muerto), o cuando se llega a un estado en el que la transición ocurre una y otra vez y no es posible avanzar a otro estado.

Estas redes se originaron en el trabajo realizado por C. Petri en su tesis doctoral de 1961. Han sido investigadas intensamente desde entonces, dado su aplicación generalizada, especialmente en los sistemas de computación, donde se trata de lograr mayor velocidad y eficiencia.

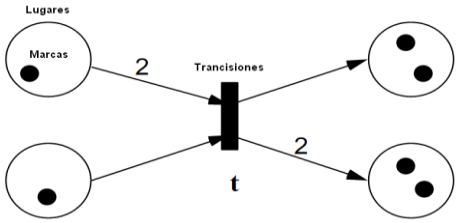


Figura 1.7. Componentes de una red de Petri [28].

En la figura 1.6. se indican los elementos que componen una red de Petri. Los lugares representan condiciones. Las transiciones representan, igual que en los diagramas de estado, eventos. Las marcas o fichas (tokens), que se ubican en los lugares, representan condiciones cumplidas. La ausencia de fichas indica condiciones no cumplidas. Los eventos o transiciones representan las instrucciones que se deben ejecutar en caso que se cumplan las condiciones [17].

- **GRAFCET**

Surge en Francia a mediados de los años 70, debido a la colaboración entre algunos fabricantes de PLC(Controlador Lógico Programable), como Telemecanique y Aper con dos organismos oficiales, AFCET (Asociación Francesa para la Cibernética, Economía y Técnica) y ADEPA (Agencia Nacional Para el Desarrollo de la Producción Automatizada).

Es un diagrama funcional que describe la evolución de los procesos que se pretende automatizar, indicando las acciones que hay que realizar sobre el proceso y que informaciones las provocan. El diagrama GRAFCET, está compuesto de Etapas que es el nexo de unión entre las actuaciones que hay que hacer sobre el proceso (Ej. Activar un motor, cerrar una Válvula, etc.) y el programa de usuario, cargado en el PLC; y las condiciones de transición, que son condiciones que se deben cumplir para que el proceso evolucione de una etapa a otra [18].

- **DEVS (Discrete Event System Especification)**

Un modelo de este tipo procesa una trayectoria de eventos de entrada y, de acuerdo a las condiciones iníciales del modelo y a esta trayectoria de entrada, produce una trayectoria de eventos de salida.

El formalismo DEVS permite representar cualquier sistema que satisfaga la condición siguiente: en cualquier intervalo acotado de tiempo, el sistema experimenta un número finito de eventos. La trayectoria de entrada al sistema es una secuencia ordenada de eventos y la función de transición del modelo tiene una forma especial, que limita la trayectoria de salida para que sea también una secuencia de eventos [19].

- **Autómatas finitos**

Un autómata también puede verse como un grafo (conjunto de objetos llamados nodos unidos por enlaces llamados arcos), donde los nodos, representan los estados posibles del sistema. Los nodos son vinculados mediante transiciones. Existe un estado especial, llamado estado inicial, que representa las condiciones iníciales del sistema desde donde seguirá evolucionando según los eventos. Un cambio en el sistema se representa en la máquina de estados como un cambio de estado a través de la transición correspondiente, en un autómata cada entrada diferente genera una salida diferente, pero siempre el mismo resultado con los mismos datos de entrada [14].

Este último lenguaje formal o formalismo, constituye la base de la metodología para el diseño de supervisores a través de autómatas finitos descrita en [1].

1.2.3 Lenguajes representados mediante autómatas

Habitualmente el comportamiento de un sistema se define mediante un conjunto de secuencias de acciones que puede ejecutar ese sistema. Desde este punto de vista formal podemos considerar que un conjunto de secuencias es un lenguaje formal y una forma habitual de representar los lenguajes formales es mediante los autómatas.

Entre los diferentes modelos de autómatas que existen, los más utilizados son los autómatas finitos ya que la mayor parte de los sistemas estudiados poseen un número finito de estados. Así, Un Sistema de Eventos Discretos puede ser completamente modelado como una secuencia lineal de estados y y eventos del sistema.

Un autómata finito consiste de un conjunto finito de estados y un conjunto de transiciones de estado a estado, que se dan sobre los símbolos tomados del alfabeto de un lenguaje. Para cada símbolo de entrada existe exactamente una transición a partir de cada estado (posiblemente de regreso al mismo estado).

Un Autómata recibe secuencialmente una cadena de símbolos y cambia de estado por cada símbolo leído o también puede permanecer en el mismo estado. Al final de la lectura el estado del Autómata nos indica si la cadena es aceptada o mejor dicho pertenece al Lenguaje que describe nuestra maquina. Si al final de leer todos los símbolos de entrada la maquina esta en alguno de los estados Finales entonces esa cadena es aceptada, si el estado no es final entonces la cadena no pertenece al lenguaje.

Las partes que componen una Autómata denotado por G son 6 definidos así:
G = {X, E, f, Γ, x_0,X_m}

Donde:

X:	Conjunto finito de estados.
E:	Es el conjunto de eventos asociados con las transiciones en G.
f: X *x*E $\rightarrow$X	es la función de transición f (x,e) = y, significa que hay una transición marcada por el evento e del estado x al estado y.
Γ: X $\rightarrow 2^E$	es la función de evento activo. $\Gamma_{(x)}$es el conjunto de todos los eventos e para el cual f(x,e) es definido y esto es llamado el conjunto de eventos activos de G a x.
x_0:	Es el estado inicial.
$X_m \subseteq X$	Es el conjunto de estados marcados [20].

Así, un autómata finito es un modelo matemático de un sistema, con entradas y salidas discretas. El sistema puede estar en cualquiera de un número finito de configuraciones o estados. El estado del sistema resume la información concerniente a entradas anteriores y que es necesaria para determinar el comportamiento del sistema para entadas posteriores [15].

El mecanismo de control de un elevador es un buen ejemplo de un sistema de estados finitos. El mecanismo no recuerda todas las demandas previas de servicio, sino solo el piso en el que se encuentra, la dirección del movimiento (hacia arriba o hacia abajo), y el conjunto de demandas de servicio aun no satisfechas [24].
Un estado, por lo general denotado como *xo*, es el estado inicial, en el que el autómata comienza. Algunos estados están designados como final o

aceptación. A un autómata finito se le asocia un grafo dirigido, como se muestra en la figura 1.8

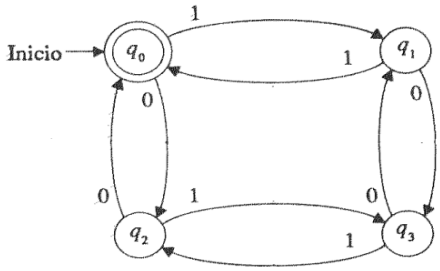


Figura 1.8 Composición paralela [23].

- **Autómata supervisor**

Un autómata supervisor, realiza funciones de supervisión y vigilancia, el autómata supervisor ve (observa) algunos, o posiblemente todos, los estados que el sistema ejecuta. Entonces, el supervisor le dice al sistema cual estado está asignado después del estado actual. Más precisamente, el supervisor tiene la capacidad de deshabilitar algunos de un conjunto de eventos, pero no necesariamente todos, verificando los eventos posibles del sistema. La decisión sobre qué eventos se deshabilitarán puede cambiar cada vez que el supervisor observa la ejecución de un nuevo evento por el sistema. G o generador. De este modo, el supervisor ejerce un control dinámico realimentado del sistema. Las dos consideraciones clave aquí es que el supervisor está limitado en cuanto a la observación de los eventos ejecutados por el sistema y también está limitado en términos de la deshabilitación de dichos eventos [15].

- **autómata extendido**

Un autómata extendido es una ampliación de un autómata ordinario, donde además de estados y eventos se permite el uso de variables, condiciones para la transición de estados, entradas y salidas, entre otras.. En este trabajo se han considerado las siguientes extensiones: autómatas temporizados y autómatas con variables, a continuación se describe cada uno de ellos.

Autómatas temporizados

Un autómata temporizado es una máquina de estados finita, dotada de un conjunto finito de relojes; dichos relojes son variables reales que representan el tiempo transcurrido entre la ocurrencia de eventos.

Los relojes de un autómata temporizado están sincronizados, esto implica que todos avanzan a la misma velocidad. En la evolución de los relojes pueden aparecer discontinuidades, provenientes de la ejecución de una transición en la máquina de estados, en las que a un subconjunto de relojes se les establece un nuevo valor, que generalmente es cero. Las transiciones tienen asociadas una condición de activación, que es un predicado sobre los relojes del autómata: para que una transición en la componente discreta pueda ser tomada, los valores de los relojes deben satisfacer dicha condición de activación.

Los grafos temporizados constituyen un modelo matemático-computacional en el cual pueden representarse de manera formal, simple, clara y modular problemas del mundo real. Varias definiciones similares de *grafos* (*autómatas*) *temporizados* han sido propuestas, como es el caso de [5, 6, 7, 8, 9]. Un grafo temporizado es un autómata extendido con un conjunto *finito* de variables reales, llamadas *relojes*, cuyos valores se incrementan uniformemente con el paso del tiempo. Dichos relojes son utilizados para medir, por ejemplo, el tiempo transcurrido entre dos eventos, el tiempo de espera o la demora de una comunicación. Las restricciones temporales inherentes a las acciones del sistema son expresadas ligando condiciones de *activación* a cada una de las transiciones del autómata. Una transición está *habilitada*, es decir puede ser atravesada, cuando su condición asociada es satisfecha por los valores de los relojes. Un reloj puede ser puesto a cero en cualquier transición. A todo instante, el valor de un reloj es igual al tiempo transcurrido desde la última vez en que fue puesto a cero [5, 10].

Autómatas con variables

Las variables pueden ser usadas como una transición dentro de un autómata y deben ir acompañadas de un evento, los valores que se les asignen deben pertenecer a un dominio finito y su valor puede cambiarse utilizando transiciones controlables u observables, es decir transiciones asociadas a un contador o un actuador.

El uso más frecuente de las variables es para realizar conteos de las veces que ocurre un evento dentro de un autómata, sin embargo en este proyecto se ampliara su uso para que además pueda cambiar su valor con la ocurrencia de un estado, es decir no usarla solo como un contador de eventos sino también como un contador de estados.

Las variables son compartidas por todos los autómatas de la composición (es decir pueden verse como globales a todas los autómatas) y deben ser definidas de antemano [25].

En la figura 1.9 se muestra un ejemplo de un autómata que usa variables, el ejemplo describe el comportamiento un local de ropa puede estar vacío o lleno según la cantidad de personas que hay en su interior. Inicialmente está vacío, pero luego de ingresar por lo menos una persona se pasa a un estado de lleno, el cual indica que hay en el sitio por lo menos una persona, se utiliza la palabra lleno por referirse a que no está vacío. A partir de ahí, la gente puede ingresar o salir. Cuando sale el último, vuelve a estar vacío.

Se puede observar que existe un evento “ingresa persona” y otro evento “egresa persona”, que corresponden a la señal de un sensor, es decir a un evento observable; ahora bien, a estos eventos podemos asociarle una variable que nos permita realizar un conteo del las personas que ingresan al sitio, así, cada vez que se active el sensor de ingreso la variable se incrementara en uno, y cada vez que se active el sensor de egreso el valor de la variable decrementará.

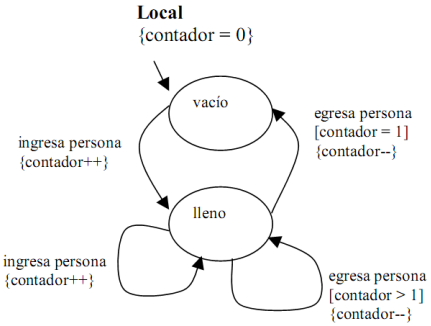


Figura 1.9. Autómata usando variables [28].

Definición de autómata temporizado.

Según [32] Un grafo (autómata) temporizado es una quíntupla $G = \langle L, X, E, l_0, I \rangle$, donde:

- ***L*** es un conjunto finito de nodos llamados locaciones.
- ***X*** es un conjunto finito de relojes.
- ***E*** es un conjunto finito de aristas llamadas transiciones. Cada transición es una quíntupla $e = \langle l, \alpha, \psi_X, \rho_X, l' \rangle$ que consiste en una locación origen $l \in L$, una locación destino $l' \in L$, una etiqueta $\alpha \in A$, una condición ψ_X y un conjunto $\rho_X \subseteq X$ de relojes que son puestos a cero (reseteados) simultáneamente con la transición.

- l_0 es la locación inicial. El estado inicial del sistema es $\langle l_0, v_0 \rangle$, con $v_0(x) = 0$ para todo $x \in X$, es decir, la locación inicial con todos los relojes puestos en cero.
- I es el conjunto de invariantes de las locaciones del grafo. Para cada $l \in L$, $I_l \in \Psi_X$ es el invariante de la locación. Cuando el sistema se encuentra en un estado $\langle l, v \rangle$, éste puede permanecer en la locación l dejando pasar el tiempo, mientras la valuación corriente de los relojes satisfaga el invariante I_l.

1.2.4 Operaciones con autómatas

- **Bloqueo**

El bloqueo en una autómata es una acción no deseable, un autómata se dice está bloqueado cuando ocurre ya sea un bloqueo vivo o un bloqueo muerto. Un bloqueo muerto significa que hay un estado donde no hay eventos posibles, el autómata se atasca en un estado que no está definido como un estado final. Un bloqueo vivo significa que hay un conjunto de estados no marcados que no tiene transiciones fuera de este conjunto, el autómata queda en un ciclo infinito. La figura 1.10 muestra un estado con bloqueo muerto en el estado 5 y un estado con bloqueo vivo entre los estados 3 y 4 [21].

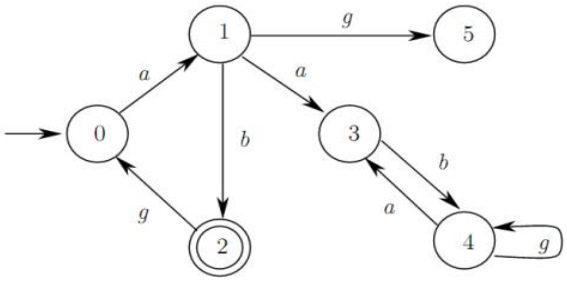


Figura 1.10. Bloqueo vivo y muerto en un autómata [21].

- **Autómata bien comportado**

Para definir un autómata bien comportado (*trim* en inglés) se debe entender el concepto de accesibilidad y coaccesibilidad; la accesibilidad implica que si se tiene un estado *x* es posible llegar a *x* desde el estado inicial x_o mediante alguna secuencia de símbolos. La coaccesibilidad implica que existe alguna ruta que lleve al autómata de *e* a algún estado terminal.

Entonces un autómata bien comportado es aquel que es accesible y coaccesible al mismo tiempo, la figura 1.11 muestra un autómata, el estado uno es accesible desde el estado 0 a través de la transición *a*, y es coaccesible al estado final a través de *b* [22].

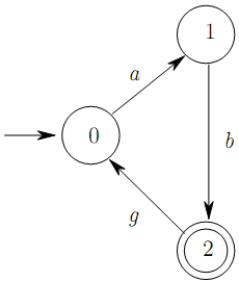
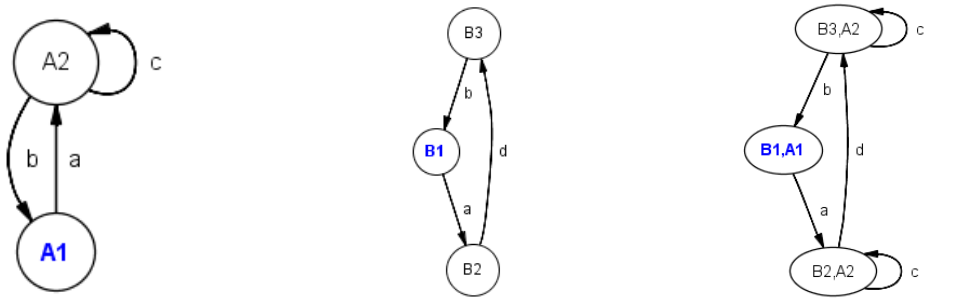


Figura 1.11. Autómata bien comportado [22].

- **Composición Paralela**

Muchas veces es posible descomponer un complicado problema en problemas más pequeños y más fáciles de tratar. Esto también es aplicable a los autómatas, un autómata puede descomponerse en varios autómatas más simples. Sin embargo (como se verá resolviendo los ejercicios de la práctica) no es trivial hacerlo, y hay que tener mucho cuidado de que la composición de las maquinas sencillas represente exactamente a él autómata descompuesto y no a un algo "parecido" [23].

En el ejemplo de la figura 1.12 se tiene dos autómatas el autómata *A* tiene dos estados (*A1, A2*) y tres transiciones (*a,b,c*), el autómata *B* tiene tres estados (*B1, B2, B3*) y tres transiciones (*a, b, d*), la operación composición paralela, comienza en los estados iníciales de cada autómata, este estado esta nombrado como (*B1, A1*), El paso al estado (*B2, A2*), sucede porque la transición *a* es verdadera en los dos autómatas, ya en este estado pueden pasar dos cosas, la primera es que *c* ocurra lo que resulta en un cambio nulo de estado en el autómata de la composición, la segunda es que *d* ocurra lo cual lleva al autómata de la composición a (*B3, A2*), en este punto pueden nuevamente ocurrir dos opciones, la primera es que *c* ocurra, lo que desencadena un cambio de estado nulo o puede ocurrir *b* que lleva al autómata a un estado inicial. Se debe observar que en el estado (*B2, A2*), la ocurrencia de *b* no lleva al autómata a su condición de inicio, debido a que en el autómata B primero tiene que suceder *d* para que esto suceda, por lo tanto *b* no puede ocurrir hasta que no esté el estado anterior habilitado en todos los autómatas.



Autómata A Autómata B c. Autómata A//B

Figura 1.12. Composición paralela [23].

1.3 Control supervisor y herramientas de simulación

Tradicionalmente, en la industria de procesos, los operadores se sitúan en el nivel jerárquico más alto de un sistema de control, asumiendo la responsabilidad de identificar sus estado de funcionamiento, y de iniciar acciones correctoras siempre que el dominio de la validez del control sea excedido o que un fallo impida su correcto funcionamiento. El objetivo principal de un sistema de supervisión es el asesoramiento de los operadores en su toma de decisiones (vigilancia) o la toma automática de decisiones (control).

1.3.1 Concepto de supervisor

El concepto general de supervisor, viene del latín “visus” que significa examinar un instrumento poniéndole el visto bueno; y del latín “super” que significa preeminencia o en otras palabras: privilegio, ventaja o preferencia por razón o mérito especial. Supervisión es, dar el visto bueno después de examinar y la supervisión tiene por objetivos básicos vigilar, algunos ejemplo de ello son vigilar (costo, tiempo y calidad) de algún proyecto que se esté desarrollando.

En el supervisor de un proceso se contemplan dos tareas principales: vigilancia y control supervisor. La vigilancia consiste en la adquisición y procesado de datos con el fin de tener un conocimiento actualizado sobre el estado de la planta. Así se debe identificar si el proceso está en un estado deseado o no. Entonces, en caso que sea necesario, el supervisor debe ser capaz de emprender las acciones oportunas, que le conduzcan al estado deseado. El control supervisor contempla precisamente esta capacidad de

reacción. Para ello será conveniente dotar al supervisor de cierta inteligencia y autonomía [26].

Un supervisor realizado con autómatas es capaz de vigilar el estado actual del proceso a partir de los datos de entrada, comparar el estado en que el sistema debería estar y tomar la acciones correctivas para llevar el proceso al estado deseado; para esto el autómata está en capacidad de habilitar o deshabilitar estados, de acuerdo a las necesidades del proceso.

1.3.2 Herramienta para diseño de supervisores.

DESUMA es una integración de la biblioteca UMDES, una biblioteca en c que permiter ralizar operaciones con autómatas y el entorno gráfico llamado GIDDES, una herramienta para la visualización de sistemas de eventos discretos, (escrito en Java), desarrollado en la Universidad de Mount Allison. DESUMA permite al usuario realizar una variedad de manipulaciones de los sistemas de eventos discretos modelados por la observación relacionada con la construcción de modelos, el diagnóstico de fallas, verificación, control total y parcial.

DESUMA está diseñado para respetar la interfaz de usuario de Windows directriz, por lo que pueden esperar los usuarios una interfaz de usuario intuitiva y familiar.

1.3.3 Herramienta para simulación de supervisores.

UPPAAL es un ambiente con herramientas integradas para el modelado, validación y verificación de sistemas de tiempo real. Los componentes del sistema de tiempo real son modelados por autómatas temporizados y los requerimientos de sintaxis temporal son expresados en la lógica TCTL, lógica que permite especificar los requerimientos temporales. Su funcionamiento es a través de un ambiente gráfico y permite el modelado, simulación y verificación de un sistema de tiempo real.

Presenta tres vistas principales que son:

- **System Editor**

El *system editor* se usa para crear y editar los sistemas a ser analizados. La descripción del sistema está definida por un conjunto de plantillas de proceso (Posiblemente con declaraciones locales), declaraciones globales, y

definiciones del sistema, posee un árbol de navegación en el panel izquierdo, y es usado para acceder a los diferentes componentes de descripción del sistema, la figura 1.13., muestra la pantalla del *system editor* [27].

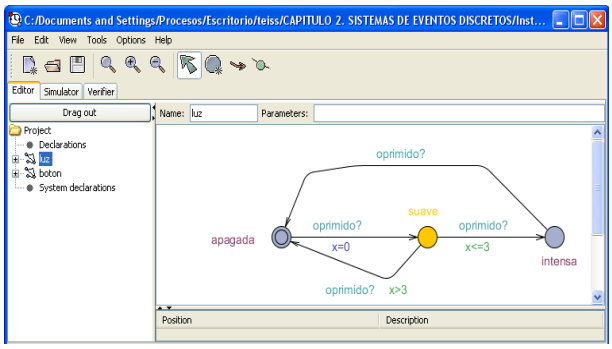


Figura 1.13. System Editor de UPPAAL [28].

- **Simulator**

El *simulator* es la herramienta de validación que permite examinar la posible dinámica en ejecución del sistema durante una etapa de diseño temprano (o modelado). Este suministra un medio económico de detección de fallas a priori, el simulador es también usado para visualizar ejecuciones generadas por el verificador. El *simulador* tiene cuatro paneles: el del extremo izquierdo se llama *simulation control*, el del medio es *Variables*, el de la parte superior derecha *processes* y el parte inferior derecha se llama *messagesequence chart*. La figura 1.14. muestra la pantalla del *simulator* de Uppaal[27].

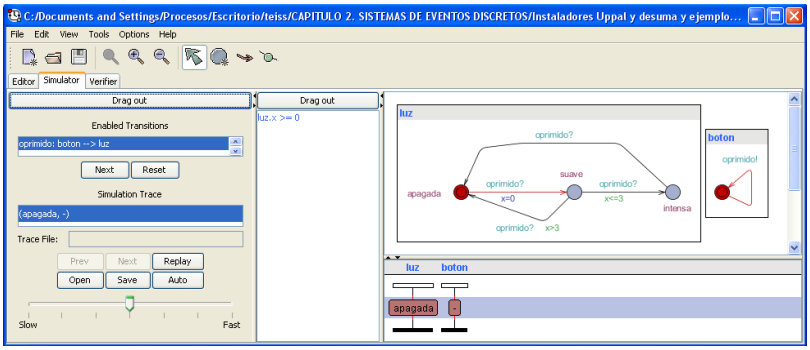


Figura 1.14. Simulator de UPPAL [28].

- **Verifier**

La pestaña *verifier* de uppaal permite chequear por medio de instrucciones el cumplimiento o no de una instrucción; esta instrucción puede corresponder bien sea, a determinar si existe un bloqueo o no dentro de un modelo, o a determinar si es posible llegar a un estado especifico bajo determinadas

condiciones sobre los relojes o variables. En la figura 1.15., se muestra la interfaz de la ventana *verifier* de Uppal[27].

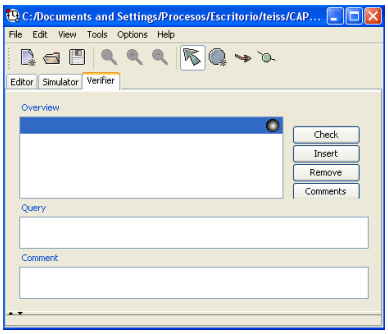


Figura 1.15. Verifier de UPPAL [28].

2 METODOLOGIA PARA EL DISEÑO DE SUPERVISORES

Gracias al alto grado de automatización en los procesos industriales en la actualidad, se ha producido un aumento en la demanda de eficiencia y calidad de las operaciones, lo que a su vez incrementa la posibilidad de fallos en los diseños dada la complejidad del análisis de sistemas a medida que crece el número de variables de entrada.

Anteriormente los operadores eran quienes se encargaban de identificar y solucionar inconvenientes que se presentaban durante las secuencias de operaciones del proceso. Sin embargo, en muchos casos esto no es suficiente ya que ante la complejidad de la estructura procesos y subprocesos, un operario humano puede no tener la capacidad de llevar la supervisión y actuación ante las posibles fallas en los procesos.

Así, se hace necesaria la implementación de métodos adecuados de vigilancia y control supervisor, capaces de detectar la aparición de situaciones potenciales de fallos y su ubicación, siendo este requerimiento indispensable para el establecimiento de las tareas y estrategias adecuadas de recuperación o de mantenimiento de los sistemas de producción [14].

En este capítulo se describe una metodología que permite el correcto diseño de supervisores automatizados mediante autómatas que sean capaces de llevar el registro de todos los estados posibles que el modelo de un proceso productivo puede llegar a alcanzar.

2.1 DESCRIPCION GENERAL DE UNA METODOLOGIA

Un aspecto fundamentar para poder diseñar un control supervisor es la vigilancia, En [26] se diseña una metodología que busca obtener un modelo supervisor vigilante de eventos discretos capaz de determinar el estado de funcionamiento de un proceso, de manera que si el sistema entra en un estado en fallo, el supervisor estará en capacidad de detectar el estado en fallo y emprender las acciones correctoras oportunas. Así, en función de las secuencias de eventos significativos que ocurran en el proceso, el supervisor puede determinar el estado de funcionamiento del proceso. Es decir, la aparición de uno o más eventos desencadena un procedimiento de respuesta.

Tradicionalmente, en la industria de proceso, los operadores se sitúan en el nivel jerárquico más alto de un sistema de control, asumiendo la responsabilidad de identificar su estado de funcionamiento, y de iniciar acciones correctoras siempre que el dominio de validez del control sea excedido o que un fallo impida su correcto funcionamiento. El objetivo principal de un sistema de supervisión es el asesoramiento de estos operadores en su toma de decisiones [10].

La metodología de modelado se basa en descomponer el modelo en varios autómatas finitos deterministas, uno por cada elemento del proceso, es decir, uno por cada sensor, actuador o elemento significativo, así por ejemplo, si un proceso consta de dos sensores de nivel, una motobomba, y tres electroválvulas se tendrán seis autómatas finitos deterministas que representan cada uno un elemento del proceso, sin embargo y como es lógico el proceso estará conformado por la interacción entre estos elementos, para lo cual se usara la función encardada de hallar todas las posibles combinaciones entre los estados de los autómatas conocida como composición paralela y que fue explicada con detalle en el capítulo 1.

Pero realizar la composición paralela de los autómatas correspondientes a cada estado no es suficiente puesto que el resultado puede contener estados inalcanzables o no deseados, para resolverlo la metodología propone realizar uno o varios autómatas que representen las restricciones del proceso, por ejemplo si en el proceso mencionado en el párrafo anterior se parte de que el tanque esta vacio, el sensor de nivel superior no podrá activarse si antes no ha sido activada la motobomba y la electroválvula, la motobomba permite que el agua suba desde el silo de almacenamiento hasta el tanque y la electroválvula habilita el paso del agua al silo al tanque, por lo cual si no se ha bombeado agua al tanque y no se ha habilitado el paso del agua no es posible que el sensor de nivel diga que el tanque llego al nivel superior.

En la metodología se propone utilizar el conocimiento heurístico que el experto posee para modelar el proceso mediante un autómata determinista. Este autómata permitirá diseñar un supervisor vigilante capaz de reconocer los estados de funcionamiento a partir de las secuencias de eventos significativos observados [26].

El objetivo de la metodología es sintetizar un modelo a eventos discretos del proceso. Este comprende no solo el propio proceso físico, con sus actuadores y sensores, sino también los sistemas que lo controlan. Es decir, un proceso se compone de depósitos, válvulas, motores, termopares, canalizaciones, reacciones químicas, reguladores PID, PLC, DSP, etc. En definitiva, el modelo debe representar la dinámica en lazo cerrado del proceso a supervisar [26].

2.2 PRESENTACIÓN DE LA METODOLOGIA EXISTENTE

La metodología descrita a continuación fue diseñada por Rammor Serrate Estruch de la Universidad Politécnica de Catalunya y por Josep Aguilar Martin del Laboratorio de arquitectura y de Análisis de Sistemas. Debido a la complejidad de los términos usados en la metodología original se considero conveniente expresarla en términos de fácil dominio para el lector. Sin embargo en anexo 3 se muestra la metodología en su forma original. Para proteger los derechos de autor los conceptos originales de la metodología se conservan en letra cursiva dentro de este documento.

Esta metodología, permite describir parcialmente diversos aspectos de la dinámica del proceso mediante el formalismo de autómatas y obtener el modelo del proceso al realizar la composición paralela de estos. Esta metodología se utilizará para obtener el autómata supervisor resultado de la composición paralela. A continuación se expresan los 7 pasos que la conforman.

En la metodología de supervisores, se utilizan los conceptos de generador y autómata supervisor. Con el fin de no causar confusiones a continuación se describe cada uno.

El nombre generador se usa para el autómata que representa al sistema aumentado en lazo abierto (descrito en el ítem 2.2.3) , es decir, el autómata obtenido después de la composición de los elementos. De esta manera, por ser el generador el resultado del modelo aumentado del sistema en lazo abierto, este generador puede producir cualquier palabra (secuencia de eventos) que podría ocurrir en el sistema que se está modelando.

Por otra parte, un autómata supervisor es un autómata que habilita o deshabilita la ocurrencia de ciertos eventos controlables que podría producir el generador, para que a pesar de la ocurrencia de cualquier evento no controlable, el sistema no llegue a estados prohibidos especificados de antemano. Un supervisor es entonces el equivalente a un controlador para un sistema de variables continuas.

2.2.1 Enumeración de los eventos significativos

El experto debe enumerar todos los eventos relativos al proceso que considere significativos, esto es el conjunto de sensores y actuadores que interactúan en el proceso, asignando un nombre a cada sensor o actuador del proceso y constituyendo el conjunto Σ_{PRO} conformado por estos.

A continuación se requiere determinar cuáles son controlables y cuáles no controlables. Los eventos controlables se originan en las señales medidas correspondientes a actuadores y se clasifican en el conjunto (Σ_C) y los no controlables se originan en las señales medidas correspondientes a sensores y se clasifican en el conjunto (Σ_{NC}).

2.2.2 Identificación y modelado de todos los componentes

El experto debe identificar los componentes del proceso. Un componente es un elemento que forma parte del proceso real. Son componentes tanto los actuadores, como los sensores, como otros elementos del propio proceso físico (por ejemplo, un motor o una reacción química).

Se supone que el experto conoce el comportamiento libre de cada componente, debiéndolo expresar mediante un generador. El comportamiento libre quiere decir el comportamiento del componente (sensor o actuador) *sin tener en cuenta la interacción con los otros componentes del proceso.* Así pues, el experto debe crear un generador por cada componente del proceso *Cada generador debe reproducir el comportamiento físicamente lógico del componente al que representa.*

2.2.3 Obtención del modelo aumentado de la planta en lazo abierto

Una vez obtenidos los generadores correspondientes a todos los componentes, debe procederse a su composición paralela, para obtener el modelo aumentado de la planta en lazo abierto. La composición paralela de todos los componentes descritos en el ítem 2.2.2 proporciona el modelo de la planta en lazo abierto.

Se denomina modelo aumentado ya que resulta de la composición del comportamiento libre de los componentes individuales (sin tener aun en cuenta la interacción entre ellos) y, por tanto, incorpora más comportamiento que el admisible realmente.

2.2.4 Identificación y modelado de todas las restricciones físicas

A continuación el experto debe identificar las restricciones físicas del proceso. Una restricción física vincula el comportamiento entre componentes debido a la física del proceso.

La convivencia de diversos componentes en un mismo proceso comporta un cierto grado de interacción entre ellos. Esta interacción se traduce en una alteración del comportamiento libre de los componentes individuales. Por ejemplo, una tubería establece un vínculo entre el caudal y la pérdida de presión.
Una restricción física establece las interacciones que surgen entre dos o más componentes debido a fenómenos propios del proceso físico. La física del proceso regula la admisibilidad de los eventos no controlables. Por esto, una restricción física se expresa en base a impedir o permitir la ocurrencia de eventos no controlables.

Se supone que el experto conoce todas las interacciones entre componentes, pudiéndose expresar mediante uno o varios generadores.

Se requiere que la restricción física cubra todos los eventos asociados a los componentes que relaciona. Es decir, existen dos maneras de obtener un generador de una restricción física. La primera consiste en partir del generador resultante de la composición de los componentes de los que se quiere acotar su comportamiento. Entonces, conociendo el significado de cada estado se trata de eliminar transiciones no controlables donde convenga. La segunda consiste en partir del generador de uno de los componentes. Entonces, conociendo el significado de cada estado se trata de añadir transiciones reflexivas no controlables en aquellos estados donde convenga.

En determinadas circunstancias, una restricción física impone una dinámica que se podrá asimilar al comportamiento de un componente más. Por ejemplo, la tubería anterior. No importa si esta dinámica se asocia a una restricción física o a un componente. Lo importante es que sea tenida en cuenta de alguna manera.

2.2.5 Obtención del modelo de la planta en lazo abierto

Para obtener el modelo de la planta en lazo abierto, es necesario realizar la composición paralela del modelo aumentado de la planta en lazo abierto (obtenido en el ítem 2.2.3) con todos los generadores que contienen las restricciones físicas del proceso (obtenidos en el item 2.2.4).

Normalmente esta operación supone una reducción en el número de estados y de transiciones del modelo aumentado de la planta en lazo abierto.

2.2.6 Identificación y modelado de todas las restricciones de control

A continuación el experto debe identificar las restricciones de control del proceso. Una restricción de control vincula el comportamiento entre componentes debido a las especificaciones de control deseadas.

La tarea del sistema de control es limitar el comportamiento del proceso en lazo abierto de acuerdo con unas especificaciones de control preestablecidas. Asi, una restricción de control limita la interacción entre dos o más componentes. Por ejemplo, se desea parar un motor solo si se alcanza una velocidad de rotación máxima. En este caso se trata de una restricción controlable porque busca la admisibilidad de un evento controlable (el motor), Por tanto, el autómata acotara la interacción entre la alimentación del motor y la medida del taco-dinamo.

Esta limitación se consigue regulando la admisibilidad de los eventos controlables. Por esto, una restricción de control se expresa en base a impedir o permitir la ocurrencia de eventos controlables. Se supone que el usuario conoce todas las interacciones entre componentes limitadas por el sistema de control, pudiéndose expresar mediante uno o varios generadores.

Igual que ocurría con las restricciones físicas, existen también dos formas de obtener un generador de una restricción de control. La primera consiste en partir del generador resultante de la composición de los componentes de los que se quiere acotar su comportamiento. Luego, conociendo el significado de cada estado, se trata de eliminar transiciones controlables donde convenga. La segunda consiste en partir del generador de uno de los componentes. Entonces, conociendo el significado de cada estado, se trata de añadir transiciones reflexivas controlables en aquellos estados donde convenga.

Las restricciones de control solo expresan algunas de las especificaciones de control: aquellas que se pueden describir fácilmente en base al ordenamiento de eventos, Otras especificaciones de control basadas en criterios de optimización o temporales, como un algoritmo PID, por ejemplo, no se podrán describir mediante una especificación de control. Por tanto, estos criterios de control no podrán acotar el comportamiento de la planta.

2.2.7 Obtención del modelo de la planta en lazo cerrado

La composición paralela del modelo de la planta en lazo abierto (generador obtenido en el ítem 2.2.5) con las restricciones de control (generadores obtenidos en el ítem 2.2.6), proporciona el modelo de la planta en lazo cerrado.

Se llega así al final de la metodología que debe proporcionar un modelo a eventos discretos del proceso. Dicho modelo contemplara la dinámica de los componentes, de las restricciones físicas y de las restricciones de control.

Como se ha visto hasta ahora, el objetivo de la metodología es obtener un autómata finito capaz de reconocer los eventos significativos que se producen en el proceso e identificar los estados de funcionamiento del mismo. Se obtiene así un autómata, listo para ser integrado dentro de un sistema de supervisión el cual determine el estado en el que se encuentra el proceso.

El objetivo de nuestro proyecto plantea este resultado solo como una primera fase, consistente en obtener el supervisor del proceso. Sin embargo, el supervisor obtenido deberá permitir incluir restricciones de tiempo y contadores. Dado que la metodología trabaja con el concepto de autómatas finitos y no con el concepto de autómatas finitos extendidos, esta se considera como una primera ampliación y será describe en el capítulo 4.

2.3 APLICACIÓN DE LA METODOLOGIA [1] EN LA CONSTRUCCIÓN DE AUTOMATAS SUPERVISORES DE PROCESOS QUE USEN CONTADORES Y CONDICIONES DE TIEMPO.

Al estudiar la metodología y tratar de usarla en procesos en los que el tiempo juega un papel importante, se obtuvieron los siguientes resultados:

- Se logro determinar que la aplicación de la metodología [26] permite diseñar un supervisor a través de autómatas finitos, pero no abarca el uso de temporizadores, ni contadores y, por lo que en su forma original no es aplicable al desarrollo de supervisores con PLC en procesos industriales.

- La metodología por su misma naturaleza no considera las condiciones sobre contadores, por ejemplo, suponga que en un proceso de empaque específico se necesita que cuando un contador de cajas llegue a un determinado número se active una banda transportadora que lleva ese número de cajas a la sección embalaje. Para lograr representarlo, se debería incluir el uso de operaciones de comparación tales como <,>,o = que permitan verificar el estado de un contador, en el ejemplo en mención, asumamos que el contador de cajas tiene el nombre CONT_CAJAS se debe incluir una condición como: si CONT_CAJAS=24 entonces que ejecute una acción o pase al siguiente estado.

- Los alcances de la metodología llegan hasta proporcionar el modelo a eventos discretos del sistema, donde se contemplara la dinámica de los componentes, de las restricciones físicas y de las restricciones de control, sin embargo el alcance de nuestra proyecto va mas allá, busca que una vez se tenga el supervisor, es decir, el modelo a eventos discretos, se pueda generar de manera automática el código de programación LADDER entendido por el PLC Micrologix 1500.

2.4 COMPONENTES DE LA METODOLOGIA PROPUESTA Y SOLUCIÓN PARA QUE LA METODOLOGIA [26] PUEDA SER USADA PARA CONSTRUIR AUTOMATAS SUPERVISORES, EN PROCESOS EN LOS QUE EL TIEMPO JUEGA UN PAPEL IMPORTANTE.

Para hacer posible incluir tanto condiciones de tiempo como condiciones de contadores y comparadores a la hora de diseñar un autómata supervisor, se hace una primera ampliación a la metodología que busca usar el concepto de autómatas extendidos para ampliar la metodología, en vez de usar el concepto de autómatas finitos. Esta ampliación se describirá con detalle en el ítem 2.5.

En los ambientes industriales es común que se programen secuencias de operaciones de supervisión utilizando Controladores Lógicos Programables (PLC). La no utilización de métodos formales para modelado del sistema, el diseño y la validación del software para el PLC, es fuente de errores y dificultades tales como la falta de portabilidad y confiabilidad para los programas desarrollados en los PLC [28]. Las herramientas actuales no proponen ninguna metodología para el diseño del programa y por lo tanto no se proponen métodos sistemáticos e integrales de diseño, donde se contemplen aspectos tales como la formalización, la validación y verificación del programa diseñado, dando como resultado algoritmos erróneos, soluciones que dependen de estilos de programación y dificultades para hacer modificaciones y mantenimiento de los programas [28].

A continuación se mencionan los componentes a tener en cuenta para el diseño de la metodología propuesta en este proyecto, que incluye no solo la construcción del modelo supervisor sino la validación y verificación del mismo, y la generación del código LADDER.

Según [29] la especificación y diseño de controladores lógicos para automatismos secuenciales debe incluir:

- Un formalismo matemático, con una sintaxis y una semántica bien definidas para expresar la parte estructural del modelo y el comportamiento dinámico sin ambigüedades.
- Una herramienta formal para realizar el análisis estático del modelo, mediante validación de propiedades.
- Una plataforma de simulación para efectuar la validación dinámica del modelo.
- Un método para generar automáticamente el código de programación del controlador lógico.

Dado un proceso, el objetivo de la teoría es el diseñar un supervisor de manera que el proceso sea supervisado, es decir, el sistema DES se acopla con el supervisor, y debe comportarse de acuerdo a varias restricciones deseadas. Se puede sintetizar un supervisor tal que el comportamiento del proceso supervisado resultante cumpla con tres condiciones: 1) no contradiga algunas especificaciones del comportamiento; 2) que el sistema resultante alcance las especificaciones; y 3) no contenga bloqueos. La teoría puede ser extendida para cubrir supervisión modular, descentralizada y jerárquica [30].

En este tipo de supervisores, el estado deseado del sistema se sintetiza en un controlador, que es a su vez un DES que se representará mediante un autómata que genere el lenguaje que supervisa al modelo del sistema. El "sistema supervisor" incluye un proceso de decisión discreto que es típicamente un sistema de eventos discretos representado por un autómata finito o una red de Petri [31].

En este trabajo de tesis se explorará la aplicación de la metodología general en [26] al caso particular de la programación de PLC para el diseño de supervisores para procesos de automatización típicos, desarrollando herramientas software tanto para la implementación de la metodología como para la generación automática de código para la programación del PLC; con lo cual se contribuye a llenar el vacío metodológico existente previo a la programación. El disponer de un formalismo matemático para guiar ese diseño, tiene la ventaja de posibilitar el uso de técnicas de análisis para detectar propiedades de alcanzabilidad de los estados deseados y posibles bloqueos del sistema.

2.5 METODOLOGÍA PROPUESTA

A continuación se describe una metodología que busca obtener un modelo supervisor vigilante de eventos discretos capaz de determinar el estado de funcionamiento de un proceso, de manera que se puedan evaluar todas las posibilidades dentro de un proceso y en caso de que sea necesario, emprender las acciones correctoras oportunas. Así, en función de las secuencias de eventos significativos que ocurran en el proceso, el supervisor puede determinar el estado de funcionamiento del proceso. Es decir, la aparición de uno o más eventos desencadena un procedimiento de respuesta.

2.5.1 Enumeración de los eventos significativos y clasificación

El diseñador debe enumerar todos los eventos relativos al proceso que considere significativos, esto es el conjunto de sensores y actuadores que interactúan en el proceso, asignando un nombre a cada uno y constituyendo el conjunto Σ_{PRO} conformado por estos.

Por ejemplo, suponga que se considera dentro del proceso de un proceso, se van a numerar los eventos significativos correspondientes a una electroválvula (actuador) y a un sensor de nivel. En el actuador se identifican dos eventos: abrir la válvula y cerrar la válvula nombrados como Von y Voff. Ahora bien, para el sensor de nivel se identifican dos eventos significativos: TK2NS cuando el sensor indica que se superó el nivel superior dentro del tanque y TK2NI cuando el sensor indica que ingrediente del tanque se encuentra en el nivel inferior. Cada tanque contiene este par de sensores, en los cuales es relevante el valor activo de cada uno para determinar si el ingrediente dentro del tanque ha alcanzado el valor superior o inferior. Así el conjunto de eventos significativos estaría formado por:

Σ_{PRO}= {Von, Voff, TK2NS, TK2NI,…}

A continuación se requiere determinar cuáles son controlables y cuáles no controlables u observables. Los eventos controlables se originan en las señales medidas correspondientes a actuadores y se clasifican en el conjunto (Σ_C) y los no controlables se originan en las señales medidas correspondientes a sensores y se clasifican en el conjunto (Σ_{NC}), en el caso de la electroválvula entonces pertenecería a al conjunto Σ_C por tratarse de un actuador y el sensor de nivel se clasificaría en el conjunto de los eventos no controlables.

Σ_c= {Von, Voff,…}

Σ_{NC}= {TK2NS, TK2NI,…}

2.5.2 Identificación y modelado de todos los componentes

Una vez identificados todos los eventos se debe hacer la identificación de los componentes del proceso. Un componente es un elemento que forma parte del proceso real. Son componentes tanto los actuadores, como los sensores, como otros elementos del propio proceso físico (por ejemplo, un motor o una reacción química).

Se supone que el diseñador conoce el comportamiento libre de cada componente, debiéndolo expresar mediante un autómata. El comportamiento libre quiere decir el comportamiento del componente (sensor o actuador) sin tener en cuenta la interacción con los otros componentes del proceso. Así pues, el experto debe crear un autómata por cada componente del proceso.
Para el caso de la electroválvula y el sensor se deberá construir un autómata con los eventos que estén asociados a cada uno de ellos, independientemente de la interacción que tenga con otros componentes.

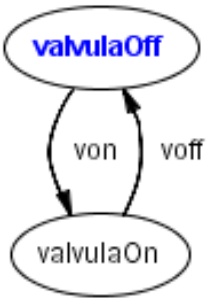


Figura 2.1 Ejemplo de un autómata con eventos controlables.

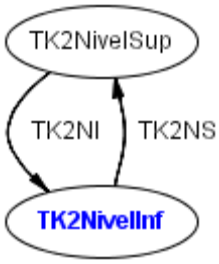


Figura 2.2 Ejemplo de un autómata con eventos observables.

2.5.3 Obtención del modelo aumentado de la planta en lazo abierto

Una vez obtenidos los generadores correspondientes a todos los componentes, debe procederse a su composición paralela, para obtener el modelo aumentado de la planta en lazo abierto. La composición paralela de

todos los componentes descritos en el ítem 2.5.2 proporciona el modelo de la planta en lazo abierto.
Se denomina modelo aumentado ya que resulta de la composición del comportamiento libre de los componentes individuales (sin tener aun en cuenta la interacción entre ellos) y, por tanto, incorpora más comportamiento que el admisible realmente. Para realizar la composición paralela use la herramienta DESUMA que se encuentra dentro de la carpeta *instaladores y* consulte el manual de usuario descrito en el anexo A.

2.5.4 Identificación y modelado de todas las restricciones físicas

A continuación el experto debe identificar las restricciones físicas del proceso. Una restricción física vincula el comportamiento entre componentes debido a la física del proceso.

La convivencia de diversos componentes en un mismo proceso comporta un cierto grado de interacción entre ellos. Esta interacción se traduce en una alteración del comportamiento libre de los componentes individuales. Por ejemplo, una tubería establece un vínculo entre el caudal y la pérdida de presión.
Una restricción física establece las interacciones que surgen entre dos o más componentes debido a fenómenos propios del proceso físico. La física del proceso regula la admisibilidad de los eventos no controlables. Por esto, una restricción física se expresa en base a impedir o permitir la ocurrencia de eventos no controlables.

Se supone que el experto conoce todas las interacciones entre componentes, pudiéndose expresar mediante uno o varios generadores.

Se requiere que la restricción física cubra todos los eventos asociados a los componentes que relaciona. Es decir, existen dos maneras de obtener un generador de una restricción física. La primera consiste en partir del generador resultante de la composición de los componentes de los que se quiere acotar su comportamiento. Entonces, conociendo el significado de cada estado se trata de eliminar transiciones no controlables donde convenga. La segunda consiste en partir del generador de uno de los componentes. Entonces, conociendo el significado de cada estado se trata de añadir transiciones reflexivas no controlables en aquellos estados donde convenga.

Una restricción física limita transiciones que físicamente no es posible que ocurra, por ejemplo para alcanzar el nivel superior dentro del tanque 2 es necesario que antes la válvula este encendida, suponiendo que el tanque se

encontraba inicialmente vacio. Así, por tratarse de una restricción física, se regula la admisibilidad del evento no controlable TK2NS.

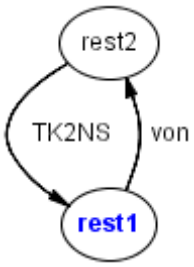


Figura 2.3. Ejemplo restricción física.

2.5.5 Obtención del modelo de la planta en lazo abierto

Para obtener el modelo de la planta en lazo abierto, es necesario realizar la composición paralela entre el modelo aumentado de la planta en lazo abierto (obtenido en el ítem 2.5.3) y los autómatas de las restricciones físicas del proceso (obtenidos en el ítem 2.5.4).

Normalmente esta operación supone una reducción en el número de estados y de transiciones del modelo aumentado de la planta en lazo abierto.

2.5.6 Identificación y modelado de todas las restricciones de control

A continuación el diseñador debe identificar las restricciones de control del proceso. Una restricción de control vincula el comportamiento entre componentes debido a las especificaciones de control deseadas.

Una restricción de control limita la interacción entre dos o más componentes. Esta limitación se consigue regulando la admisibilidad de los eventos controlables. Por esto, una restricción de control se expresa en base a impedir o permitir la ocurrencia de eventos controlables.

Se supone que el experto conoce todas las interacciones entre componentes limitadas por el sistema de control, pudiéndose expresar mediante uno o varios autómatas.

Igual que ocurría con las restricciones físicas, existen también dos formas de obtener un generador de una restricción de control. La primera consiste en que del autómata de lazo abierto se eliminen transiciones controlables donde convenga como muestra la figura 2,4 y la segunda consiste en construir un autómata que contenga la secuencia de los eventos controlables. Entonces, conociendo el significado de cada estado, se trata de añadir transiciones reflexivas controlables en aquellos estados donde convenga.

En la figura 2.4 se muestra una restricción de control obtenida al eliminar el evento “ApagarMB”, este evento fue eliminado puesto que para llegar a el se debió haber encendido la motobomba (es decir debió haber ocurrido el evento “prenderMB”) y no tendría sentido encender la motobomba y en el siguiente evento apagarla, al eliminar este evento, la secuencia de control sera correcta ya que una vez se encienda la motobomba, es posible permitir que ocurra el evento “energizarV1”. La segunda forma de hacer una restricción de control debe crearse un autómata que determine el orden de los eventos controlables, como se muestra en la figura 2.5, en esta se describe el orden en que deben ejecutarse estos eventos, asi, dado que el estado inicial es RC1, una vez ocurra el evento prenderMB, podrá ocurrir el evento energizarV1. Cualquiera de estas dos formas de crear una restricción de control es válida.

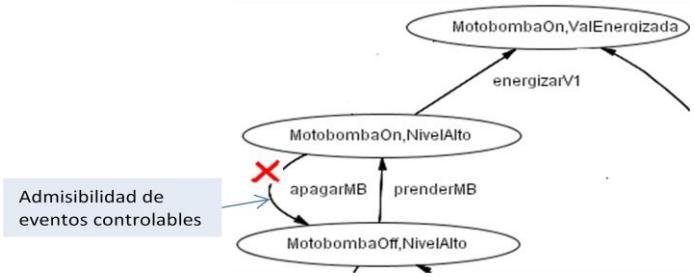


Figura 2.4 Ejemplo restricción de control mediante la eliminación de eventos.

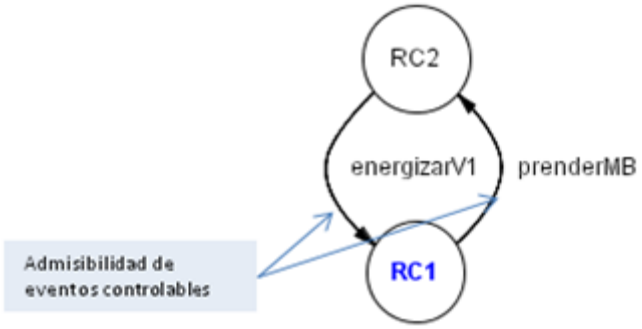


Figura 2.5. Ejemplo restricción de control como un autómata.

2.5.7 Obtención del modelo de la planta en lazo cerrado

Este paso solo se hace necesario en caso de que se hayan creado autómatas en las restricciones de control, asi la composición paralela del modelo de la planta en lazo abierto (generador obtenido en el ítem 2.5.5) con las restricciones de control (generadores obtenidos en el ítem 2.5.6), proporciona el modelo de la planta en lazo cerrado. Si los eventos

controlables han sido eliminados directamente del modelo de la planta en lazo abierto no será necesario realizar ninguna operación adicional.
Se llega así al final de la metodología que debe proporcionar un modelo a eventos discretos del proceso. Dicho modelo contemplara la dinámica de los componentes, de las restricciones físicas y de las restricciones de control.

Como se ha visto hasta ahora, el objetivo de la metodología es obtener un autómata finito capaz de reconocer los eventos significativos que se producen en el proceso e identificar los estados de funcionamiento del mismo. Se obtiene así un autómata, listo para ser integrado dentro de un sistema de supervisión el cual determine el estado en el que se encuentra el proceso.

El objetivo de nuestro proyecto plantea este resultado solo como una primera fase, consistente en obtener el supervisor del proceso. Sin embargo, el supervisor obtenido deberá permitir incluir restricciones de tiempo y contadores, lo cual se incluirá en el ítem 2.6.

2.6 USO DE LA HERRAMIENTA DESUMA PARA LA CONSTRUCCIÓN DEL AUTOMATA SUPERVISOR SIGUIENDO UNA SINTAXIS DISEÑADA PARA LA APLICACIÓN DE LA METODOLOGIA.

A continuación se propone una sintaxis creada para construir autómatas en DESUMA, esta sintaxis es necesaria cuando lo que se busca es obtener el código LADDER del supervisor, dicha sintaxis será interpretada por la herramienta que realiza la conversión de autómatas a código LADDER, por lo que se debe prestar especial atención en la sintaxis usada.

2.6.1 Sintaxis para la definición de los eventos significativos.

Como se observa en la figura 2.6, los nombres de los eventos significativos deberán siempre de estar precedidos por el símbolo underscore seguido del estado en binario al que se desea llevar el evento significativo.
Asi, se usara “_0”, para determinar el estado apagar, desenergizar o desactivar de un evento significativo, y “_1” para determinar el estado encender, energizar o activar asociado de un evento significativo

Por ejemplo, suponga que se tiene una electroválvula, en la cual pueden ocurrir dos eventos significativos, uno para energiza y otro que desenergizar la electroválvula. haciendo uso de la metodología el usuario deberá, definir estos dos eventos con el mismo nombre puesto que están asociado al mismo componente, salvo que deberá precederlo de "_0" para indicar el estado desenergizar y "_1" para indicar el estado energizar.la electroválvula.

La figura 2.1 se ilustra el autómata resultante de este caso, donde el estado inicial del autómata es *valvula10ff,* en este estado es posible que ocurra el evento *v1_1* el cual hace que la válvula tome el valor binario de 1, es decir que se energice; una vez suceda esto, se llega al estado *Valvula1On* que indica que la electroválvula ya ha sido energizada y por lo tanto puede ocurrir el evento *V1_0,* que cambia el valor de la salida de 1 a 0, desenergizando la válvula y permitiendo volver al estado inicial. *Valvula1Off.* En este ejemplo se ve claramente, como los dos eventos V1_0 y v1_1 aun cuando son diferentes, están asociados a un mismo actuador y por tanto a una misma salida física del PLC.

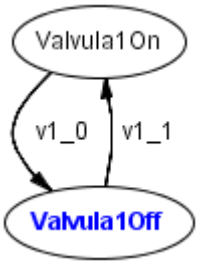


Figura 2.6 Ejemplo de definición de un evento significativo en DESUMA [28].

Lo anterior garantiza que la herramienta FSMLADDER interprete esta sintaxis, y asocie los dos eventos significativos a una única salida en el PLC (la salida correspondiente a la electroválvula en este caso), considerando el efecto energizar o desenergizar que ocasiona cada evento sobre las salida. De igual como se define un evento significativo asociado a un actuador, deben definirse los eventos significativos asociados a un sensor y a una variable.

La herramienta FSMLADDER, es capaz de reconocer estos dos eventos V1_0 y V1_1 como dos eventos diferentes, pero asociarlos a una misma salida en el PLC a la hora de generar el código LADDER, por lo que es indispensable para el correcto funcionamiento de la metodología que el usuario siempre utilice esta sintaxis, independientemente de si se trata de una entrada, una salida o un bit, deberá decir a que estado binario se desea llevar el evento significativo.

2.6.2 Sintaxis eventos significativos asociados a una variable, propuestos como ampliación a la metodología existente.

Como vimos en el apartado 2.5 según la metodología original, un evento significativo se clasifica en controlable y no controlable, donde los eventos significativos asociados a un actuador son controlables y los asociados a un sensor son los no controlables. Los eventos significativos observables, deben ser marcados como "observe" en el campo "events" de DESUMA como ilustra la figura 2.7(a) y los controlables como "control" como ilustra la figura 2.7. (b).

Ahora bien, el ampliar la metodología para el uso de autómatas finitos extendidos involucra grandes cambios, en primer lugar, hay que incluir una nueva clasificación de los eventos significativos, que hemos llamado como ***evento significativo asociado a una variable***, consistente en un evento que no depende de una entrada o una salida física del proceso; es decir, su ejecución no involucra la activación o desactivación de un actuador (evento controlable), ni tampoco depende de la activación de una entrada física o sensor (evento observable), sino, que estos significativos son usados para entradas o salidas virtuales, es decir, entradas y salidas que no existen físicamente, pero pueden ser simuladas mediante el sistema de supervisión SCADA o simplemente utilizadas en el código LADDER.

Los eventos significativos asociados a una variable pueden corresponder bien sea a un actuador virtual o a un sensor virtual. Un ejemplo del primer caso podría ser una alarma virtual, se dice que es virtual porque el estado que enciende o apaga la válvula no es un evento significativo que corresponda a un actuador físico, sino que solamente va a ser simulado o visualizado en el LADDER. Este tipo de eventos significativos deben ser marcados en la herramienta DESUMA, como se muestra en la figura 2.7. (c), donde se observa que en la pestaña *events*, que los eventos son marcados como observables y controlables.

Por otra parte, un evento significativo asociado a una variable puede en vez de representar un actuador simulado (como en el caso de la alarma), representar un sensor simulado. En este último caso, supóngase que se necesita simular el comportamiento de un sensor de presencia (bien sea representado por un botón en el SCADA o por un bit dentro del código LADDER), donde el estado activo o inactivo dependerá del valor (1 o 0) dado por el usuario. Dado que el sensor no existe físicamente se deberá usar el concepto de *evento significativo asociado a una variable,* y para diferenciarlo del caso anterior, no deberán ser marcados los iconos *control* y *observe*, en la pestaña *events* de la herramienta DESUMA, tal como muestra en la figura 2.1 (d).

En conclusión, una primera ampliación de la metodología corresponde en clasificar los eventos significativos en tres tipos: controlables, observables y asociados a una variable, y en definir una sintaxis que le permita al usuario diferenciarlos en la herramienta DESUMA; esto garantiza la correcta interpretación de las entradas, salidas y variables en herramienta FSMLADDER, y su asignación como “output”, “input” o “bit” respectivamente en la herramienta Rs Logix (Herramienta software en la que se programa el PLC, usando el lenguaje LADDER).

Figura (a). Evento significativo observable (sensor)

Figura (b). Evento significativo controlable (actuador).

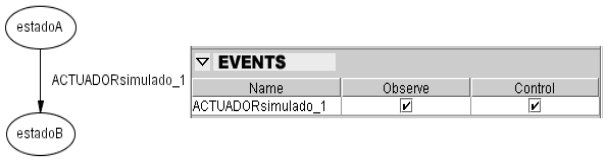


Figura (c). Evento asociado a una variable (actuador simulado).

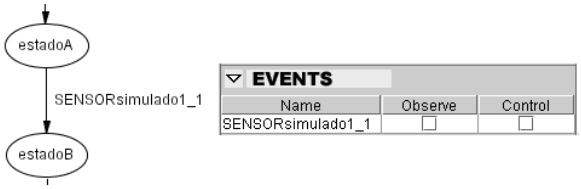


Figura (d). Evento asociado a una variable (sensor simulado).

Figura 2.7 Clasificación de los eventos significativos en DESUMA [28]

2.6.3 Temporizadores propuestos como ampliación a la metodología existente.

Los temporizadores como se menciono el capitulo anterior son un conjunto finito de variables reales, llamadas relojes, cuyos valores se incrementan uniformemente con el paso del tiempo.

En esta metodología se clasifican los temporizadores de acuerdo a el tipo de evento al que este asociado, así, un temporizador puede estar asociado bien sea a un actuador -lo que permite que el actuador se mantenga activo o inactivo durante determinado tiempo -, o bien, estar asociado a una variable, -lo que permite de igual manera que un actuador simulado o virtual se mantenga activo o inactivo durante un lapso de tiempo determinado por el usuario -.

Esta diferencia entre los dos tipos de temporizadores es indispensable a la hora de definirlos en la herramienta DESUMA, debido a que son estos, los parámetros que le indican a la herramienta FSMLADDER si el temporizador va a actuar sobre una salida del PLC o sobre un bit. En caso de tratarse de una salida del PLC (es decir de tratarse de un evento controlable o actuador) la definición del temporizador deberá ser marcada como tal, es decir como *control* en el campo *events* de DESUMA como se indico en el ítem 2.6.1; por otra parte, en caso del de tratarse de un temporizador de un evento representado por un bit en el PLC (es decir un evento significativo asociado a una variable o actuador virtual) el temporizador deberá ser marcado en los campos *control* y *observe* en DESUMA como se especificó en el ítem 2.6.1.

El funcionamiento de un temporizador es el siguiente: El actuador o la variable que se va a temporizar se mantendrá en el estado 1 o 0 según sea el caso (use _0 para mantener en estado 0 y _1 para mantener en estado 1) por un tiempo igual al tiempo especificado en la definición del temporizador (ver sintaxis). Una vez se inicie la temporización el autómata pasara inmediatamente al estado siguiente, si es necesaria su culminación para pasar al siguiente evento, entonces se deberá agregar un evento posterior con una condición sobre el temporizador que lo garantice.

El motivo por el cual no se espera a que pase el tiempo de temporización para pasar al siguiente estado, se debe a que para muchos de los casos en los que fueron aplicados los temporizadores, ocurría que mientras el temporizador estaba activo no se podía atender ningún otro evento, es decir, si ocurre un evento critico para el proceso que deba ser atendido en cualquier momento, por ejemplo que se active un sensor de nivel que indica que un tanque llego a un nivel crítico, el autómata supervisor debe estar en la capacidad de atender este evento y tomar las medidas correctivas para las que fue programado.

Un temporizador en ningún caso podrá estar asociado a un sensor, puesto que el sensor es un evento no controlable y por lo tanto no es posible manipular su tiempo de activación como en el caso de los actuadores.

A continuación se describe la sintaxis de los temporizadores y se da un ejemplo básico de aplicación de cada caso.

- **Sintaxis**

NombreTemporizador/ValorPreseleccionado/$

Cada definición de un temporizador consta del nombre del evento significativo controlable o el nombre del evento significativo asociado a una variable (solo para el caso de actuadores virtuales), un entero y el símbolo $.
El entero almacena el valor preseleccionado, correspondiente el valor al cual debe llegar el temporizador antes que caduque el tiempo de espera del mismo. El rango del este entero es de 0 a 32767 (máxima capacidad del temporizador en Rs Linx de Rockwell) y se retiene hasta que es restablecido por una instrucción reset.

Ejemplo de temporizador asociado a un evento controlable:

Supóngase que dentro de un proceso se desea activar durante 120 segundos un mixer como muestra la figura 2.8. Así, cuando el autómata alcance el estado llamado como *estadoA,* el mixer se encenderá, se iniciara el tiempo de temporización y se pasará inmediatamente al estado siguiente (*estadoB)*.
En el caso que se quiera esperar al que el mixer acabe el tiempo de temporización para realizar determinada acción, se deberá crear un evento posterior con una condición de temporizador, como se muestra en el ítem 2.6.5.
Lo que garantiza que el mixer va a pasar al estado encendido es " _1" que se hace parte del nombre del mixer, si por el contrario se necesitara que en determinado momento el mixer llegara a una acción de reposo apagándolo, se utilizaría la sentencia _0, es decir el evento seria mixer_0/120/$ en vez de mixer_1/120/$.

Figura 2.8 Ejemplo de temporizador asociado a un evento controlable [28].

Ejemplo de temporizador asociado a una variable:

Ahora suponga el caso, que dentro de un proceso una vez se adicionaron todos los ingredientes se llego al estado *RecipeCompleto* como se ilustra en la figura 2.9, sin embargo antes de pasar al siguiente estado denominado *Reacción completa,* debe ocurrir una reacción química que tarda una hora (3600 segundos), por lo que el usuario deberá crear una variable de temporización, ej. Temp1/3600/$. En este caso, a diferencia del ejemplo anterior, no se necesita energizar o desenergizar un actuador real por determinado tiempo, sino que se puede crear un actuador no real que se active por determinado tiempo y permita garantizar la espera de 3600 segundos que tarda la reacción.

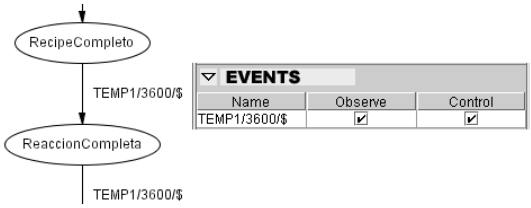


Figura 2.9 Construcción de un temporizador [28].

2.6.4 Contadores

El contador es un elemento capaz de llevar el cómputo de las activaciones de sus entradas, por lo que resulta adecuado para memorizar sucesos que no tengan que ver con el tiempo pero que se necesiten realizar un determinado número de veces.

- **Sintaxis**

NombreContador/ValorPreseleccionado/+o –

Cada definición de un contador consta de una palabra, un entero y el símbolo + o -, todo separado por un slash.

Donde la palabra (**NombreContador**) corresponde al nombre del evento significativo seguido por el símbolo underscore y el estado en binario (1 o 0) al que se desea llevar el evento significativo.
Asi, si el evento significativo esta precedido de “_0”, indica que el evento se llevara al estado 0 (apagar, desenergizar o desactivar) en el caso de tratarse de un actuador o que se espera a que la entrada tome el valor de cero en el caso de un sensor; una vez suceda esto el contador asociado a este evento incrementara o decrementará en uno su valor.

El entero almacena el valor preseleccionado, correspondiente el valor al cual debe llegar el contador. El rango del este entero es de -32768 a 32767.y el símbolo (+ o -) indica el conteo ascendente (+) o descendente (-) de los eventos.

Para el uso adecuado del contador internamente se maneja un acumulador el cual contiene el conteo actual. El valor acumulado aumenta (/+) o disminuye (/-) cada vez que ocurre un evento significativo. El valor acumulado podrá llegar hasta el valor preseleccionado y se retiene hasta que es restablecido por una instrucción reset con la misma dirección que el contador. El valor de conteo debe permanecer en el rango de -32,768 a +32,767.

- **Clasificación**

Se propone clasificar los contadores en en tres grupos dependiendo la naturaleza del proceso. Un contador podrá ser controlable, observable o asociado a una variable. A continuación se describen con detalle cada una de las clasificaciones.

Contador de un evento controlable: Este clasificación de contadores permite al usuario crear un contador ascendente o descendente de las veces que se energiza o desenergiza un actuador, es decir a un evento controlable. El contador aumenta o disminuye su valor en una unidad cada vez se active el evento controlable al que está asociado y cuenta hasta llegar al valor preseleccionado.

Contador de un evento no controlable u observable: permite al usuario crear un contador ascendente o descendente de las señales de entrada de un sensor, es decir a un evento no controlable. El contador aumenta o disminuye su valor en una unidad cada vez se active el evento controlable al que está asociado y cuenta hasta llegar al valor preseleccionado.

En la figura 2.10 se observa un contador de cajas ascendente que cuenta de uno en uno hasta llegar a 24 con el fin de que cuando existan 24 numero de cajas de determinado producto pasen a la etapa de embalaje, así cada vez que ocurra la transición “SensorCP_1” que corresponde a la activación de un sensor de cajas pequeñas, el cual debe estar definido en otro autómata, el contador de cajas pequeñas irá incrementando en uno; una vez el conteo llegue a 24 se activará el evento “SensorCP_1/24/+” y permitirá que se avance a el “Estado2” que indica que ya están listas las 24 cajas para ser embaladas.

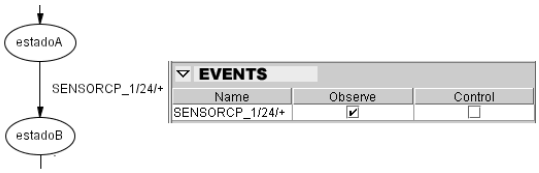


Figura 2.10 Construcción de un contador [28].

Contador asociado a un estado: permite al usuario crear un temporizador que no está asociado ni a un sensor ni a un actuador, sino que está asociado al evento inmediatamente anterior, es considerado como un contador de estados. El contador aumenta o disminuye su valor en una unidad cada vez se active el estado anterior y llega hasta al valor preseleccionado. Es decir cada que pasa el evento Cont1 en otro autómata u otros autómatas la variable Cont1 se incrementará en uno, en la figura 2.11 por ejemplo solo hasta que Cont1 alcance el valor preseleccionado de 12, el autómata cambiara del "Estado1" a "Estado2".

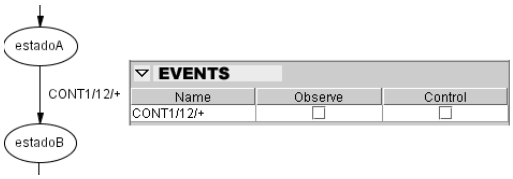


Figura 2.11 Construcción de un contador [28].

Instrucción de Reset: La instrucción reset restablece el valor de los contadores. Para reinicializar el valor de un contador basta con colocar el nombre del contador entre paréntesis. (nombreContador).

Para reinicializar el contador de cajas de la figura 2.12 se debe colocar el Nombre del temporizador, el cual puede ser un evento controlable, no controlable u observable o ni controlable ni observable, entre paréntesis.

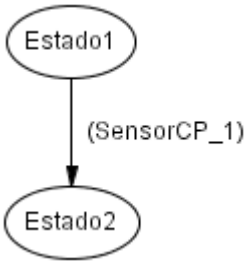


Figura 2.12. Construcción de un reset [28].

2.6.5 Comparadores

Los comparadores nos permiten condicionar el valor de un contador mediante el uso de símbolos matemáticos SM como =,<,o >.

- Determinar si un valor de contador es igual que un entero m. (==) ej. [Cont == m]
- Determinar si un valor de contador es menor que un entero m. (<) ej. [Cont <m]
- Determinar si un valor de contador es mayor que un entero m. (>) ej. [Cont >m]

Los rangos válidos de estas instrucciones son: De -32768 a 32767 (palabra) y de -2,147,483,648 a 2,147,483,647 (palabra larga).

Sintaxis.

[NombreContador SM n]

Donde NombreContador es un arreglo de caracteres, SM corresponde al símbolo matemático ==,<,o > y n es un entero.

Condición de comparador como evento significativo.

[NombreContador1==n]

Use esta sintaxis para el uso de una condición del comparador como un evento significativo. Esto permite que si se cumple la condición de que NombreContador==n pase al siguiente estado.

2.6.6 Conjuntos de condiciones.

Los conjuntos de condiciones como parte de un solo evento nos permiten que dentro de un solo evento o transición se incluyan varias condiciones de contadores y solo una condición asociada a un evento significativo, es decir a un sensor, o un actuador. Existen varias posibilidades a la hora de formar conjuntos de condiciones, las más usadas son:

- **Combinaciones de condiciones de comparador:** Se utiliza cuando en un autómata no se pueda pasar al siguiente estado hasta cuando un conjunto especifico de contadores lleguen a un valor especifico, para determinar si un contador llega a un valor especifico se utilizan comparadores, por lo que lo hemos denominado condiciones de

comparador, así, solo se hará verdadero el evento o transición y pasara al siguiente estado, si todo el conjunto de comparadores asociados a los contadores han llegado al valor definido, esto se debe a que las condiciones están unidas con una conjunción lógica and.

Para definir un evento formado por un conjunto de condiciones and, basta con colocar cada condición de contador entre corchetes, seguida una de otra [NombreContador1==x][NombreContador2==m] [NombreContadorn==p].

- **Combinaciones entre comparadores y un evento significativo:** a diferencia de la condición anterior, se le adiciona la ocurrencia de un evento significativo bien sea controlable, observable o asociado a un estado.
  Para el caso de un evento significativo observable o no controlable solo podrá pasar al siguiente estado si se cumplen todas las condiciones de comparador asociadas a los contadores y además ocurre un evento significativo observable es decir, se energiza o desenergiza un sensor.

  Si por el contrario se trata un evento significativo controlable solo podrá pasar al siguiente estado si se cumplen todas las condiciones de comparador asociadas a los contadores y además es posible energizar o desenergizar un actuador, el termino posible se debe a que como se verá más adelante, al incluir restricciones al modelo, no siempre es posible que ocurra un evento.

2.7 SIMULACIÓN Y VERIFICACIÓN MEDIANTE LA HERRAMIENTA UPPAAL DEL AUTOMATA SUPERVISOR OBTENIDO

Hasta ahora hemos usado La aplicación software DESUMA para diseñar y aplicar la operación de composición paralela entre los autómatas de los eventos controlables, no controlables y de las restricciones tanto físicas como de control, obteniendo un modelo en lazo cerrado de la planta. A continuación y siguiendo la metodología el usuario deberá verificar si lo obtenido con la composición paralela es el resultado deseado del comportamiento del proceso, de no serlo deberá agregar nuevas restricciones en DESUMA o eliminar aquellas que no permitan llegar a los resultados esperados.

Para lograr esta conversión se diseño la herramienta FSMLADDER que permite en primera instancia llevar el supervisor obtenido, del formato de DESUMA al formato de UPPAAL, donde se comprobará mediante sus

herramientas de verificación extendidas que no existen bloqueos ni puntos muertos. Esta herramienta además es capaz de reconocer la composición paralela ya simulada y verificada y convertirla al lenguaje de programación en escalera LADDER para finalmente ser llevada a un PLC Micrologix 1500, previa asignación de la dirección en el PLC de los bits, las entradas y las salidas por el usuario.

A continuación se describen las sentencias utilizadas en el Query para realizar la verificación del modelo del autómata supervisor.

Verificación de un modelo.

La verificación de un modelo en UPPAAL se realiza básicamente mediante la evaluación de fórmulas de la lógica TCTL, a través de la herramienta *verifier* mostrada en la figura 1.15.

Para realizar la verificación se utilizan tres comandos básicos:

- A[] P Siempre P
- E<> P Existe la posibilidad P
  Donde P es una proposición o conjunción de proposiciones lógicas.

- A[] *deadlock* Significa que el sistema se bloquea.
- *A[] not deadlock* el sistema nunca se bloquea
- E<>. Not deadlock no existe la posibilidad de que autómata se bolquee. deadlock sólo puede ser usado junto a A[] o E<>.

Estas sentencias deberán ser incluidas en la pestaña Query, mostrado en la figura 2.13, en caso de el led indicador mostrarse como verde indicara que la condición se satisface (es verdadera).

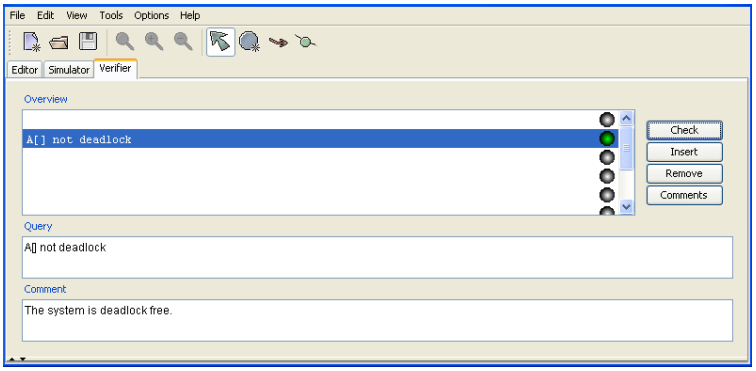


Figura 2.13 verificación de no bloqueos mediante el Query de UPPAAL

Para hacer el chequeo de los temporizadores se utilizaran condiciones sobre los mismos, en el ejemplo de la figura 2.14 se observa que se tiene un

temporizador y que se ha definido con un tiempo igual a 20 segundos y necesitamos verificar si el temporizador ha llegado a un valor o no mediante las sentencias Query.

Asi, se definen dos instrucciones básicas, para realizar las preguntas en la interfaz.

- A[] not (estadoC and t<20)

se cumple si desde cada estado alcanzable se cumple que no puede llegarse al estadoC y que el reloj de tiempo (t) sea menor que 20.

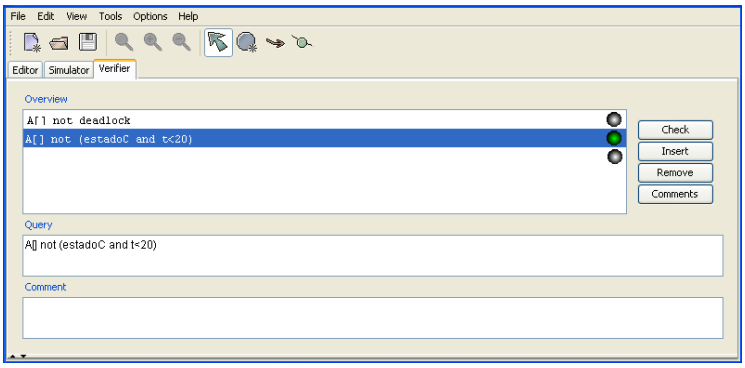


Figura 2.14. Verificación de que no puede llegarse a un estado sin que se cumpla la condición de reloj

- E<> EstadoX imply t>20

Se cumple si existe una secuencia de transiciones desde el estado inicial donde el hecho que el estadoX esté en un estado seguro implica que el tiempo (t) es mayor a 20, como lo indica la figura 2.15.

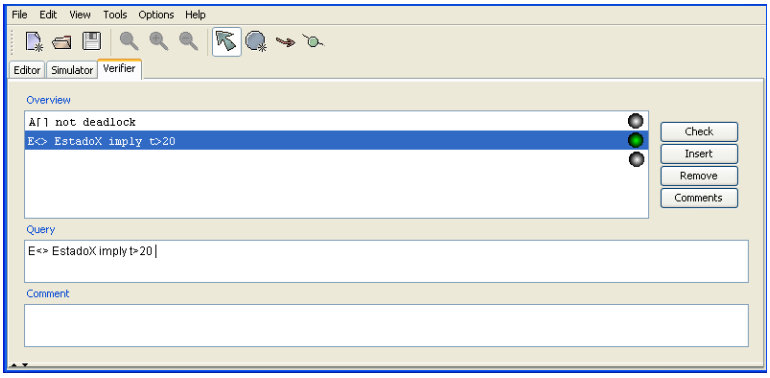


Figura 2.15. Verificación de que existe una secuencia de eventos desde el estado inicial donde el implica que el tiempo en el reloj es mayor a un valor

Simulación de un modelo.

Para realizar la simulación del autómata supervisor obtenido con la metodología y convertido mediante la herramienta FSMLADDER a código UPPAAL, se deberá seleccionar la pestaña “simulator” mostrada en la figura 1.14, esta ventana como es descrita en el ítem 1.3.3 . permite realizar una simulación de un modelo, con el fin de que el usuario pueda observar el comportamiento del autómata supervisor en el paso del tiempo.

2.8 GENERACIÓN DEL CÓDIGO LADDER MEDIANTE EL SOFTWARE FSMLADDER

Una vez se haya realizado el proceso de verificación y simulación del autómata supervisor, comprobando que no existan puntos muertos o bloqueos se procede a generar el código automático LADDER, el cual toma el archivo XML, lo interpreta y lo transforma en un archivo de extensión *.RSS que es entendido por el software RSLOGIX 1500 del PLC Allen Bradley. Para realizar el proceso consulte el anexo B. Manual de usuario Software FSMLADDER.

¿Qué es FSMLADDER?

El programa FSMLADDER, cuyo nombre se deriva de las siglas en ingles “finite state machine” (maquinas de estados finito) y la palabra LADDER (escalera), ofrece una manera eficiente y segura de establecer una conversión entre la lógica que manejan las maquinas de estado finito (autómatas finitos) y el lenguaje de programación Ladder, fue construida dentro de este proyecto de grado, utilizando el lenguaje de programación JAVA bajo el IDE ECLIPSE y utilizando la librería SWT para la interfaz grafica.

Esta herramienta permite importar los autómatas modelados en DESUMA con la sintaxis especificada por la metodología ampliada, realizar la composición paralela, traducir este resultado en el formato XML de UPPAAL, validar y verificar, y si no existen bloqueos traducirlo al código LADDER de un Micrologix 1500 de Rockwell.

Esto parece sencillo, sin embargo la herramienta desarrollada FSMLADDER cuenta con más de 2000 líneas de código que hacen todo esto posible. Se desarrollo en el IDE Eclipse utilizando Java como lenguaje de programación y está diseñada para que funcione bajo los sistemas operativos Linux o Windows. En el anexo B. se especifica con detalle los usos, alcances y limitaciones de la herramienta FSMLADDER.

Requerimientos

La herramienta FSMLADDER es una aplicación desarrollada en el entorno de trabajo SDK eclipse 3.2, por lo tanto es necesario que el ordenador tenga instalado una versión de java igual o superior a la 1.41, cumpliendo con este requisito se puede ejecutar FSMLADDER en ordenadores que usen plataformas Windows que tengan las especificaciones listadas a continuación.

- Windows 95 SE, Windows 98, Windows Me, Windows 2000, Windows XP o Windows 7.
- Procesador Pentium II 450 MHz como requerimiento mínimo.
- 50 MB de espacio en su disco duro.
- 64 MB de memoria RAM (Se recomienda 128 MB o más).
- Unidad lectora de CD (sólo para ejecutar FSMLADDER desde un CD).
- Resolución de pantalla de 800x600 o más.
- Entorno Java (JRE) versión 1.4.1_01 o más.

Nota: Cumpliendo con estos requisitos mínimos el usuario solo tiene que ejecutar la aplicación, ya sea desde una ubicación de disco duro o desde una unidad de Cd en donde se encuentre el ejecutable de la herramienta.

3 APLICACIÓN DE LA METODOLOGIA A CASOS DE ESTUDIO

En este capítulo describiremos dos casos de estudio de sistemas a eventos discretos a los que se les ha aplicado la metodología ampliada (ítem 2.5) para el uso de autómatas extendidos. Se detallará paso a paso el desarrollo de la metodología hasta obtener un supervisor vigilante capaz de reconocer los estados de funcionamiento a partir de las secuencias de eventos significativos, una vez diseñado el supervisor se hará uso de la herramienta software realizada, con el fin de realizar en primera instancia la verificación y simulación del supervisor con el fin de determinar que no existan bloqueos o puntos muertos, y poder generar el código LADDER a partir del autómata supervisor usando la herramienta software creada en este proyecto FSMLADDER.

Los casos de estudio corresponden a un caso típico de secuencias de procesos a eventos discretos y un caso práctico de la planta de sistemas a eventos discretos de la universidad del Cauca, simulando un proceso de elaboración de bebidas carbonatadas.

Los casos de estudio se describen a continuación:

- proceso de bebidas carbonatadas del laboratorio de control de procesos de la universidad del Cauca, en el cual se hace uso de temporizadores.
- Línea de producción formada por dos robots, tres máquinas y dos unidades de almacenamiento, haciendo uso de contadores y temporizadores.

El principal objetivo es comprobar que la metodología es aplicable a diferentes procesos de eventos discretos.

Inicialmente se aplicara la metodología al proceso de “elaboración de bebidas carbonatadas” ubicado en el laboratorio de control de procesos de la universidad del cauca, en el que se usan temporizadores para controlar el tiempo de mezclado de un dos mixer, además se muestra claramente la interacción entre varios componentes del proceso.

Un factor importante que se observa en este proceso consiste en que cuando un mixer asociado a un temporizador esta en ejecución, el autómata supervisor debe seguir escuchando a los sensores del proceso, porque en cualquier momento podrán ocurrir y debe realizar las acciones

correspondientes. Así por ejemplo si el mixer1 está ejecutándose durante dos minutos y en ese momento un sensor que indica que el nivel del tanque llego al nivel superior ocurre, el supervisor debe ser capaz de atenderlo, y apagar inmediatamente la válvula que lo está alimentando.

En el ejemplo dos, consistente en un brazo robótico que debe transportar piezas desde dos bandas transportadoras que contienen dos productos diferentes hasta dos sitios de almacenaje de manera alternada, en caso de que una banda se quede sin piezas atenderá únicamente a la banda que si contiene.
Este ejemplo se selecciono por dos razones, inicialmente porque hace uso de contadores y porque utiliza el concepto de recursos compartidos, consistente en un recurso que se utiliza en más de un componente del proceso a la vez, en este caso, el robot, el cual se utiliza para los procesos de transporte en las dos bandas.

A continuación se describen los dos casos en los que se aplica la metodología mostrando los resultados obtenidos.

3.1 PROCESO DE BEBIDAS CARBONATADAS DEL LABORATORIO DE CONTROL DE PROCESOS DE LA UNIVERSIDAD DEL CAUCA, EN EL CUAL SE HACE USO DE TEMPORIZADORES.

A continuación se detalla la aplicación de la metodología, para ello se hace uso de las herramientas DESUMA, UPPAAL, y FSMLADDER, en caso de el no manejo de estas herramientas recomendamos revisar antes los anexos.

3.1.1 EL PROCESO

El proceso de la fig. 3.1 consiste en un sistema a eventos discretos para la elaboración de dos tipos de bebidas carbonatadas, bebida light y bebida tradicional.

En funcionamiento normal el sistema de control regula el nivel de los cuatro tanques TK2, TK3, TK4 y TK5 de acuerdo con una consigna determinada, inicialmente se asume que los tanques 2,3,4 y 5 están vacios y que el tanque de alimentación (TK1 está lleno). Cada tanque cuenta con dos sensores que proporcionan información de cuando el líquido se encuentra en nivel inferior y superior del depósito. El nivel se controla actuando sobre las electroválvulas.

Se supone que la alimentación del depósito inferior (Tanque 1) debe estar garantizada sin embargo el supervisor debe activa una alarma en caso de que llegase a estar vacio y el proceso no podría iniciar.

El proceso se simulará solo hasta la descarga de la mezcla final en el carbonatador (TK1), donde se supondrá que la bebida se encuentra lista para la adición del dióxido de carbono.

Para la fabricación de gaseosa light y gaseosa tradicional se utilizan los siguientes ingredientes: Agua (tratada química y bacteriológicamente), saborizantes (naturales o artificiales), Acidulantes (dan el sabor levemente ácido a la bebida y actúa como preservante) y endulzantes (sacarosa para la bebida tradicional utiliza sacarosa, y edulcorantes para la bebida light).

El proceso Inicialmente se encuentra en un estado de reposo, en el cual los tanques 2,3,4 y 5 están vacios, las electroválvulas cerradas, la motobomba y los mixer apagados. Al pulsar el botón de inicio se enciende la motobomba y se energizan las válvulas V2, V3, V4 y V5 que permiten el paso de los ingredientes que se encuentran en el tanque 1 (TK1) hacia los tanques 2, 3, 4 y 5 respectivamente.

Se asume que a cada tanque llega una materia prima diferente, distribuidos así: el tanque dos contiene el azúcar para la endulzar la gaseosa tradicional, el tanque tres (TK3) contiene los edulcorantes para la endulzar la gaseosa light, el tanque cuatro (TK4) contiene los saborizantes. (ingrediente común para los dos tipos de bebidas) y el tanque cinco (TK5) contiene los acidulantes. . (ingrediente común para los dos tipos de bebidas).

Existen un sensor de nivel superior para cada uno de los cuatro tanques, TK2NS, TK3NS, TK4NS, y TK5NS que indican la cantidad del ingrediente a depositar en cada tanque, cuando este sensor se active en cada tanque se debe impedir el paso de más materia prima a ese tanque, des energizando las válvulas correspondiente.
Una vez los ingredientes hayan alcanzado el nivel superior en los cuatro tanques y las válvulas que permiten el paso de la materia prima hacia ellos se encuentren apagadas, se debe energizar la valvula1, que permite el paso de los ingredientes comunes (saborizantes y acidulantes) para las dos bebidas, hacia los tanques tres y cuatro.

Cuando los saborizantes y acidulantes provenientes de los tanques cuatro y cinco hayan sido depositados por completo en los tanques dos y tres (esto se garantiza cuando los sensores de nivel inferior TK4NI y TK5NI de los tanques cuatro y cinco se activen), se des energiza la válvula v1 y se activan los mescladores o mixer que agitan la mezcla hasta obtener una bebida homogénea.

Al finalizar el tiempo de mezclado se vacía la mezcla que se encuentra en los tanques TK2 y TK3 al tanque uno (simulando que el tanque 1 son dos tanques diferentes uno para la bebida light y otro para la bebida tradicional donde se hace el proceso de carbonatado). Activando las válvulas 7 y 6 respectivamente.
Cuando las dos bebidas hayan sido depositadas por completo en el tanque uno (esto se garantiza cuando los sensores de nivel inferior TK2NI y TK3NI de los tanques dos y tres respectivamente se activen), se des energizan las válvulas 5 y 6 que permitían el paso de las bebidas de los tanques dos y tres al tanque uno, y se inicia nuevamente la etapa de reposo.

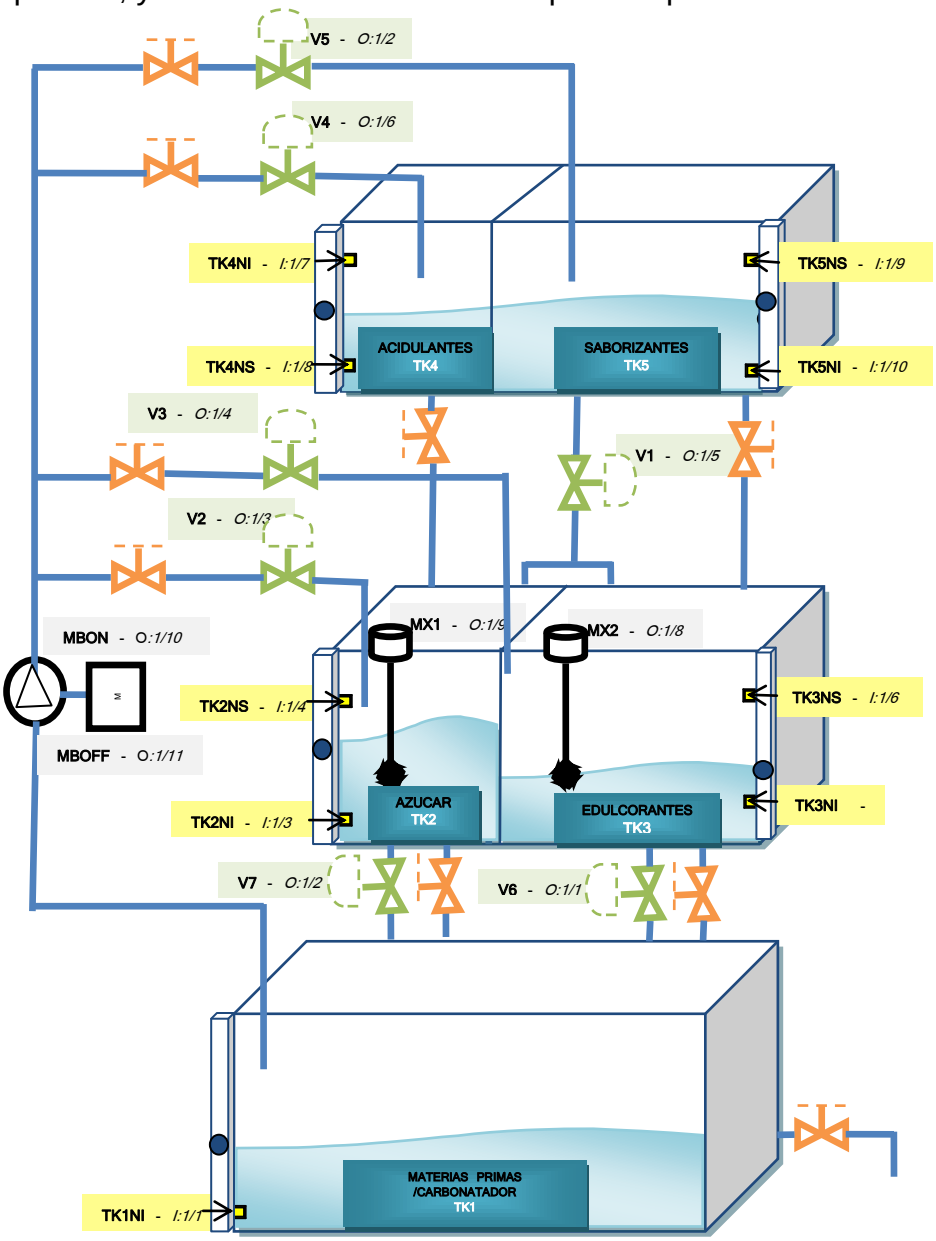


Figura 3.1Proceso bebidas carbonatadas [28].

3.1.2 LOS EVENTOS SIGNIFICATIVOS

En el proceso de tanques interactuantes, se identifican los siguientes eventos significativos:

Para el sensor de nivel de cada tanque se identifican dos eventos significativos: TKNS cuando el sensor indica que se superó el nivel superior dentro del tanque y TKNI cuando el sensor indica que ingrediente del tanque se encuentra en el nivel inferior. Cada tanque contiene este par de sensores, por lo que se clasificaron los sensores como TK**2**NS, TK**2**NI para el tanque 2, TK**2**NS, TK**2**NI para el tanque 3, TK**2**NS, TK**2**NI para el tanque 4,y para el tanque 1 solo se considera el sensor de nivel inferior puesto que se asume que la alimentación de este tanque esta siempre garantizada.

Para cada electroválvula se identifican dos eventos: V_1 abrir la válvula y V_0 cerrar la válvula.

Para cada mixer se identifican dos eventos: MX_1 para encender el mixer y MX_0 para apagar el mixer.

Para la motobomba se identifican cuatro eventos significativos: MBON_0, MBON_1, MBOFF_0 y MBOFF_1, para que la motobomba este encendida debe estar MBOFF desenergizado y MBON energizado, y para que la motobomba este apagada debe estar MBON desenergizado y MBOFF energizado.

En definitiva, se disponen de 9 eventos significativos del proceso

∑pro = {TK1NI_1, TK1NI_0, TK2NS_1, TK2NS_0, TK2NI_1, TK2NI_0, TK3NS_1, TK3NS_0, TK3NI_1, TK13NI_0, TK4NS_1, TK4NS_0, TK4NI_1, TK4NI_0, TK5NS_1, TK5NS_0, TK5NI_1, TK5NI_0, V1_0, V1_1, V2_0, V2_1, V3_0, V3_1, V4_0, V4_1, V5_0, V5_1, V6_0, V6_1, V7_0, V7_1, MX1_0, MX1_0, MX2_0, MX2_0, MBON_0, MBON_1, MBOFF_0, MBOFF_1}

Los eventos obtenidos a partir de las señales gobernadas por el sistema de control son controlables $\sum_C$= { V1_0, V1_1, V2_0, V2_1, V3_0, V3_1, V4_0, V4_1, V5_0, V5_1, V6_0, V6_1, V7_0, V7_1, MX1_0, MX1_0, MX2_0, MX2_0, MBON_0, MBON_1, MBOFF_0, MBOFF_1 }

los eventos obtenidos a partir de las señales asociadas a sensores del proceso son no controlables $\sum_{NC}$= { TK1NI_1, TK1NI_0, TK2NS_1, TK2NS_0, TK2NI_1, TK2NI_0, TK3NS_1, TK3NS_0, TK3NI_1, TK13NI_0, TK4NS_1, TK4NS_0, TK4NI_1, TK4NI_0, TK5NS_1, TK5NS_0, TK5NI_1, TK5NI_0 }.

Los eventos obtenidos a partir de la ocurrencia de un elemento no físico, es considerado un evento asociado a una variable. Hace parte de este evento la acción de la alarma que aunque es una salida, no es una salida física del proceso. El motivo principal por la cual se considera como una salida virtual es porque su ejecución no accionara ninguna salida del PLC, sin embargo podría accionar una salida simulada en un supervisorio.

$\sum_{NC}$= {ALAR1}

3.1.3 Los componentes

Los componentes están formados por los eventos controlables, no controlables y asociados a un bit. Para realizar los autómatas correspondientes a cada evento procedemos a tomar todos los estados posibles, descritos en el ítem 3.1.2.

A continuación se ilustran los autómatas correspondientes a los eventos controlables $\sum_{C}$, observables o no controlables $\sum_{NC}$,y asociados a un bit.. Recuerde que estos debe ser marcados como “control” en el campo “events” en DESUMA, como “observe” si se tratan de un evento controlable,o marcados ambos campos (“control” y “observe”) si se tratan de un evento controlable que no exite físicamente pero que en un sistema de supervisión se asociara a una salida, en este caso a una alarma que se activará en un supervisorio del proceso, recuerde siempre realizar estas clasificaciones, puesto que de ella depende la correcta interpretacio0n del la herramienta **FSMLADER,** la cual toma esta clasificación y las convierte en entradas, salidas o bis del PLC. que es un evento controlable y será interpretado como una salida física del PLC.

- **Autómatas correspondientes a los eventos observables (sensores)**

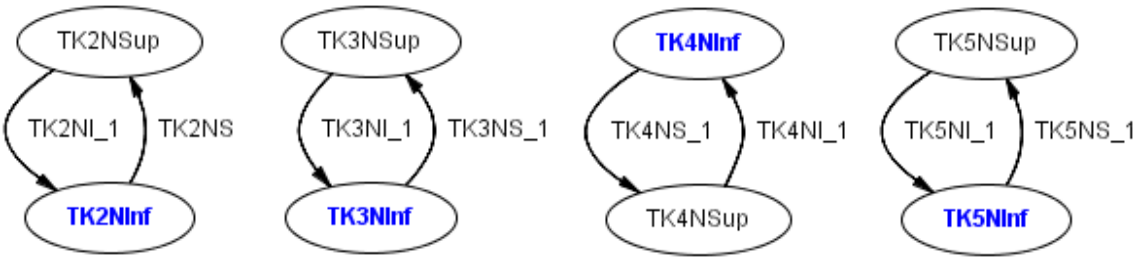


Figura 3.2. Autómatas observables correspondientes a los sensores de cada tanque.

- **Autómatas correspondientes a los eventos controlables (actuadores)**

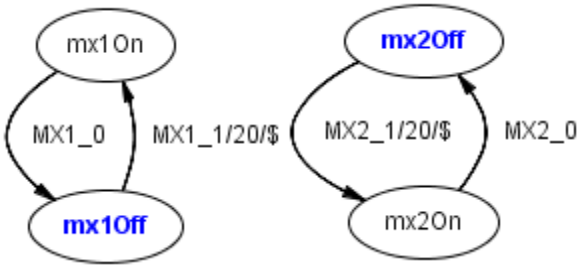


Figura 3.3. Autómatas controlables correspondientes a los mixer en los que se hace uso de temporizadores [28].

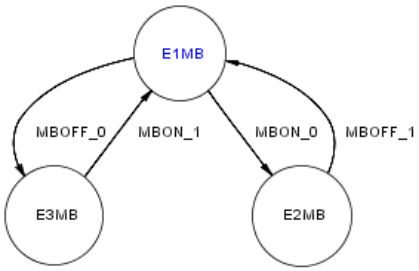


Figura 3.4. Autómata controlable correspondiente a la motobomba [28].

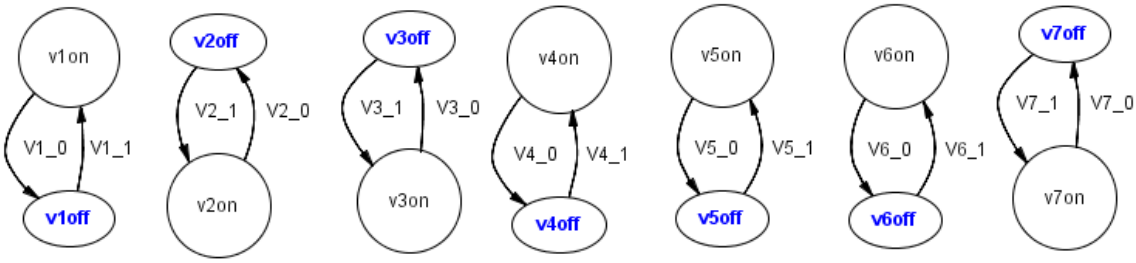


Figura 3.5. Autómata controlable correspondiente a las electroválvulas [28].

3.1.4 Modelo aumentado de la planta en lazo abierto

Una vez diseñados los autómatas correspondientes a todos los componentes del proceso en la herramienta DESUMA, se utilizo la herramienta desarrollada en este proyecto FSMLADDER para realizar la composición paralela de los autómatas diseñados mostrados en el ítem 3.1.3, el resultado

no se muestra debido a su tamaño pero se encuentra dentro de la carpeta casos de estudio de este documento. Esta composición paralela permite que los eventos se sincronicen haciendo que las acciones comunes se ejecuten simultáneamente.

3.1.5 Restricciones físicas y de control

El modelo anterior refleja cierta dinámica que no es coherente con el proceso. Con el fin de eliminar estas incoherencias, es necesario incorporar las restricciones de la fig. 3.10. Para el correcto funcionamiento del proceso y siguiendo la metodología propuesta el usuario deberá diseñar una restricción por cada sistema (por cada tanque) y una restricción general que involucre la interacción entre todos los sistemas del proceso (entre los tanques).

Para la elaboración de las restricciones se tienen en cuenta las siguientes condiciones:

Las válvulas que alimentan los tanques 2, 3, 4 y 5 solo se pueden encender si la motobomba está encendida y la motobomba solo puede ser apagada si se cumplen dos condiciones, que los cuatro tanques hayan alcanzado el nivel superior (es decir si los sensores TK2NS_1, TK3NS_1, TK4NS_1, TK5NS_1 están activas) ,y se han apagado las válvulas V2,V3,V4,V5 (es decir cuando V2_0, V2_0, V3_0, V4_0 están activas),que permiten o no, el paso de los ingredientes desde el tanque de alimentación hasta los tanques 2,3,4 y 5.

La válvula 1 que permite el paso de los tanques 4 y 5 hacia los tanques 2 y 3 solo se podrá encender una vez los todos los tanques hayan alcanzado el nivel superior y las válvulas V2,V3,V4,V5 y la motobomba estén apagados.

los mixer 1 y mixer 2 solo se podrán activar si todas las válvulas de alimentación V2,V3,V4,V5 han sido apagadas, y solo hasta que los mixer lleguen al valor preseleccionado se podrán encender las válvulas v7 y v6.

Solo cuando los tanques dos y tres hayan terminado de depositar las bebidas en el tanque uno, es decir cuando los sensores TK2NI y TK3NI estén activos se podrá iniciar el proceso nuevamente.

A continuación se describen cuatro restricciones para los tanques 2, 3 , 4 y 5 los cuales se muestran en las figuras 3.6, 3.7, 3.8 y 3.9 respectivamente.

Por ejemplo, la restricción de la fig. 3.6 corresponde al sistema del tanque dos (TK2) y afecta a siete componentes: la motobomba, las electroválvulas

V2 Y **V7**, el Mixer uno (MX1), y los sensores de nivel superior e inferior del tanque dos (TK2NI y TK2NS), además se tienen las condiciones para el encendido de la válvula uno V1, esta condición es tenida en cuenta dentro de las restricciones de cada sistema (tanque) puesto que garantiza que solo se activa si en los cuatro tanques se puede cumplir.

De igual manera se observan los componentes que interactúan en cada uno de los tanques 3,4 y 5, en las figuras 3.7, 3.8 y 3.9.

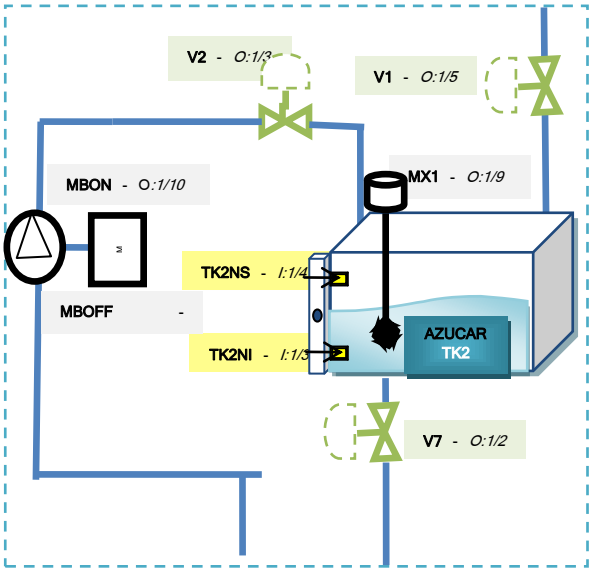


Figura 3.6. Componentes que interactúan en el sistema del tanque dos. [28].

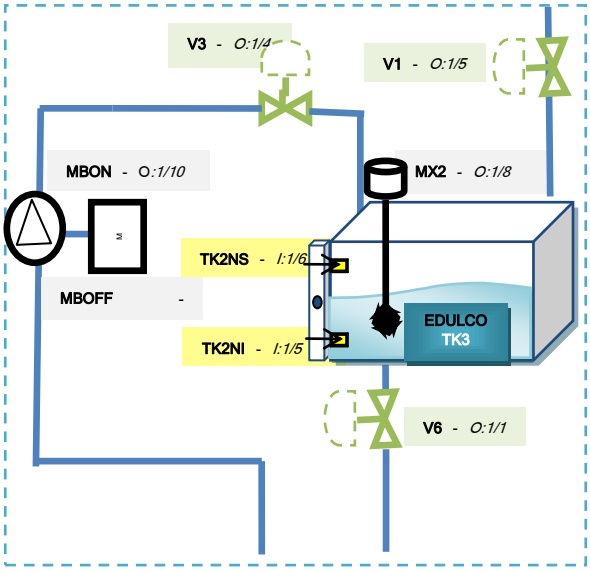


Figura 3.7. Componentes que interactúan en el sistema del tanque tres. [28].

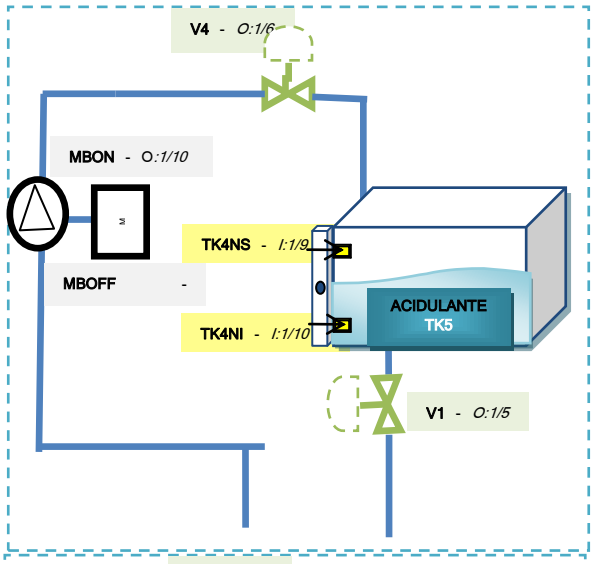


Figura 3.8. Componentes que interactúan en el sistema del tanque cinco. [28]

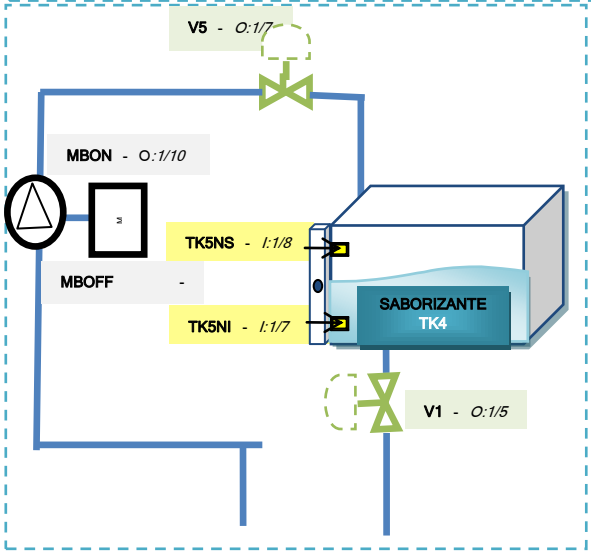


Figura 3.9. Componentes que interactúan en el sistema del tanque cuatro. [28].

Una vez observados los sensores y actuadores que interactúan en cada procedemos a realizar la restricción de cada uno, que corresponde al comportamiento individual de cada tanque, es decir, consiste en aislar el comportamiento de los demás tanques e iniciar el proceso. La composición paralela será la encargada de unir los autómatas de todos los tanques y obtener una secuencia de eventos posibles, sincronizando los eventos comunes.

En la figura 3.10 se ilustran los resultados obtenidos de los tanques 2,3,4 y 5, siguiendo este orden de izquierda a derecha.

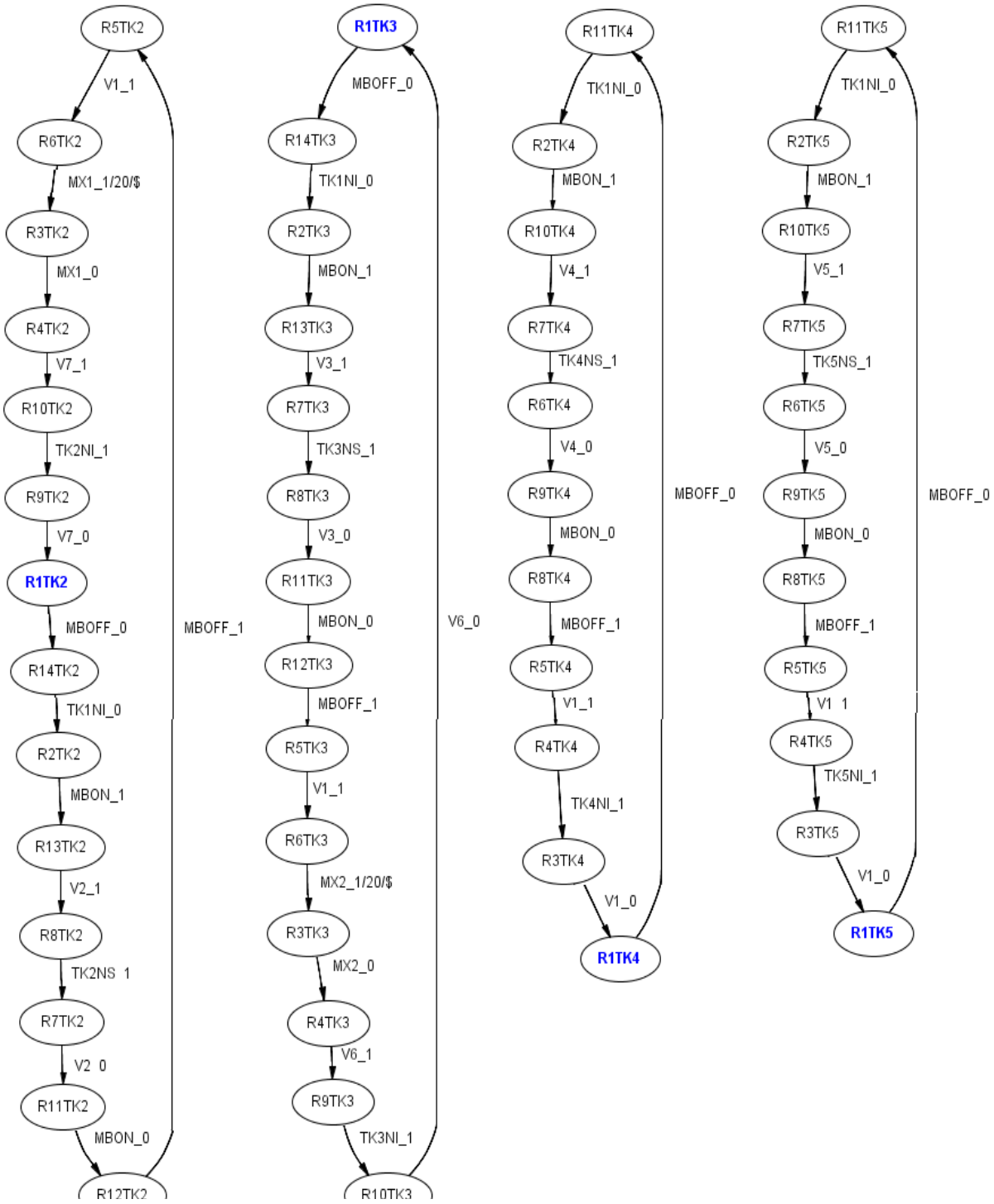


Figura 3.10. Restricciones de los componentes del proceso (tanques 2, 3, 4 y 5). [28].

Una vez se describan las restricciones de cada componente, se procede a realizar las restricciones generales que involucran los cuatro componente, es decir los cuatro tanques. En estas se hace referencia a la secuencia en que algunos eventos controlables deben ser activados, por ejemplo se especifica que las válvulas que alimentan los tanques 2, 3, 4 y 5 deben ser activadas en el orden v2, v3,v4 y v5. En caso de no colocar esta restricción la composición paralela permitirá todas las combinaciones de encendido de las cuatro válvulas, y el orden dependerá del orden en que sean almacenados los escalones dentro del código LADDER.

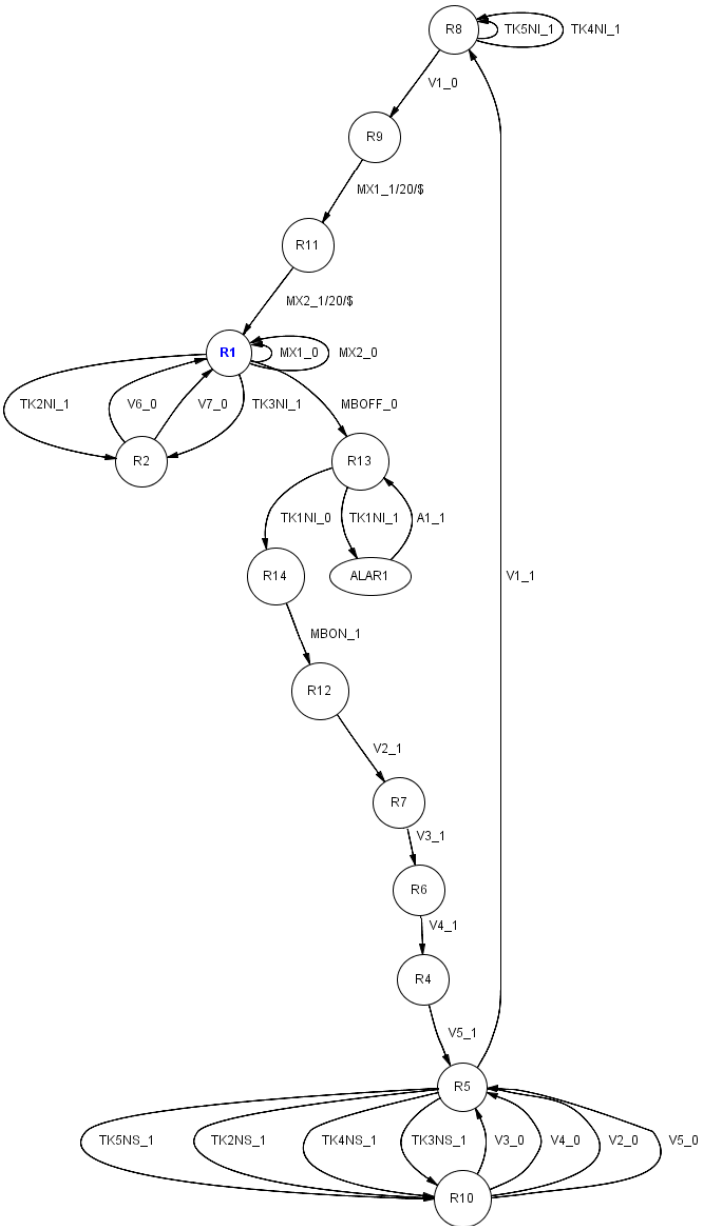
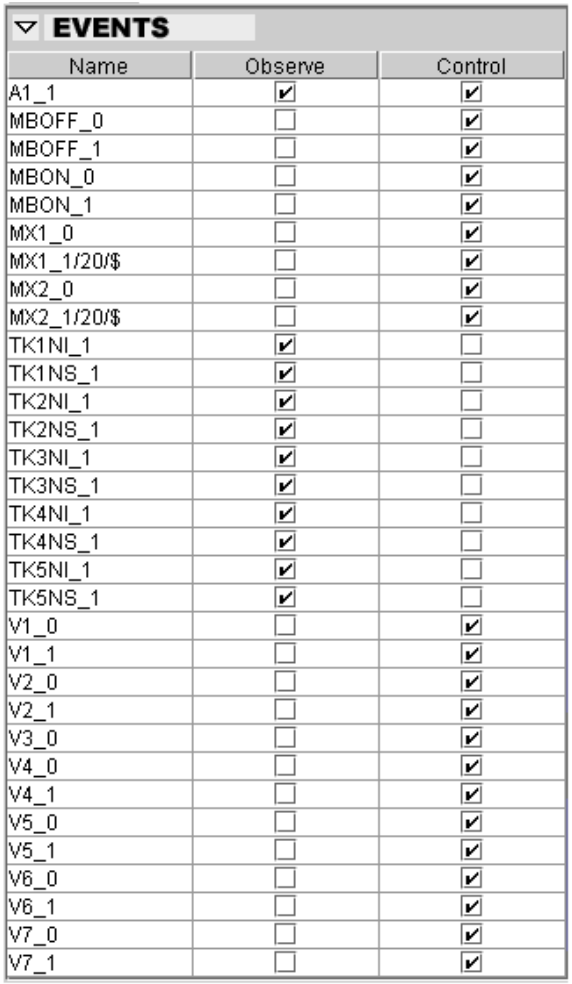


Figura 3.11. Generadores de las restricciones de control [28]

3.1.6 El modelo de la planta en lazo cerrado

La composición de G_{PLA} con las cuatro restricciones de los sistemas (tanques) y la restricción general proporciona un autómata supervisor con menos estados y transiciones, las transiciones o eventos se ven con detalle en la figura 3.12. Este generador representa el modelo de la planta en lazo cerrado, donde se observa claramente los eventos observables o sensores, los eventos controlables o actuadores y además los eventos asociados a un estado o asociados a un bit del PLC, los cuales no corresponden ni a una entrada ni a una salida física pero si pueden estar presente dentro de un supervisorio.



EVENTS

Name	Observe	Control
A1_1	☑	☑
MBOFF_0	☐	☑
MBOFF_1	☐	☑
MBON_0	☐	☑
MBON_1	☐	☑
MX1_0	☐	☑
MX1_1/20/$	☐	☑
MX2_0	☐	☑
MX2_1/20/$	☐	☑
TK1NI_1	☑	☐
TK1NS_1	☑	☐
TK2NI_1	☑	☐
TK2NS_1	☑	☐
TK3NI_1	☑	☐
TK3NS_1	☑	☐
TK4NI_1	☑	☐
TK4NS_1	☑	☐
TK5NI_1	☑	☐
TK5NS_1	☑	☐
V1_0	☐	☑
V1_1	☐	☑
V2_0	☐	☑
V2_1	☐	☑
V3_0	☐	☑
V4_0	☐	☑
V4_1	☐	☑
V5_0	☐	☑
V5_1	☐	☑
V6_0	☐	☑
V6_1	☐	☑
V7_0	☐	☑
V7_1	☐	☑

Figura 3.12. Eventos del proceso de tanques interactuantes [28].

En este caso, la imposición de las restricciones de control ha acotado la dinámica de G_{PLA} traduciéndose en una disminución del número de transiciones, como se muestra en la figura 3.13.

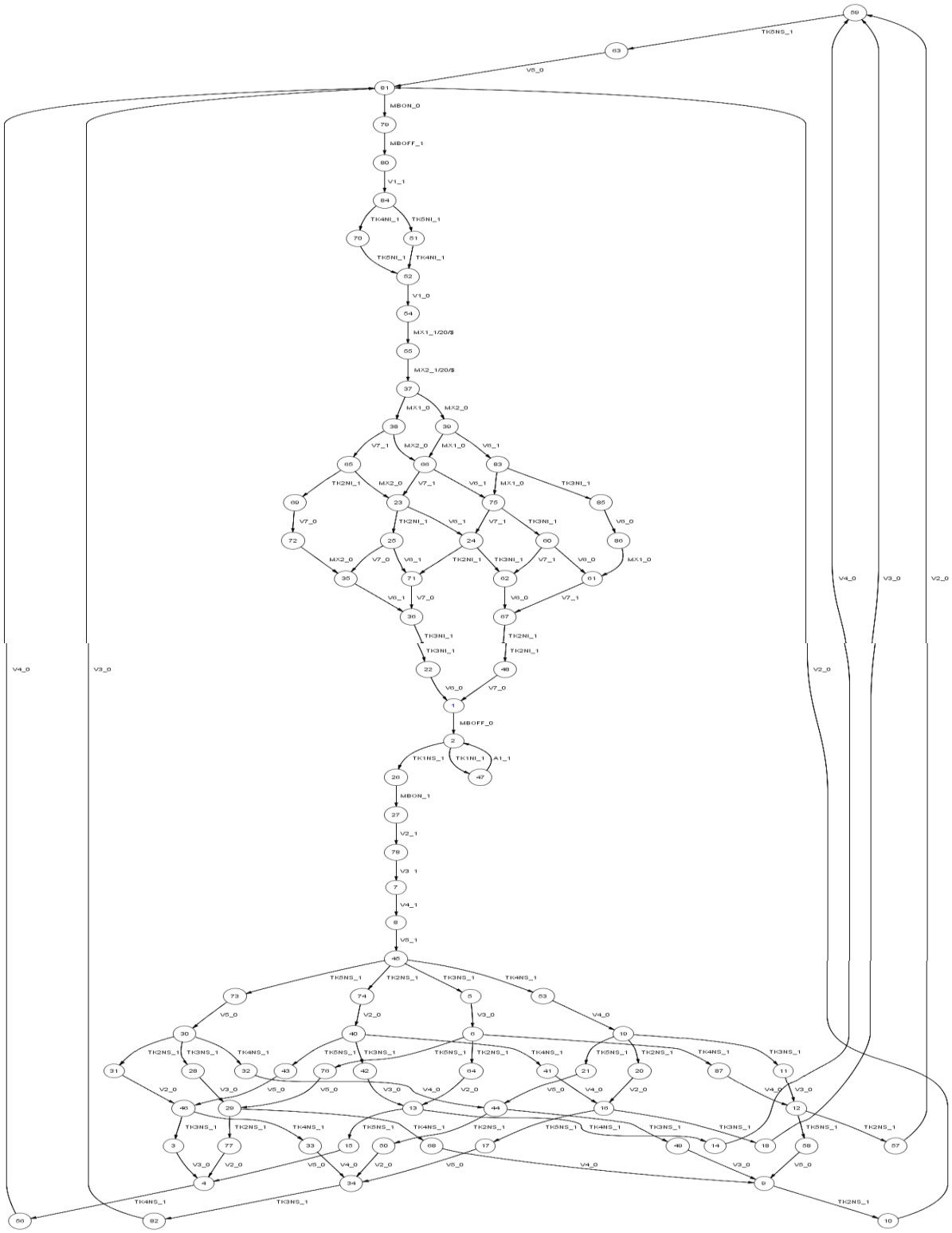


Figura 3.13. Modelo en lazo cerrado

3.1.7 Simulación y verificación usando la herramienta UPPAAL

A continuación se muestran los resultados de la simulación realizada en UPPAAL al aplicarle en el campo *query* la sentencia que determina si existen bloqueos en el autómata supervisor, sin embargo como se observa en el campo *coment* la composición paralela del sistema de tanques interactuantes no presenta bloqueos.

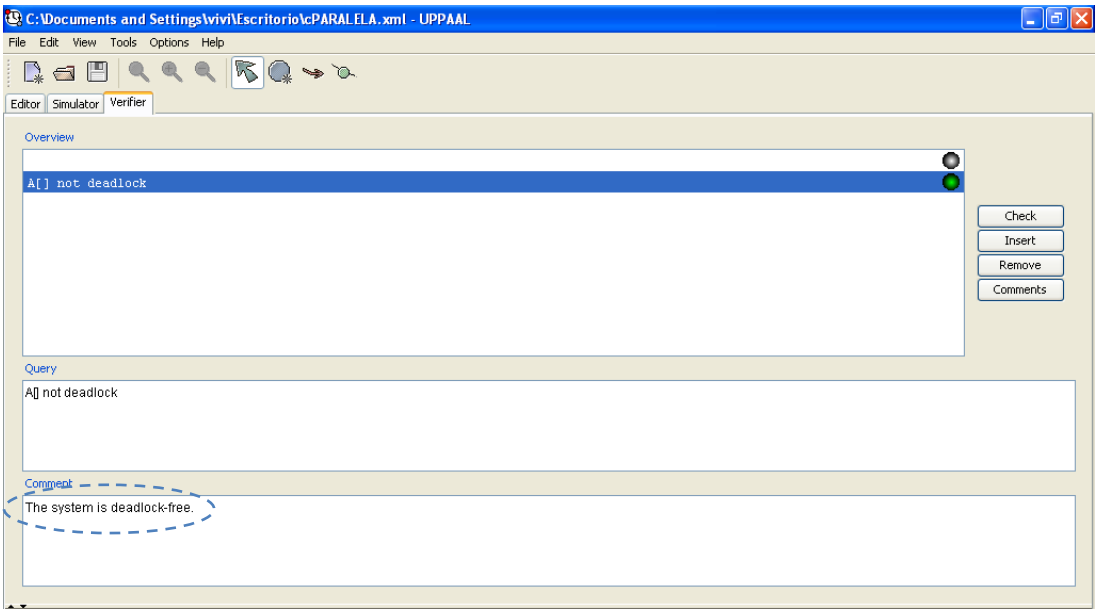


Figura 3.14. Verificación de la no existencia de bloqueos [28].

A continuación vamos a determinar si existe la posibilidad de llegar a los estados finales V6_0 y V7_0 , los cuales apagan las válvulas de vaciado de los tanques dos y tres, para ello introducimos la sentencia de “existe la posibilidad” E<> , con el fin de determinar si existe algún camino por el cual se llega al final correcto. Resultados se muestran en figura 3.15, de igual forma podemos preguntar si es posible que ocurra un evento determinado.

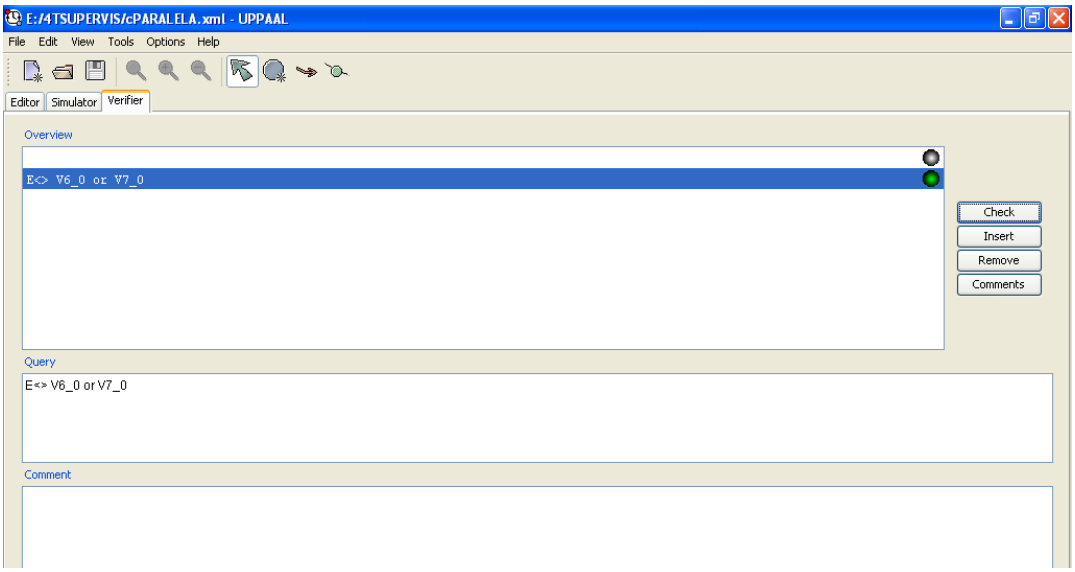


Figura 3.15. Verificación de la no existencia de estados inalcanzables [28].

3.1.8 Generación automática del código LADDER utilizando la herramienta FSMLADDER

Como se determino en el ítem anterior que el autómata supervisor no contaba con puntos muertos, podemos proceder a realizar la composición paralela, utilizando la herramienta FSMLADER, en el anexo B (Manual de usuario) se explica de forma detallada la forma de operación del software. Al cargar los autómatas, realizar la composición paralela y asignar las direcciones de las entradas, y salidas (como muestra la figura 3.16 y 3.17), obtuvimos el código LADDER adjunto en el material digital, en la carpeta casos de estudio, en la figura 3.18 se observa una muestra del resultado obtenido.

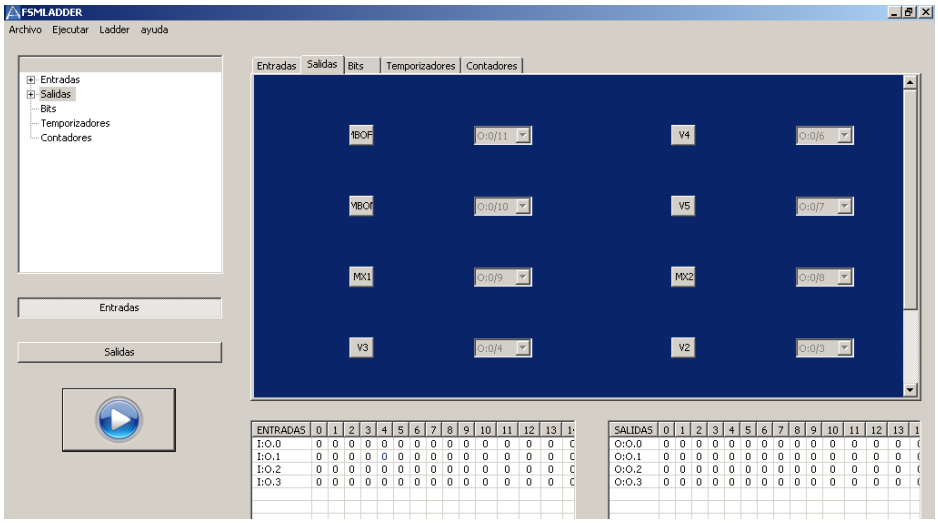


Figura 3.16. Configuración de los eventos controlables en la herramienta FSMLADDER [28].

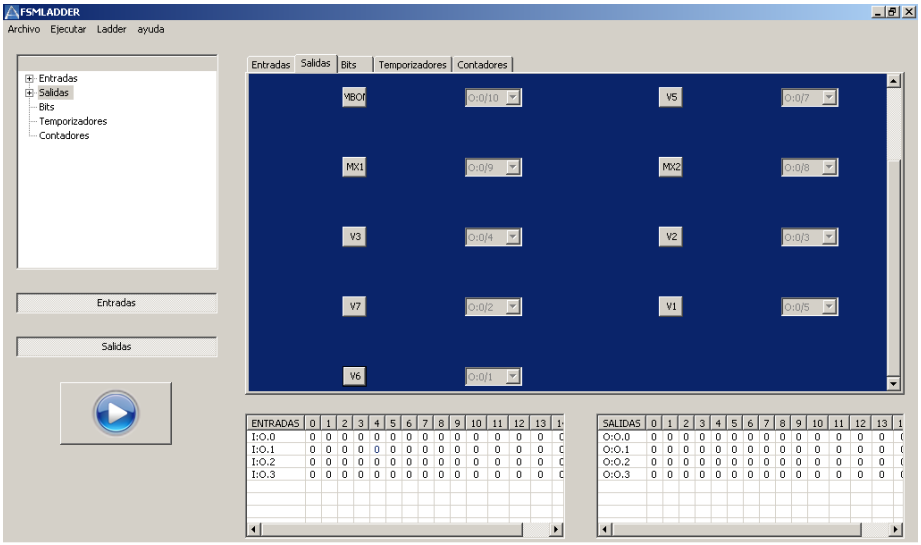


Figura 3.17. Configuración de los eventos observables en la herramienta FSMLADDER [28].

Una vez se establezcan las direcciones de los eventos controlables y observables correspondientes a las salidas y las entradas respectivamente del PLC, se nos habilitan los capos entradas y salidas como muestra la figura 3.17 donde los campos entradas y salidas se colocan en un tono más claro, indicando que todas las direcciones han sido asignadas.

A continuación es posible darle clic en el botón de ejecutar  de la herramienta FSMLADDER, lo cual permite obtener el código LADDER correspondiente a ese autómata supervisor.

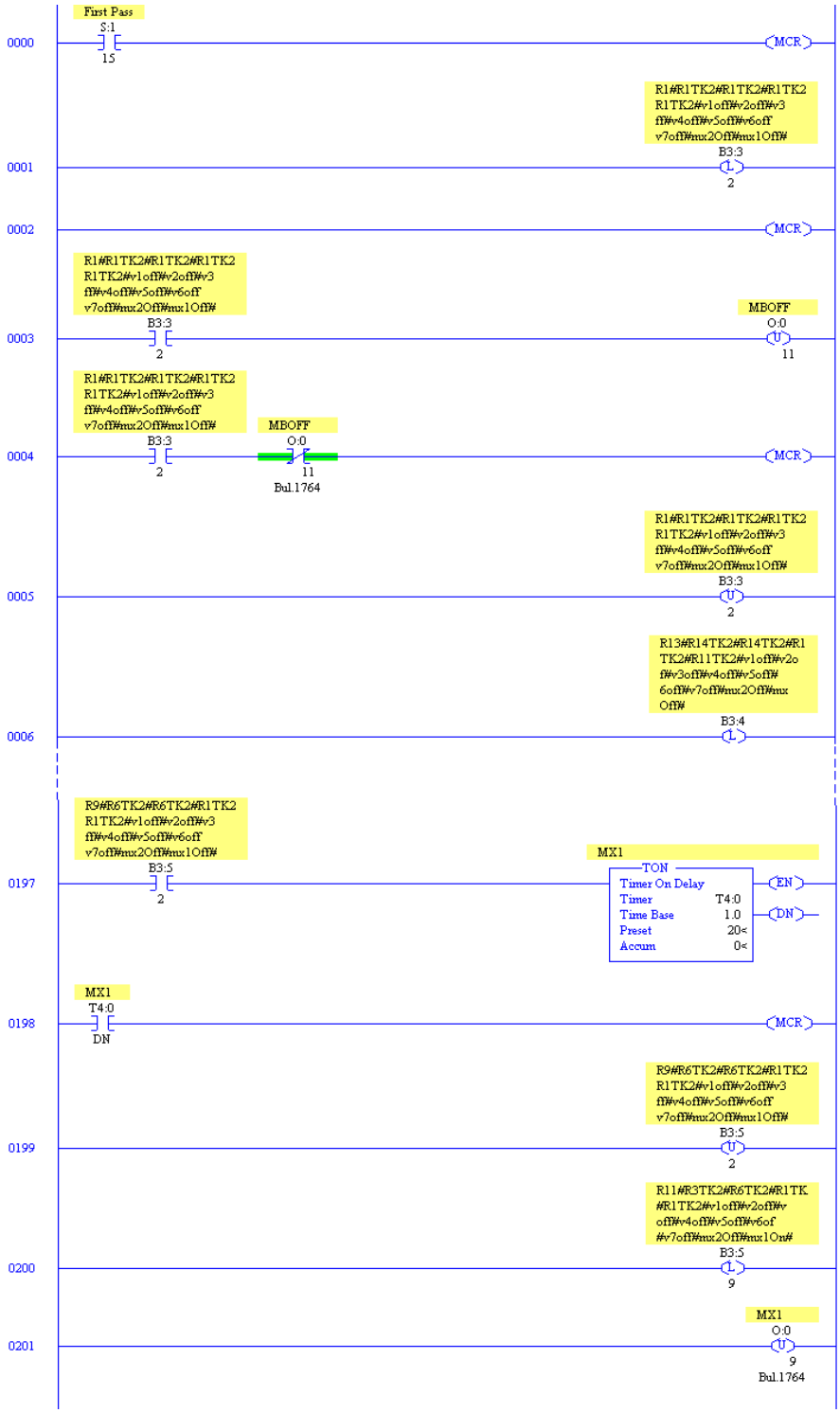


Figura 3.18 Fracción de código LADDER del proceso de bebidas carbonatadas [28].

3.1.9 Supervisión del proceso.

Se ha utilizado la herramienta Rs-view de Rockwell para la creación, edición y ejecución de un supervisorio que permita simular el comportamiento de la planta. Sin embargo gracias a la disposición de la planta del laboratorio de control de procesos se ha podido realizar también el ejercicio de manera práctica, a continuación se muestran los resultados obtenidos en el supervisorio.

Inicialmente, el tanque uno está lleno y los demás tanques(TK2, TK3 TK4, TK5) están vacios como se muestra en la figura 3.16(a), donde se ve claramente que los sensores de nivel inferior (TK2NI_0, TK3NI_0, TK4NI_0, TK5NI_0) de cada tanque están en azul claro lo que indica que están encendidos.; una vez se le da INICIO al proceso inicia encendiendo la motobomba y energiza las válvulas V2, V3, V4 Y V5 que alimentan a los tanques 2, 3, 4, y 5 respectivamente., el orden de encendido de las válvulas se muestra en la figura 3.19(b) que representa las restricciones generales del proceso, una vez se inicia el llenado de los tanques existe la posibilidad de que cualquiera de ellos llegue al nivel superior, y gracias a la composición paralela el supervisor va a estar atento a cualquier sensor que se active y a toda la secuencia de activaciones. Aunque en el proceso es claro el orden de llenado de los tanques (inicialmente se llena el tanque 3, seguidos por los tanques 2, 4 y 5 en el mismo orden) no tendría sentido establecer un orden de llenado, pues pueden existir disturbios que impidan que se siga ese orden, y el proceso se trastornaría, además por ser dado que los sensores corresponden a variables observables no podemos predecir su comportamiento. El orden de encendido de los sensores como se suponía fue el siguiente: inicialmente llego al nivel superior el tanque 3 como se muestra en la figura 3.19(c), seguidos por los sensores de los tanques 2, 4 y 5. Como se muestra en las figuras 3.19(d) , 3.19(e) , 3.19(f) respectivamente. A medida que cada tanque iba alcanzando el nivel superior, la válvula que permitía su alimentación iba siendo cerrada. Una vez los cuatro tanques alcanzaron el nivel superior y por ende las válvulas estaban cerradas, la motobomba se apago como se muestra en la figura 3.19(g) y se encendió la válvula 1, como muestra la figura 3.19(h), una vez los sensores de los tanques 4 y 5 llegaron al nivel inferior, esto indica que se han vaciado por completo y se inicia el proceso de mesclado por veinte segundos, encendiendo el mixer uno y dos como muestra la figura 3.19(h), completado el tiempo de mezclado se observa cómo se encienden las válvulas 6 y 7 (V6 y V7) que permiten llevar las bebidas de los tanques 2 y 3 de nuevo al tanque 1. El proceso tarda 9 minutos en entregar las bebidas listas para el proceso de carbonatado.

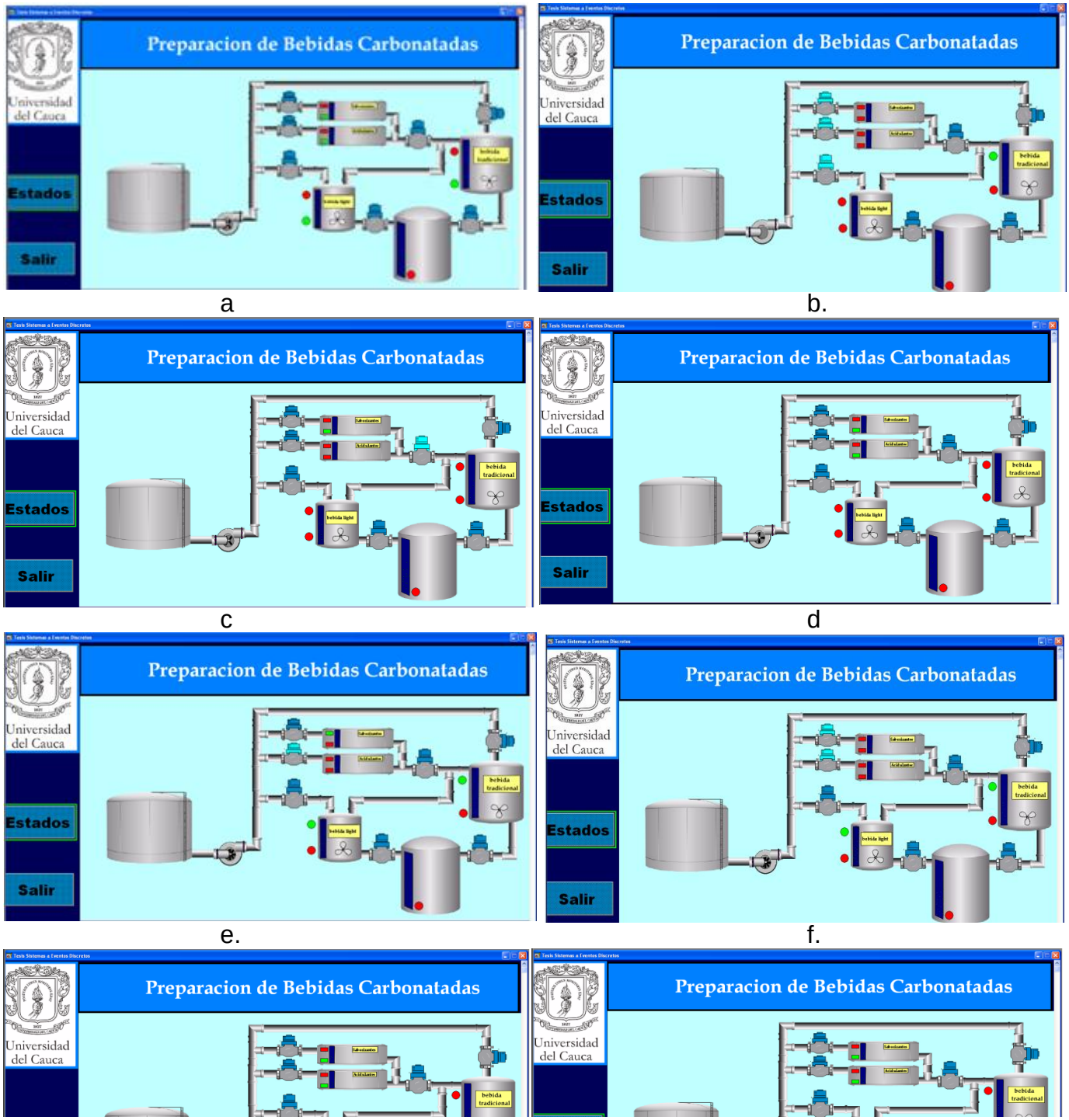


Figura 3.19. Supervisorio SCADA para el proceso de bebidas carbonatadas [28].

Figura 3.19. SCADA del proceso de bebidas carbonatadas.

3.2 AUTOMATA SUPERVISOR PARA UNA LÍNEA DE PRODUCCIÓN FORMADA POR UN ROBOT, DOS BANDAS TRANSPORTADORAS Y DOS UNIDADES DE ALMACENAMIENTO.

En cierto proceso robotizado simple, dos flujos diferentes de producción compiten por un único recurso, un brazo manipulador, que posiciona piezas tipo 1 o tipo 2 (T1 y T2). Las piezas de los flujos pueden llegar en cualquier momento y esperan por el robot disponible en colas, una de dos posiciones para piezas T1 y otra de dos posiciones para piezas T2. La política que emplea el robot para elegir una pieza desde las dos colas es la siguiente:

El robot atenderá una pieza de T1 y T2 de forma alternada (primero de una cola y luego de la otra), mientras haya piezas en ambas colas. Si solo hay piezas en una cola, atenderá esa cola. Si no hay piezas, el robot queda libre. Solo si el robot esta libre se atenderán nuevas piezas. Solo se admite el ingreso de nuevas piezas a una cola, si hay lugar en la cola. Suponga que hay sensores que detectan llegada de piezas a la cola 1 y cola 2.

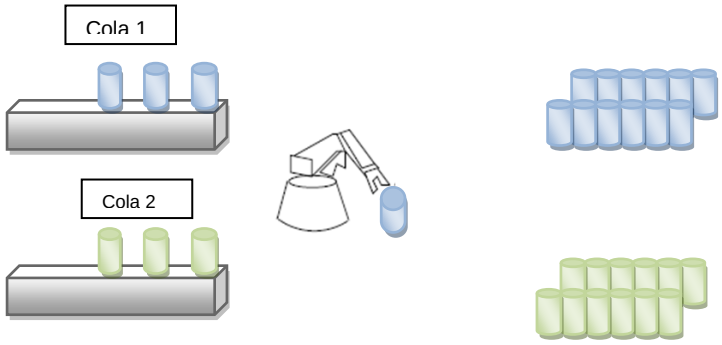


Figura 3.20 Clasificador de Piezas [28].

3.2.1 LOS EVENTOS SIGNIFICATIVOS

En el proceso se identifican los siguientes eventos significativos:

Por cada cola se identifica un estado significativo, PIEZACOLAT1 y PIEZACOLAT2 que corresponden a los sensores que se activan cuando llega una nueva pieza a la COLA T1 o COLA T2 respectivamente.

Para el brazo se identifican cuatro estados significativos, correspondientes a los eventos TOMART1_1, SOLTART1_1, TOMART1_1 y SOLTART1_1, estos corresponden a eventos significativos controlables correspondientes a actuadores tales como motores.

En definitiva, se disponen de 6 eventos significativos del proceso ∑**pro = { PIEZACOLAT1_1 , PIEZACOLAT2_1 ,TOMART1_1, SOLTART1_1, TOMART1_1 y SOLTART1_1 }**

Los eventos obtenidos a partir de las señales gobernadas por el sistema de control son controlables $\sum_C$**= {TOMART1_1, SOLTART1_1, TOMART1_1 y SOLTART1_1}**

los eventos obtenidos a partir de las señales asociadas a sensores del proceso son no controlables $\sum_{NC}$**= { PIEZACOLAT1_1 y PIEZACOLAT2_1 }.**

3.2.2 Los componentes

Los componentes están formados por los eventos controlables, no controlables y asociados a un bit.

A continuación se ilustran los autómatas correspondientes a los eventos controlables $\sum_C$, y observables o llamados también no controlables $\sum_{NC}$, Recuerde que estos deben ser marcados como "control" en el campo "events" en DESUMA, como "observe" si se tratan de un evento controlable.

- **Autómatas correspondientes a los sensores de las colas de piezas (cola 1 y cola 2)**

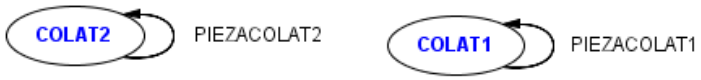


Figura 3.21. Autómatas correspondientes a los sensores [28].

- **Autómatas correspondientes al robot**

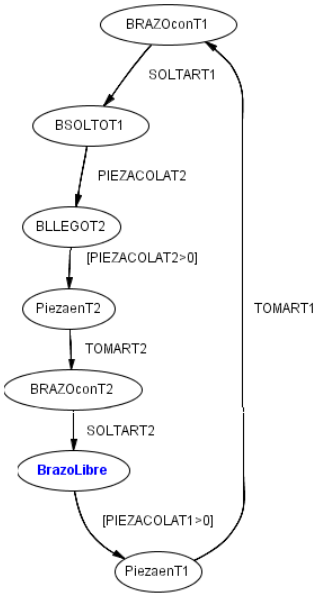


Figura 3.22. Autómata correspondiente al robot[28].

3.2.3 Modelo aumentado de la planta en lazo abierto

Una vez diseñados los autómatas correspondientes a todos los componentes del proceso en la herramienta DESUMA, se utilizo la herramienta desarrollada en este proyecto FSMLADDER para realizar la composición paralela de los autómatas diseñados mostrados en el ítem 3.3.5, el resultado no se muestra debido a su tamaño pero se encuentra dentro de la carpeta casos de estudio de este documento. La composición paralela obtenida permite que los eventos se sincronicen haciendo que las acciones comunes se ejecuten simultáneamente.

3.2.4 Restricciones físicas y de control

Para la elaboración de las restricciones se tienen en cuenta las siguientes condiciones: Los contadores de Piezas, PIEZACOLAT1/6/+ y PIEZACOLAT2/6/+ solo se podrán incrementar en uno, si antes una pieza a ingresado a la respectiva cola. Y solo podrá decrementar en uno (PIEZACOLAT1/6/- y PIEZACOLAT2/6/-) si ha ocurrido el evento TOMART1 (toma una pieza de la cola T1 para llevarla a su destino) para PIEZACOLAT1/6, o si ha ocurrido el evento TOMART2 (toma una pieza de la cola T2 para llevarla a su destino) para PIEZACOLAT2/6/-.

Solo podrán ocurrir los eventos TOMART1y TOMART2 si se garantiza que existan piezas disponibles en la respectiva cola. Gracias a la ampliación de la metodología podemos usar condiciones de comparador, en este caso para garantizar que hayan piezas disponibles en T1 y en T2 usamos la siguente sentencia: [PIEZACOLAT1>0] para las piezas de la cola 1, y [PIEZACOLAT2>0] para las piezas de la cola dos.

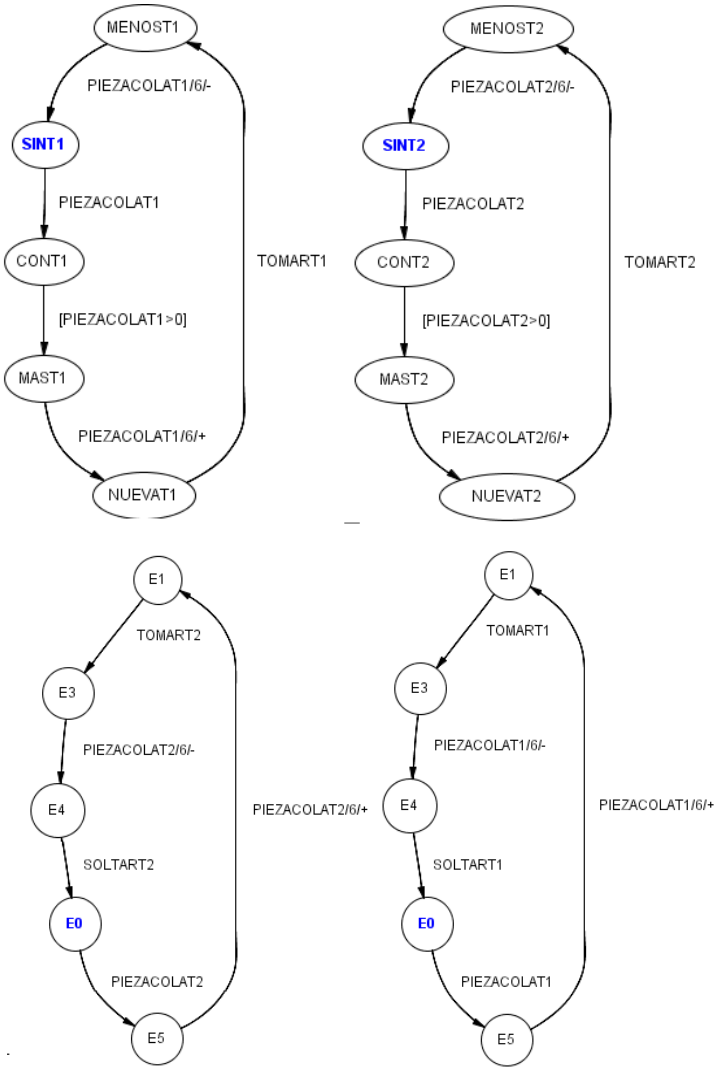


Figura 3.23. Restricciones de los componentes del proceso [28].

3.2.5 El modelo de la planta en lazo cerrado

La composición de G_{PLA} con las tres restricciones mostradas en la figura 3.23 proporciona un autómata supervisor con menos estados y transiciones que el

obtenido en el modelo aumentado de lazo abierto, las transiciones o eventos se ven con detalle en la figura 3.24. Este generador representa el modelo de la planta en lazo cerrado, donde se observa claramente los eventos observables o sensores, y los eventos controlables o actuadores.

EVENTS

Name	Observe	Control
PIEZACOLAT1	☑	☐
PIEZACOLAT1/6/+	☑	☐
PIEZACOLAT1/6/-	☑	☐
PIEZACOLAT2	☑	☐
PIEZACOLAT2/6/+	☑	☐
PIEZACOLAT2/6/-	☑	☐
SOLTART1	☐	☑
SOLTART2	☐	☑
TOMART1	☐	☑
TOMART2	☐	☑
[PIEZACOLAT1>0]	☑	☐
[PIEZACOLAT2>0]	☑	☐

Figura 3.24 Eventos del proceso de clasificación de piezas [28].

El autómata supervisor obtenido se muestra en la figura 3.25.

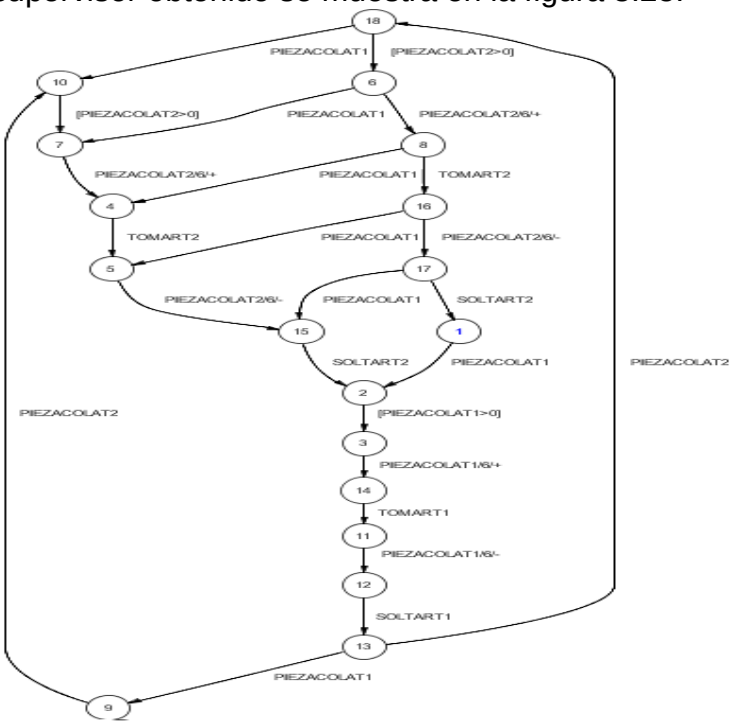


Figura 3.25 Modelo en Lazo cerrado caso de estudio Clasificador de piezas.

3.2.6 Simulación y verificación usando la herramienta UPPAAL

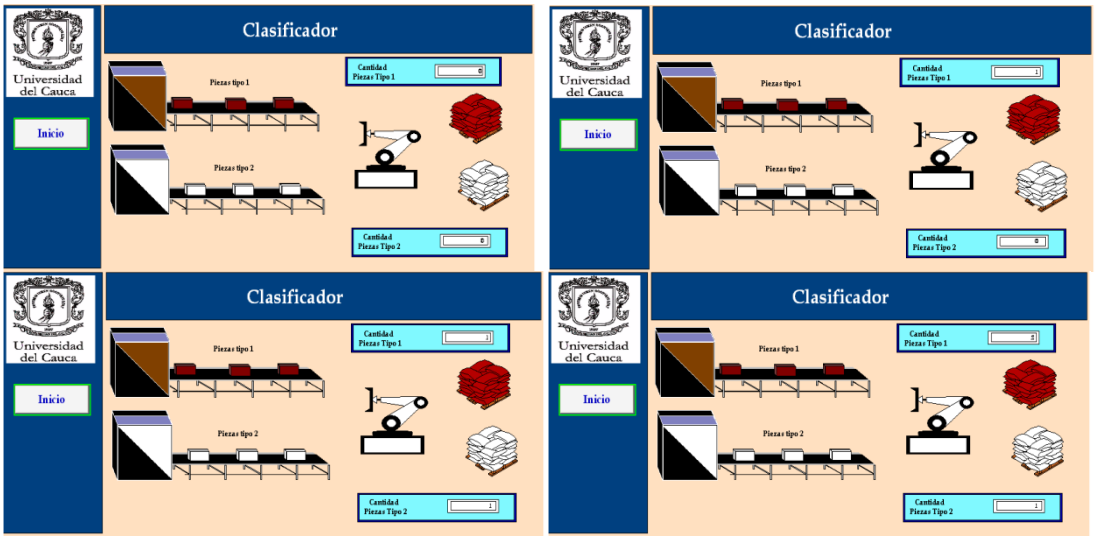
Se realizaron las pruebas de verificación y el Modelo en lazo cerrado no presenta bloqueos ni estados inalcanzables. Además se verificaron las condiciones de los contadores.

3.2.7 Generación automática del código LADDER utilizando la herramienta FSMLADDER

Como se determino en el ítem anterior que el autómata supervisor no contaba con puntos muertos, podemos proceder a realizar la composición paralela, utilizando la herramienta FSMLADER, en el anexo B (Manual de usuario) se explica de forma detallada la forma de operación del software. Al cargar los autómatas, realizar la composición paralela y asignar las direcciones de las entradas, al igual que se hizo con el ejemplo práctico de las bebidas carbonatadas. La composición paralela se muestra con detalle en la carpeta CASOS DE ESTUDIO -> clasificador.

3.2.8 Supervisión del proceso.

Inicialmente no hay piezas en ninguna de las bandas transportadoras, observese el contador en ceros de piezas, una vez llegue alguna pieza Tipo uno inicia el proceso, en el cual se toma la pieza uno y se la lleva al almacén de las piezasT1, posteriormente se transporta la pieza T2, si no llegase a haber piezaT2 el Robot toma nuevamente una pieza tipo uno.



4 CONCLUSIONES, TRABAJOS FUTUROS Y LIMITACIONES.

4.1 CONCLUSIONES

La metodología utilizada en este proyecto se basa en el modelado individual de varios aspectos comportamentales del proceso y en su combinación mediante la composición paralela de autómatas a fin de obtener un modelo a eventos discretos de un proceso industrial, aunque al contrario de lo que pudiese ocurrir con otros métodos, esta metodología no pretende minimizar el número de variables de estado, por lo que podría no parecer óptimo a la hora de realizar la conversión al lenguaje LADDER, desde el punto de vista de minimizar el hardware.

Sin embargo, el costo y volumen de un sistema dependen cada vez menos del número de variables empleadas, sobre todo si se emplean PLC y, en cambio, adquieren cada vez más importancia otros aspectos como el propio costo de diseño, el tiempo de desarrollo de software, la fiabilidad y la facilidad de prueba y mantenimiento.

Existen diagramas funcionales como el GRAFCET (nombre que también se da a las redes de Petri ordinarias con marcación ON OFF) que ayudan en la programación de PLC y se encargan de describir la evolución del proceso que se quiere automatizar, haciendo uso de unos elementos gráficos y unas reglas de evolución para reflejar la dinámica del comportamiento del sistema; sin embargo, ninguna de estas herramientas propone una metodología formal para el diseño del programa y por lo tanto no se proponen métodos sistemáticos e integrales de diseño, donde se contemplen aspectos tales como la formalización, la verificación y validación del programa diseñado, dando como resultado algoritmos erróneos, soluciones que dependen de estilos de programación y dificultades para hacer modificaciones y mantenimiento de los programas.

Con el desarrollo de este proyecto de grado se aporta a la solución de este problema, gracias a que la metodología usada permite no solo diseñar el autómata supervisor, sino además realizar la verificación y simulación del mismo, a fin de comprobar que no existan bloqueos o estados inalcanzables, posterior a lo cual el autómata estará listo para ser convertido a LADDER de manera automática con la herramienta FSMLADDER, lo que garantiza una solución sin errores, contempla todas los posibles caminos dentro de un proceso y aumenta la confiabilidad del código obtenido.

En este trabajo definimos una metodología formal para el diseño de un autómata supervisor de un proceso de eventos discretos, la cual permite el uso de autómatas extendidos a fin de que el usuario pueda seguir una serie de pasos y obtener un autómata capaz de supervisar sin error un proceso de eventos discretos. Se hizo un especial énfasis en la especificación y el análisis de un caso de estudio, considerado como *"proceso de elaboración de bebidas carbonatadas*" del laboratorio de control de procesos de la universidad del Cauca, como caso práctico de aplicación de la metodología; en el que se demuestra el correcto funcionamiento del autómata supervisor obtenido, comprobado mediante pruebas de alcanzabilidad y accesibilidad.

Se aplicó además la metodología en dos casos de estudio teóricos demostrando que la metodología aplica para diferentes tipos de procesos de eventos discretos, en los cuales el uso de temporizadores y actuadores juega un papel importante dentro del sistema.

Antes de la realización de este proyecto de grado el usuario podía utilizar la herramienta académica DESUMA solo para diseñar supervisores de procesos en los que no estuviera involucrado el uso de temporizadores y contadores, sin embargo es muy común que los procesos utilicen este tipo de mecanismos, bien sea para esperar determinado tiempo la finalización de una operación o la liberación de un recurso, operar una maquina durante un tiempo determinado, o contar el número de piezas de un producto; ahora el usuario siguiendo la sintaxis definida en el capitulo cuatro podrá diseñar autómatas extendidos (autómatas con temporizadores, contadores y variables) para los contadores, temporizadores y las variables.

Herramientas como son DESUMA y UPPAAL son muy utilizados en campos académicos, mas no en el campo industrial, sin embargo LADDER es una herramienta utilizada tanto en la industria como en la academia, resultando interesante a partir de una herramienta académica poder programar una herramienta industrial, con un alto grado de confiabilidad y disminución de errores.

La Herramienta software FSMLADDER está diseñada para trabajar únicamente con la familia SLC500, específicamente con el procesador SLC 5/03, el cual proporciona hasta 960 puntos de E/S, soporta 12K en palabras y 4 K en almacenamiento de datos adicional, programación en línea y un interruptor de llave para seleccionar 1 de 3 métodos de operación (marcha, programación y remoto), e incluye un canal RS-232. Se eligió este procesador debido a que es uno de los más usados en la familia de procesadores SLC500 debido a que presenta alta flexibilidad a un "precio justo" para la aplicación, por lo

que actualmente cuenta con una gran popularidad en nuestro medio, el hecho de permitir que la herramienta FSM LADDER utilice otro tipo de procesador se consideraría en un trabajo futuro.

Finalmente diseñamos una herramienta software llamada FSMLADDER que integre la elaboración de la composición paralela de los autómatas finitos diseñados con la metodología propuesta, la conversión de los autómatas del formato DESUMA al formato UPPAAL para la simulación y verificación con autómatas extendidos, y la generación automática del código LADDER.

Una contribución al estudio de construcción de supervisores utilizando las herramientas DESUMA y UPPAAL es el haber automatizado la traducción del código generado en DESUMA, donde se hace la composición paralela y verificación de no existencia de bloqueos, a código que puede leerse desde la herramienta UPPAAL donde pueden simularse y validarse autómatas extendidos, operación que resulta dispendiosa de hacerse manualmente porque la composición paralela generalmente está compuesta por un número significativo de estados y eventos.

En definitiva la ampliación de la metodología formal al uso de autómatas extendidos, la conversión de este supervisor al formato XML para simulación y verificación en la herramienta UPPAL y la generación automática del código LADDER, constituyen los aportes de este trabajo y un punto de partida para la realización de nuevos trabajos de investigación en torno a los sistemas de eventos discretos.

Con La aplicación de la metodología como método formal y la utilización del software FSMLADDER en los casos típicos de estudio, se obtuvieron las siguientes ventajas.

- Las soluciones dependen menos de los estilos personales de programación, puesto que la metodología establece la forma en que se deben crear los autómatas finitos y las restricciones, y es a partir de ellos que se obtiene el resultado final.
- Se redujo la posibilidad de errores en el código LADDER, puesto que la verificación y la simulación garantizan que no van a existir estados no alcanzables o bloqueos, además de que gracias a que la metodología se basa en el modelado individual de varios aspectos comportamentales del proceso al ser combinados mediante la composición paralela de autómatas se obtienen todas las

posibilidades de comportamiento, lo que garantiza que siempre se van a considerar todas las posibilidades.

- Tiempos de desarrollo más cortos, gracias a que el usuario con solo diseñar los autómatas de los componentes y de las restricciones del proceso, y validar y simular la composición paralela, ya podrá obtener el código LADDER para la programación del PLC.
- La facilidad de hacer modificaciones que reflejen cambios en el sistema de especificaciones funcionales.
- Los procesos de conversión de DESUMA a UPPAAL, y de UPPAAL a LADDER son una operación transparente para el usuario.
- No se necesita un alto dominio de la programación en escalera o LADDER, para poder realizar la programación de un PLC micrologix 1500. de manera que el usuario con solo diseñar los autómatas y realizar la composición paralela, podrá obtener de manera automática el código LADDER con instrucciones de contadores y comparadores si es necesario.

4.2 TRABAJOS FUTUROS

Algunos trabajos futuros son:

Ampliar la aplicación FSMLADDER para que el usuario pueda adicionar al PLC MÓDULOS DE entradas/Salidas DISCRETAS dentro de la interfaz, para esto el usuario deberá definir el tipo de modulo a adicionar según su clasificación de 4, 8, 16 y 32 puntos.

Dado que La Herramienta software FSMLADDER está diseñada para trabajar únicamente con el procesador SLC 5/03 de la familia SLC500, podría en primera instancia dejarse como trabajo futuro que el usuario pueda seleccionar el tipo de controlador a usar, por ejemplo el procesador 5/04 el cual proporciona las mismas características que el SLC 5/03 pero con funciones adicionales, como tiempos más rápidos de instrucciones matemáticas, posibilidad de comunicarse con los procesadores PLC-5 en la red DH+ sin necesidad de hardware adicional, otros procesadores que usa la familia SLC 500 son el procesador 5/01, 5/02 y 5/05.

La herramienta software FSMLADDER diseñada dentro de este proyecto de grado es aplicable para el PLC MICROLOGIX 1500, trabajos futuros podrían ampliar la herramienta para que sea aplicable a las familias PLC-2, PLC-3, PY PLC-5 y CONTROLLOGIX de Allen Bradley, siendo capaz de generar el archivo .RSS a partir de UPPAAL, la ventaja radica en que la metodología seguiría siendo la misma, y ya está implementada la conversión de

DESUMMA A UPPAL de tal forma que solo habría que modificar la estructura del archivo RSS.

4.3 LIMITACIONES

Debido a las limitantes de una de las aplicaciones que hacen parte de las librerías UMDES usada por DESUMA (cpar.exe), la cual permite realizar la operación de composición paralela entre autómatas, la metodología podría no ser aplicable a procesos cuyo número de eventos al realizar la composición paralela supere los siete mil, sin embargo una correcta definición de las restricciones podrían acotar el numero de eventos del autómata supervisor resultante. Se recomienda su uso en problemas académicos.

En caso de que el usuario por algún motivo desee modificar directamente el código LADDER desde la herramienta RS LOGIX, el proceso podría resultar engorroso si el autómata supervisor resultante está compuesto por un amplio número de estados y eventos, sin embargo la estructura seguida por el código LADDER puede ser seguida paso a paso en la composición paralela presentada en DESUMA o en UPPAAL utilizando el archivo XML generado por herramienta FSMLADDER.

BIBLIOGRAFÍA

[1] J Figueras. “Modelado y simulación. Aplicación a procesos logísticos de fabricación y servicios”. Universidad politécnica de Cataluña. España. 2003.

[2] J. Gil, A. rubio. “Ingeniería de Control. Control de Sistemas Continuos”, Escuela Superior de Ingenieros. Universidad de Navarra. Pag. 4. 2004.

[3] C. Cassandras, S. Lafortune. “Introduction to Discrete Event systems”. Editorial Springer. Segunda Edición. Pags. 45-46. 2008

[4] A. Urquia. “Modelado de Sistemas Discretos”. Departamento de Informática y Automática. Universidad Nacional de Educación a Distancia (UNED). Pag. 14. 2009.

[5] J. Gutierres “Máquinas de Estados Finitos - Breve Introducción”, Escuela Superior de Computo, 2008.

[6] C. Cassandras, S. Lafortune. “Introduction to Discrete Event systems”.Editorial Springer. Segunda Edición. Pag. 27. 2008

[7] C. Cassandras, S. Lafortune. “Introduction to Discrete Event systems”.Editorial Springer. Segunda Edición. Pag. 31. 2008

[8] R. Brena. “Automatas y Lenguajes. Enfoque de diseño”. Facultad de ingeniería y ciencias. Tecnológico de Monterrey. Pag. 26. México. 2003.

[9] C. Cassandras, S. Lafortune. “Introduction to Discrete Event systems”. Editorial Springer. Segunda Edición. Pag. 40. 2008.

[10] N. Chomsky. “Logical Structure of Linguistic Theory”.MIT Humanities Library.Microfilm. 1955.

[11] I. Navarrete, M. Cardenas, D. Sanches. “Teoría de Autómatas y Lenguajes Formales”. Departamento de Ingeniería de la información y las comunicaciones. Universidad de Murcia. Pag. 7. 2002.

[12] R. Brena. “Autómatas y Lenguajes. Enfoque de diseño”. Faculta de ingeniería y ciencias. Tecnológico de Monterrey. Pag. 17. México. 2003.

[13] C. Cassandras, S. Lafortune. “IntroductiontoDiscreteEventsystems”. Editorial Springer. Segunda Edición. Pag. 59. 2008

[14] J. Hopcroft, J. Ullman. “Introducción a la Teoría de Autómatas, Lenguajes y Computación”, Editorial Continental. Pag. 15. México 1993.

[15] S. Jiang, R. Kumar. “*Supervisory Control of discrete events Systems, Temporal logic specifications”.Society for Indutrial And Aplied Mathematics*.2006.

[16] E. Yourdon. “Analisis Estructurado Moderno”. Editorial Prentice-Hall Hispanoamericana S.A. Pág. 288. 1993.

[17] R. Johnsonbaugh. “Matemáticas discretas”. Editorial Iberoamericana. Primera Edición. Págs. 312-318. 2002.

[18] R. Romera, J. Pedro. “Automatización: problemas resueltos con autómatas programables”. Cuarta Edición. México. 2003.

[19] A. Urquia. “Modelado de Sistemas Discretos”. Departamento de Informática y Automática. Escuela Técnica superior de Ingeniería Informática. Pag. 95. 2009.

[20] C. Cassandras, S. Lafortune. “Introduction to Discrete Event systems”.Editorial Springer. Segunda Edición. Pag.63-64. 2008.

[21] C. Cassandras, S. Lafortune. “Introduction to Discrete Event systems”.Editorial Springer. Segunda Edición. Pag. 65. 2008.

[22] C. Cassandras, S. Lafortune. “Introduction to Discrete Event systems”.Editorial Springer. Segunda Edición. Pag. 76. 2008

[23] C. Cassandras, S. Lafortune. “Introduction to Discrete Event systems”.Editorial Springer. Segunda Edición. Pag. 77. 2008

[24] J. Hopcroft, J. Ullman. “Introducción a la Teoría de Autómatas, Lenguajes y Computación”, Editorial Continental. Pag. 15. México 1993.

[25]J. Hopcroft, J. Ullman. “Introducción a la Teoría de Autómatas, Lenguajes y Computación”, Editorial Continental. Pag. 19. México 1993.

[26] R. Sarrate, J. Aguilar “Supervisión basada en un modelo comportamental del proceso”. Laboratorio de arquitectura y análisis de sistemas. Universidad Politécnica de Cataluña. España. 2008.

[27] Software de simulación Uppal versión 4.0.7. *Department of Information Technology at Uppsala University de sue Suecia* y el *Department of ComputerScience at Aalborg University* en Dinamarca.

[28] G Zapata. “Diseño de automatismos secuenciales para controladores lógicos programables”. 2007.

[29] M. Zhou, E. Twiss. “Design of Industrial Automated Systems Via Relay LADDER Programming and Petri Nets”. En: IEEE Transactions on Systems, Man and Cybernetics, USA, IEEE, Vol 28, pag. 137. USA 1998.

[30] M Cantarelli, J. Roussel. “Reactive control system design using the Supervisory Control Theory: evaluation of possibilities and limits”. 2008.

[31] C. Parra. “Modelado y Simulación del Control Supervisorio para Sistemas Holónicos de producción Continua”. 2006.

[32] D. Luna. “Verificación de Sistemas Reactivos y de Tiempo Real en Teoría de Tipos”. 2006. *Instituto de Computación* (*InCo*)*. Universidad de la República. Uruguay.*