UNIVERSIDAD
DE LOS ANDES
MERIDA VENEZUELA

Automatización Integral de Sistemas de Producción Continuos: Control y Supervisión

Edgar Chacón
Gisela De Sarrazin
Universidad de Los Andes

Julio 1998

Prefacio

El desarrollo de Sistemas de Automatización Industrial basado en la utilización de sistemas híbridos jerarquizados, como elemento de conceptualización y diseño de la arquitectura de control; y en la utilización de una metodología orientada a objetos que permite la construcción de sistemas integrados de automatización, aparece como una solución para el manejo y comprensión de la complejidad propia del sistema a controlar. Esta complejidad viene dada por: 1) los múltiples sub-sistemas que a menudo tienen una dinámica difícil de modelar que transforman insumos en productos intermedios o finales, 2) las interacciones entre ellos que condicionan su evolución propia, 3) los aspectos financieros y de organización administrativa, que determinan la viabilidad de la producción y su planificación y finalmente, 4) las relaciones con el ambiente que en definitiva determinarán la aceptación de un producto sus costos y su valor de mercado.

Para establecer una automatización integrada, los sistemas de producción se representan mediante una mezcla de modelos continuos y discretos formando una jerarquía de modelos. En los niveles más cercanos al proceso físico la dinámica de los mismos se modela de una manera continua para representar la realidad, algunas veces mediante una familia de modelos continuos, mientras que a medida que los niveles de la jerarquía organizacional se alejan del proceso físico, la realidad del sistema se modela de una manera discreta. El sistema de automatización global resulta entonces en una jerarquía de sistemas, que deben estar integrados, trabajar de forma inteligente y cooperante, con el fin de responder rápidamente ante cambios en el proceso de producción.

En el presente texto se da una introducción general sobre las necesidades de automatización en una empresa, las tecnologías necesarias para lograr dicha automatización, para luego entrar en profundidad en el modelado de los sistemas discretos, híbridos y como los mismos pueden ser supervisados y controlados mediante una jerarquización de los mismos.

Los resultados aquí mostrados son producto del trabajo de investigación financiado por el CONICIT (Venezuela) y el CDCHT de la Universidad de Los Andes a los autores. La tercera parte trata de suministrar unas técnicas para el desarrollo de sistemas de automatización integral, que resulta de diversas experiencias en la introducción de sistemas de automatización integrada en aplicaciones industriales y, sobre todo, de las diferentes experiencias adquiridas a lo largo del desarrollo del proyecto I-22 *VASID* del programa de Nuevas Tecnologías del CONICIT financiado a Edgar Chacón. En particular, los problemas de modelado, son resultado de varios trabajos realizados por estudiantes de los postgrados de Ingeniería de control y de Computación de la ULA.

Quisieramos agradecer las sugerencias y comentarios de Juán Cardillo.

Índice General

I La Automatización Integrada de Sistemas **1**

1 Introducción a la Automatización Integrada **3**

1.1 El Modelo CIM. Una referencia 3

1.1.1 El modelo CIM 5

1.2 Automatización de Procesos Continuos 6

1.2.1 La Pirámide de Automatización 7

1.3 Arquitectura para la Automatización Integral 9

1.3.1 Elementos de Automatización 9

1.4 Ejemplo de un sistema complejo 12

1.5 El Proceso de Automatización 13

1.5.1 Sistemas Continuos por regiones de operación 13

1.5.2 Sistemas con una Dinámica Discreta 15

II Sistemas Dinámicos en Automatización **19**

2 Sistemas Discretos **21**

2.1 Introducción 21

2.2 Lenguajes 22

2.2.1 Operaciones con palabras 22

2.2.2 Operaciones sobre lenguajes 22

2.2.3 Expresiones Regulares 25

2.3 Autómatas de Estado Finito 25

2.3.1 Definición de Autómata de Estado Finito 26

2.3.2 Autómatas Productores de Lenguajes 29

2.3.3 Generadores 30

2.4 Autómata de Estados Finitos Comunicantes 32

2.4.1 Composición de autómatas comunicantes 33

2.4.2 Máquinas de Estado Finito Extendidas. (MEF-E) 34

2.4.3 Lenguajes de descripción de autómatas 35

2.5 Redes de Petri 35

2.6 Verificación de Sistemas de Transiciones 38

3 Modelado y Control de SED's **41**
3.1 Introducción . . . 41
3.2 Modelado de SDED's . . . 42
3.2.1 Eliminación de transiciones no posibles físicamente . . . 45
3.3 Supervisores y Estructura de Control . . . 46
3.3.1 Lenguaje controlado . . . 47
3.3.2 Supervisores de SED . . . 48
3.3.3 Realización de un supervisor mediante un SDED . . . 49
3.4 Controlabilidad y Existencia de Supervisores . . . 50
3.4.1 Cálculo de Lenguajes Controlables . . . 51
3.5 Controlabilidad y existencia de supervisores . . . 53
3.5.1 Síntesis de supervisores y Controlablidad Condicional . . . 53
3.6 Síntesis modular de supervisores . . . 55
3.7 Control Optimo para SDED . . . 55
3.8 Control Jerárquico de Sistemas Discretos . . . 55

4 Bases para el desarrollo de Sistemas Híbridos **57**
4.1 Introducción . . . 57
4.1.1 Clasificación Básica de HyDS . . . 57
4.2 Autómatas . . . 58
4.2.1 Modelo de Krogh . . . 58
4.2.2 El modelo de Antsaklis, Lemmon, Stiver . . . 59
4.2.3 Viabilidad de Sistemas Híbridos. Khon, Nerode, Remmel y Yakhnis . . . 60
4.2.4 El enfoque de implantación de Lennartson, Tittus, et alea . . . 61
4.3 Sistemas Dinámicos. . . . 64
4.3.1 El Modelo Dinámico propuesto por Branicky . . . 65
4.4 Estructuras algebraicas. . . . 65
4.5 Lenguajes de programación. . . . 66
4.6 Un enfoque algebraico abstracto para SDH . . . 66
4.6.1 Sistemas Dinámicos . . . 67
4.6.2 Sistemas pseudo-dinámicos . . . 67
4.6.3 Mapa con preservación de la dinámica . . . 69
4.6.4 Equivalencia entre sistemas E/S y $Ps - DS\&OM$ . . . 69
4.6.5 Una clase de Sistemas de Control Dinámicos como un $Ps - D^cS\&OM$ . . . 70
4.7 Definición general para los sistemas híbridos . . . 70
4.8 Ejemplos de sistemas híbridos . . . 71
4.8.1 El sistema Tres Tanques en términos de $HyDS$ . . . 71

5 Modelado y Control de Sistemas Híbridos **75**
5.1 Algunas ideas sobre el modelado . . . 75
5.2 Metodología para la construcción del modelo . . . 76
5.2.1 Modelado de las Dinámicas internas . . . 76

6 Conclusiones **81**

Parte I

La Automatización Integrada de Sistemas

Capítulo 1

Introducción a la Automatización Integrada

1.1 El Modelo CIM. Una referencia

El ambiente de manufactura ha cambiado grandemente con el avance de la tecnología en los últimos años. Este cambio se debe entre otras cosas a la búsqueda de una mayor competitividad de las empresas, con el fin de lograr maximizar sus ganancias, o simplemente, mantenerse en el mercado. Dentro de los aspectos tecnológicos que han incidido en el desarrollo de la tecnología, se tienen el auge de la electrónica, que condujo al desarrollo del microprocesador, los avances en comunicaciones y en especial el desarrollo de las redes de computadores. El proceso de manufactura ha cambiado del sistema donde operarios manejaban maquinas herramientas de manera individual a secuencias de operaciones, donde algunas máquinas trabajan de manera semi-automática, hasta llegar a los métodos modernos de producción, donde mediante computadores se generan las secuencias de operación, permitiendo la reconfiguración de las plantas de manera automática para lograr reducir los tiempos de producción con el reuso de los sistemas existentes. El diseño de estas nuevas factorías se basa en el concepto de Sistemas Flexibles de Manufactura y Celdas Flexibles de Manufactura. El incremento de la productividad en una empresa envuelve diferentes aspectos que van desde la mejora en los diseños de lo procesos de producción hasta un mejor aprovechamiento de las oportunidades del negocio, logrado mediante un conocimiento del mercado. La automatización debe proveer una infraestructura que permita cubrir todos las fases y aspectos del proceso productivo con el fin de lograr un incremento de la producción, manteniendo o reduciendo los costos.

Un ambiente de Manufactura Integrada por Computador, no solo envuelve el ambiente de transformación de los materiales, el secuenciamiento de la producción y el diseño de los productos, ligados directamente a las actividades de producción, sino también a las actividades de mercadeo, finanzas, personal e ingeniería. Una fábrica totalmente automatizada, dispone de la información al alcance de los usuarios para la realización de todas las actividades presentes en la industria.

Se considera un sistema de producción automatizado, aquel donde la producción es principalmente coordinada y controlada mediante sistemas automáticos. En los procesos de manufactura, el modelo mayormente usado es el modelo CIM "Computer Integrated Manufacturing". CIM envuelve los aspectos de control directo (manufactura y ensamblaje), planificación de la producción, diseño de piezas, así como también los aspectos administrativos y gerenciales de la planta.

Otra terminología también utilizada en automatización de manufactura incluye: los Sistemas de Manufactura Flexible, Celdas de Manufactura Flexibles y CAD/CAM ("Computer Aid Design / Computer Aide Manufacturing").

Los procesos de automatización de manufactura difieren de los de producción continua en los modos de producción, los métodos de control y los dispositivos requeridos. Sin embargo, en ambos casos, el proceso de automatización es jerárquico, con funciones específicas a cada nivel de la jerarquía. La cantidad de información necesaria y el nivel de precisión depende del nivel de la jerarquía en la automatización. Saridis [43] propone un esquema, mostrado en la figura 1.1, aplicado al diseño de robots, que caracteriza la información necesaria según el nivel de la jerarquía donde se están realizando las tareas. La precisión en los niveles inferiores es mucho mayor, tanto en tiempo, como a nivel de detalle, sin embargo el contenido de la información es más simple. Tomemos como ejemplo un sistema de control para un robot que posee un sistema de posicionamiento, un sistema de visión y un sistema de coordinación global. El sistema de visión trabaja con la imagen digitalizada, la cual debe ser actualizada a medida que el sistema de posicionamiento avanza. La información varía muy rápidamente y la imagen debe ser reconstruida continuamente. El elemento que comunica el sistema de reconocimiento de imágenes con el sistema de posicionamiento, trabaja con una frecuencia menor; ésta es determinada por la finalización de cada etapa de reconocimiento. El sistema de posicionamiento recibe información del sistema de coordinación, donde se determina distancia y dirección, fijando un nuevo objetivo. La tarea de fijar un nuevo objetivo presenta una mayor complejidad en procesamiento y en los datos necesarios para la realización de la misma.

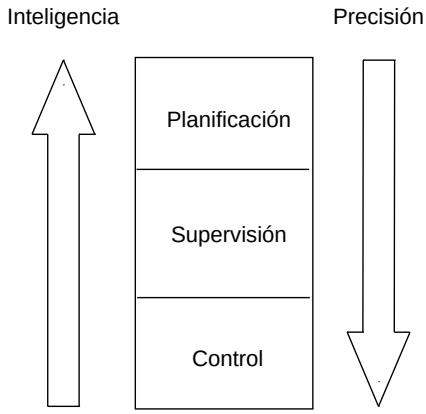


Figura 1.1: Inteligencia y Precisión en un Sistema Jerárquico

CIM trata de modelar la fábrica del futuro, donde la fabricación de un producto, desde su conceptualización hasta su venta, es realizado mediante procedimientos totalmente automatizados, con la mínima intervención humana. Las celdas de manufactura, correspondientes a unidades de producción semi-autónomas cooperantes es una tendencia para mantener una empresa altamente competitiva [27]. El proceso de integración de la manufactura se inicia con el diseño del producto mediante herramientas de *Diseño Asistido por el Computador* **(CAD)**, con el cual se generan los moldes de las piezas a ser ensambladas así como las etapas de ensamblaje. Esta información es la entrada para la fase de producción, donde mediante tcnicas de *Manufactura Asistida por el Computador*, **(CAM)** se realizan las tareas de manufactura. Estas etapas se muestran en la figura

1.2

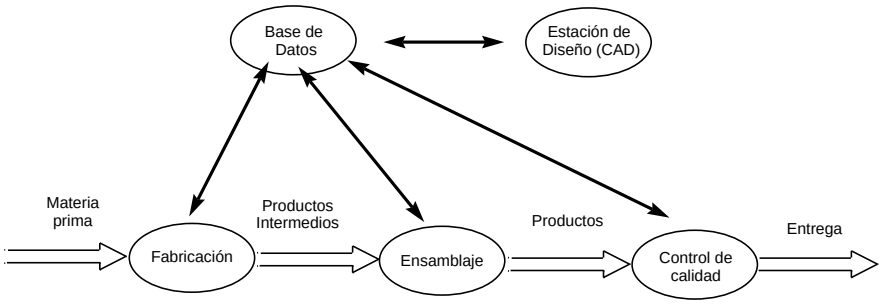


Figura 1.2: Etapas en una fábrica totalmente automatizada

1.1.1 El modelo CIM

El modelo CIM, propuesto por la International Standards Organization (ISO), considera que la empresa está organizada en seis niveles funcionales, los cuales se usarán para la integración de los procesos de producción tal como lo indica Pimentel en [36]. Estos niveles son:

- **Fábrica**
  El nivel de fábrica es el nivel más alto, incluye el procesos de planificación, gerencia de la producción, planificación a largo plazo, los aspectos asociados a las finanzas y manejo de personal, el mercado y las demás funciones administrativas.

- **Unidades de producción**
  En este nivel, se maneja la coordinación de los recursos y tareas necesarias en cada unidad de producción. Una unidad de producción se considera dedicada a un producto o línea de productos específicos. Agrupa varias celdas de ensamblaje, distribuye y coordina las actividades entre las diferentes celdas.

- **Celdas de ensamblaje**
  Cada celda de ensamblaje contiene los sistemas de control para el secuenciamiento de las tareas asignadas a la misma. Las tareas se agrupan de acuerdo a la similitud y dependencia entre las mismas. Maneja el secuenciamiento del material a través de las diferentes unidades de trabajo.

- **Nivel de estación de trabajo**
  Una estación de trabajo es uno o más equipos que efectúan una actividad única. Se considera en este nivel un robot, una máquina-herramienta. Una unidad de transporte o almacenamiento. Una estación de trabajo recibe ordenes de la coordinación de la celda de ensamblaje y ejecuta tareas o secuencias de operaciones asignadas a la celda de ensamblaje.

- **Nivel de equipo**
  Es el nivel inferior en la arquitectura CIM. Consiste en los controladores para recursos individuales, tales como un robot y los equipos sensores y actuadores asociados al mismo.

1.2 Automatización de Procesos Continuos

Los sistemas dinámicos pueden variar desde sistemas discretos puros como el computador, una red de transmisión de datos; a sistemas totalmente continuos como los que aparecen en la naturaleza. Los primeros sistemas se describen mediante un sistema a transiciones mientras que los segundos son descritos a través de ecuaciones diferenciales o en diferencias. La automatización de cada uno de ellos, exigirá métodos y herramientas diferentes adecuados a cada caso de acuerdo a sus características generales.

Dentro de los esquemas de producción podemos citar:

- **Proceso de Manufactura**
  En los procesos de *manufactura*, las principales funciones son de maquinación de piezas y ensamblaje; la maquinacin forma productos intermedios que luego serán ensamblados. No existe una mezcla de productos como en el caso de la industria química, petrolera o de alimentos.

- **Procesamiento en lotes**
  En el procesamiento por lotes, un proceso recibe un conjunto de materia prima, la cual es transformada en un nuevo producto mediante mezclas y aplicación de energía. Una vez terminado el proceso, éste arranca nuevamente para repetir el mismo o realizar uno nuevo. Este tipo de proceso lo encontramos en la industria siderúrgica, en la industria de alimentos y en gran parte de la industria química.

- **Procesos continuos**
  Los procesos continuos son los procesos de transformación o producción que trabajan de manera permanente. Como ejemplo de los mismos están: las plantas generadoras de energía eléctrica, la producción y refinación de petroleo y el manejo del gas. Algunas plantas procesan productos que son utilizados aguas abajo por otra planta. La mayoría de estos sistemas son considerados sistemas complejos, por el tipo de transformaciones que se dan y por la interrelación entre los procesos que están presentes en el sistema.

Por otra parte, los sistemas a ser automatizados no están aislados, ellos conforman una aglomeración e interactúan y debe existir un mecanismo para la integración entre ellos. Los investigadores han realizado esfuerzos para el manejo de sistemas de producción complejos, definidos así por el número de unidades que lo componen, el comportamiento de cada uno de ellos y las interrelaciones existentes. El *Control Inteligente* y los *Sistemas Autónomos* aparecen entre los resultados más prometedores para resolver los problemas planteados para el manejo de la complejidad.

El *Control Inteligente* [34, 2] es definido como un control que toma decisiones de manera inteligente (antropomorfíca), algunas veces heurística, para manejar problemas no modelados completamente y que se adaptan a nuevas situaciones. El Control Inteligente se refiere mas a los aspectos de la adaptabilidad del control para el manejo de problemas no completamente estructurados que a las técnicas de Inteligencia Artificial.

Algunos autores se refieren a los *Sistemas Autónomos*, como aquellos sistemas de control, trabajando en ambientes donde la intervención del ser humano está muy alejada, como es el caso de los sistemas de navegación espacial, aquellos ubicados en ambientes submarinos o trabajando en ambientes hostiles y que deben tomar decisiones sobre un problema complejo sin la intervención directa de un ser humano. Un sistema autónomo tiene la suficiente inteligencia para evaluar escenarios y tomar una decisión adecuada para la situación en la cual está inmerso. En [2] aparecen diferentes tipos de aplicaciones de sistemas autónomos.

Otra tendencia en los sistemas inteligentes es la representada por Saridis [43] en sus trabajos sobre máquinas jerárquicas inteligentes. Saridis plantea una jerarquía de sistemas inteligentes, donde en los niveles más bajos de la jerarquía (aquellos que están conectados con el proceso) existe una mayor precisión en los datos manejados, así como una mayor frecuencia en la interacción con los procesos, mientras que en los niveles más altos de la jerarquía, los sistemas manejan una mayor complejidad en la información con una menor precisión. Ver figura 1.1 tomada de [43]. Saridis propone una mezcla de tecnologías para la implantación de un sistema jerárquico de máquinas inteligentes, que incluyen aspectos de la teoría de control, las técnicas de Investigación de Operaciones y de la Inteligencia Artificial tal como se muestra en la figura 1.3. Las funciones en los niveles más altos de una máquina inteligente imitan las funciones de un comportamiento humano.

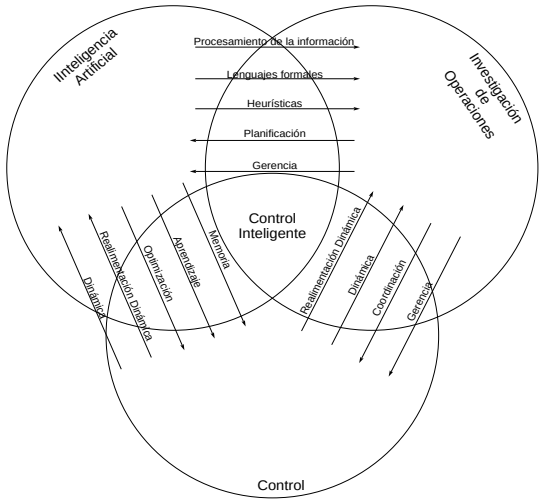


Figura 1.3: Definición de la Disciplina de Control Inteligente

1.2.1 La Pirámide de Automatización

Para la automatización de procesos continuos, la *Pirámide de Automatización*, es la propuesta para la implantación de un sistema automatizado. El modelo propuesto por la ISO [39] es similar al modelo CIM para procesos de manufactura y presenta algunas características descritas por Saridis. Este modelo consta de cinco niveles tal como se ve en la figura 1.4 que cubren las diferentes funciones de una planta coordinada de manera jerárquica, cubriendo los aspectos de control de los procesos físicos en su nivel más bajo, hasta los niveles donde se realizan las funcioes corporativas de la planta. Cada nivel se caracteriza por un tipo de información y de procesamiento diferente. La integración de un proceso automatizado, incluye la comunicación interna en cada nivel, y la comunicación entre niveles, con el fin de lograr sistemas que permitan ejecutar las diferentes tareas de control existentes en una empresa.

Los niveles encontrados en la Pirámide de Automatización son:

- **Empresa**
  Al igual que en el más alto nivel del modelo CIM, a nivel de empresa, el sistema comprende

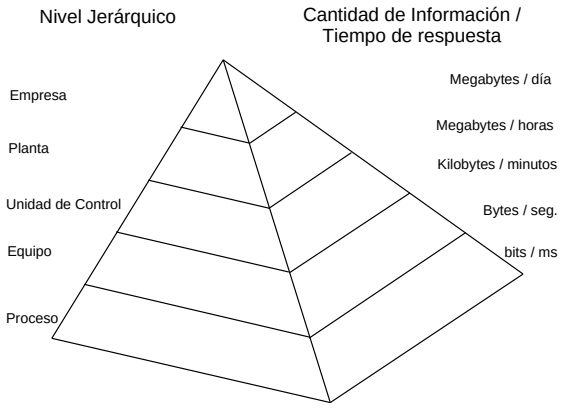


Figura 1.4: La Pirámide de Automatización

los problemas de gerencia de producción, fijación de estrategias y niveles de producción, que están asociados a políticas globales de la empresa, el factor financiero y el mercado. A este nivel se coordinan las actividades entre las diferentes plantas o unidades completas de producción. El nivel de información manejada es muy grande. Esta información proviene de las plantas y de los factores externos a la empresa. La fijación de estrategias y niveles de producción tienen una larga duración ya que involucran funciones de planificación.

- **Planta**
  Este corresponde al siguiente nivel del modelo ISO. Es la responsable del logro de metas de producción fijadas a la planta, mediante el manejo y coordinación de los recursos (económicos, humanos y equipos) para la ejecución de las diferentes actividades. Funciones como el establecimiento de parámetros y criterios de producción son realizados a este nivel. Los resultados obtenidos de las funciones realizadas a nivel de planta, directivas, son enviados al nivel de supervisión para que sean implantadas mediante el equipamiento disponible a nivel supervisorio. El nivel planta coordina las diferentes unidades de procesamiento bajo su responsabilidad. Labores como optimización y planificación de las tareas de producción se realizan a este nivel. El monto de información necesaria para estas labores es muy alto, sin embargo el tiempo promedio de separación entre la ejecución de estas actividades varía entre los días y semanas.

- **Supervisión**
  El control supervisorio tiene como función la coordinación de las diferentes unidades de procesamiento o transformación del material, mediante la parametrización de los controladores. A nivel de supervisión, los operadores de planta controlan el proceso. La fijación de los parámetros es en la mayoría de los casos establecida por el operador, aunque puede ser apoyada por sistemas en línea.

- **Unidad de Control**
  Los elementos que realizan el control de los procesos, trabajan de acuerdo a una parametrización recibida desde el nivel supervisorio. Estos sistemas actúan de manera automática manteniendo el control regulatorio y/o de secuencias de los procesos productivos, mediante una

realimentación del proceso.

- **Nivel de Proceso**
  El nivel de proceso se refiere a los instrumentos y equipos que están en contacto directo con el proceso. Por lo general son dispositivos electro-mecánicos con posiblidad de comunicación.

1.3 Arquitectura para la Automatización Integral

1.3.1 Elementos de Automatización

En cada nivel, un grupo de elementos, tanto físicos como lógicos, permiten conocer y actuar sobre el proceso, para conducirlo a los requerimientos de producción fijados para la empresa.

- **Sistemas físicos**
  Los sistemas físicos de transformación. El conjunto de máquinas - herramientas que permiten transformar o extraer un producto. Estos sistemas físicos que obedecen a unas leyes físicas, pueden ser comandados o controlados por otros elementos.

  - Los sistemas de almacenamiento, que corresponden a los dispositivos y lugares donde se almacena la materia prima, productos intermedios o productos finales.
  - Elementos de transporte. Movilizan materiales desde un depósito o lugar de transformación a otro punto. Ejemplos: Correas transportadoras, tuberías, líneas de transmisión, etc.
  - Sistemas sensores y actuadores. Interfaz con el proceso físico, por medio de los cuales se adquiere información necesaria para la determinación del estado del proceso y de la modificación del mismo mediante una modificación de sus entradas.
  - Los sistemas de procesamiento de información y de interfaz hombre/maquina. Finalmente la información del proceso bajo control debe ser almacenada, procesada y comunicada a los operadores mediante equipos orientados al control y a la interafaz con el operador. El computador pasa a ser un elemento fundamental en la automatización, puesto que es el soporte de los sistemas lógicos que controlan el proceso, permiten la transferencia de información que asegura la posibilidad de una coordinación final.

- **Sistemas lógicos**
  Dentro de los sistemas lógicos nos referimos a los sistemas responsables del mantenimiento y actualización de la información, la ejecución de la toma de decisiones y la interfaz hombre-máquina.

  - Sistemas programados para control y supervisión.
  - Sistemas de control de producción.
  - Sistemas de optimización y secuenciamiento de la producción.

Estos últimos trabajan con información proveniente del proceso, para lograr que cumpla con los objetivos asignados. Para la transmisión de la información entre los diferentes elementos y entre los diferentes niveles, es necesario la existencia de medios de comunicación. Estos medios de comunicación permiten la comunicación sensor - unidad de procesamiento y unidad de procesamiento - actuador. Asumiremos los términos actuador y sensor en el sentido más ampio posible.

Un sensor es un elemento físico o lógico que permite la medición del estado de una variable o un proceso complejo. En el último caso, un sensor puede resultar del agregado de un conjunto de variables. De la misma manera un actuador es una válvula, un relé a nivel de proceso, pero se puede considerar a un controlador como un actuador para el nivel supervisorio.

Dentro de una pirámide de automatización, la jerarquía mostrada en la figura 1.4 muestra el concepto de sensor - actuador para los diferentes niveles. El sensor permite medir variables que pueden ser medidas directamente del proceso, o de una base de datos donde se almacena información sobre el proceso; esta información a través de un observador, suministra el estado del proceso. De la misma manera el actuador funciona como un elemento de transformación del proceso, o de su control dependiendo del nivel en la pirámide de automatización. El controlador es un elemento que toma decisiones para transformar el proceso en función del estado del mismo. Las decisiones varían desde un valor de apertura a una válvula, una consigna de flujo a ser mantenida por una válvula inteligente, un volumen de producción de una calidad de producto, etc. Las estrategias de producción se pueden considerar como los comandos que son enviados al supervisor y que modifican la producción de la planta. La organización del sistema de control es entonces la mostrada en la figura 1.5.

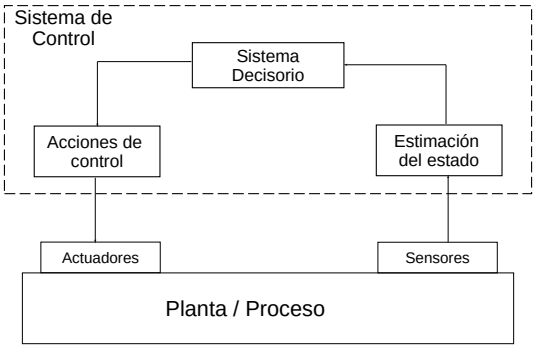


Figura 1.5: El Sistema de Control: Componentes

Una de las premisas utilizadas en automatización es la de mantener los procesos controlados de manera autónoma, con equipos que tengan una alta tolerancia a fallas. Esto implica que sistemas con multiple unidades, deben tener cada una su sistema de control, pero es necesario mantener una información común entre ellos, que aseguren un óptimo desempeño global, para lo cual tienen que estar interconectados y de esa manera transmitir información entre ellos acerca de su estado de operación. La información de más bajo nivel almacenada localmente, en sistemas orientados al control regulatorio. Estos equipos son construidos tolerantes a fallas, con un alto rendimiento en la ejecución de lazos simples de control; aunque sus características los hacen difíciles de programar. Los sistemas del segundo nivel deben mantener la comunicación con el operador, coordinar todos los procesos y servir de interfaz con los sistemas administrativos. El tercer nivel está relacionado con la planificación a largo plazo así como las interacciones con el ambiente externo a la empresa. La figura 1.6 muestra la arquitectura informática general de una empresa de producción continua.

Los niveles en la Pirámide de Automatización se van a corresponder con la infraestructura de automatización instalada. De la figura 1.6, observamos que el equipo instalado a nivel del proceso, contiene los dos primeros niveles de la Pirámide, que son la Unidad de Control y los dispositivos de interfaz con el proceso: sensores y actuadores. El centro de control, es el lugar donde se concentra

la información de cada planta, el centro de control supervisa los procesos utilizando los sistemas de comunicación con los dispositivos de control directo, coordina la producción realizada en cada planta y optimiza la misma. Las funciones relativas a la gestión de planta y la coordinación entre las mismas al más alto nivel son realizadas mediante los equipos asignados a la administración y los equipos utilizados para el modelado y diseño de los procesos. De ésto se nota que existe un mapeo entre las actividades asociadas al control y la arquitectura informática que existe en la planta. Para el desarrollo de esta infraestructura informática y su relación con la estructura de control, varios modelos se han desarrollado como son: El modelo de la Universidad de Purdue, el modelo CIMOSA y el modelo GREI, descritos en [51].

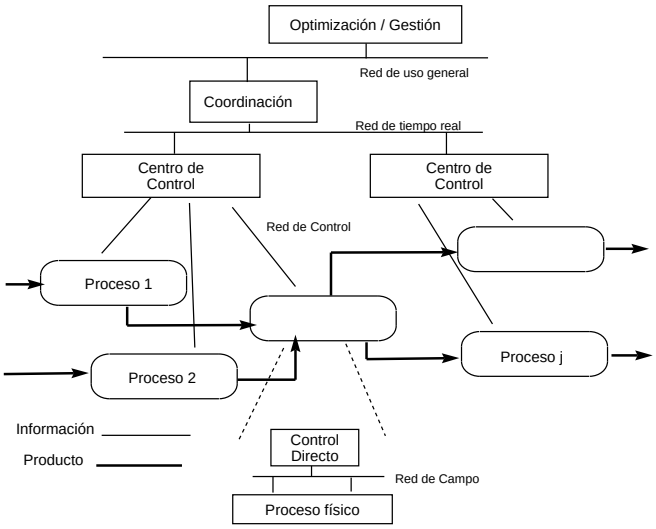


Figura 1.6: Arquitectura Teleinformática de Automatización

En los próximos capítulos nos referiremos tanto a las herramientas teóricas que permiten el desarrollo de un sistema de automatización integral, basado en el uso de los sistemas jerárquicos, sistemas discretos y sistemas híbridos, para luego continuar con una descripción de los elementos físicos de control de acuerdo a la infraestructura teleinformática, los dispositivos de comunicación internivel e intranivel, para luego tocar los aspectos relativos al software para el desarrollo de los sistemas de control y la integración de aplicaciones, necesaria para el objetivo de una planta automatizada integralmente.

1.4 Ejemplo de un sistema complejo

Los sistemas de producción, tanto continuos o de manufactura, muestran dos aspectos que deben ser tomados en cuenta en un proceso de automatización para optimizar y mejorar su producción. El primero tiene que ver con el sistema físico: sus características físicas, su estructura, número de subsistemas y las interconexiones entre ellos y con el ambiente donde está inmerso. Este conocimiento permite obtener un mayor rendimiento del mismo y lograr tener una producción de acuerdo a unos requerimientos establecidos. El segundo tiene que ver con aspectos estratégicos como son: el momento cuando el sistema debe trabajar y por cuanto tiempo debe estar en funcionamiento; su modo de operación para lograr un mayor rendimiento del mismo en base a las relaciones con otros sistemas y con el contexto de producción.

Para entender un poco mejor estos aspectos de automatización, tomemos como ejemplo el caso de un sistema compuesto de dos unidades semi-independientes: una torre de enfríamiento, que envía agua fría para todo un complejo industrial y un sistema de regulación de temperatura para un producto que debe asegurar una temperatura de salida constante para el líquido a partir del uso de agua enfríada anteriormente. Este ejemplo fue tomado de [13]. La figura 1.7 muestra el sistema a controlar. Cada uno de los sistemas puede ser controlado de manera autónoma, de acuerdo a sus características propias, y sólo se transmite entre ellos información acerca de sus puntos de operación, para permitir al otro sistema trabajar de manera óptima.

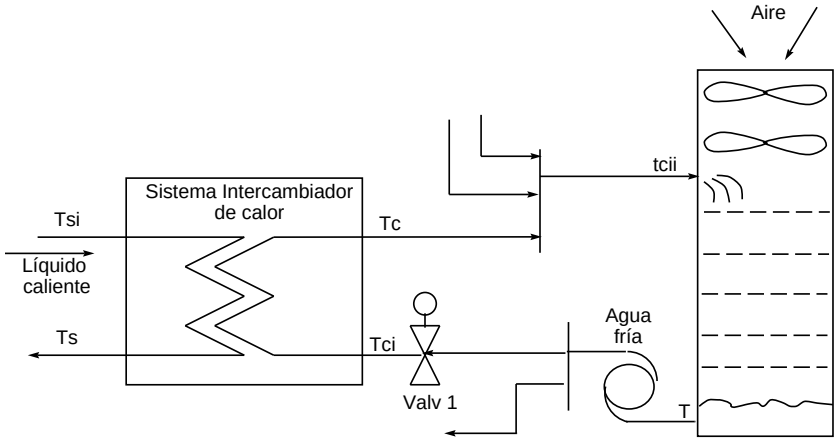


Figura 1.7: Sistema Regulador de Temperatura

En el ejemplo mencionado, nos encontramos con la composición de dos sistemas de control que funcionan para controlar la temperatura de salida del líquido, y la temperatura del refrigerante a la salida de la torre de refrigeración, respectivamente. Un sistema coordina a los dos procesos mediante la fijación del volumen de entrada al intercambiador de calor. El control de cada uno de ellos y el de su *composición* se realiza mediante un conocimiento del estado del proceso (determinación del estado), cálculo de los valores de control (Sistema de toma de decisiones) y el envío de las señales de control que pueden modificar la conducta del proceso. La figura 1.8 nos muestra los diferentes elementos que permiten conocer el estado del proceso y como actuar sobre él. Se nota la existencia de un controlador para el intercambiador de calor, otro dedicado al control

de la torre y uno final para la coordinación de los dos subsistemas.

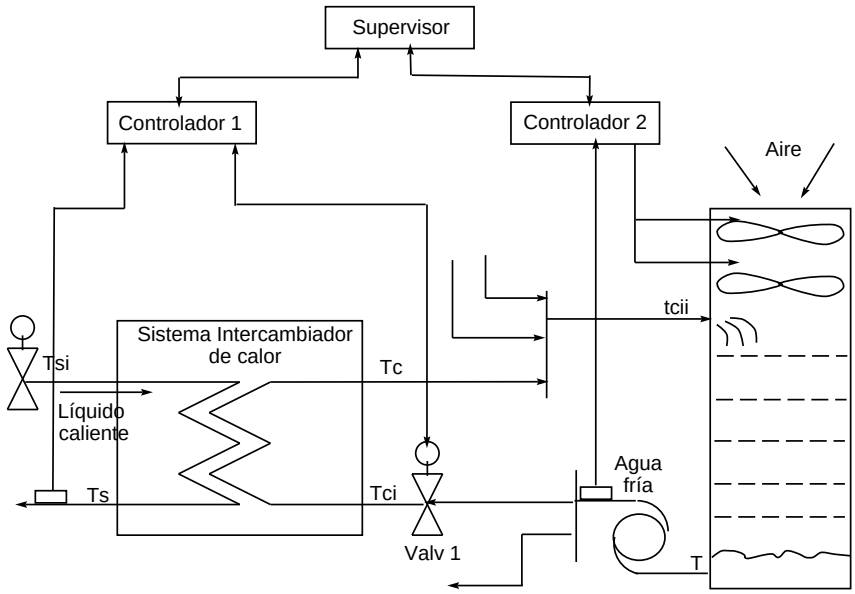


Figura 1.8: Sistema de Control Realimentado

Cada nivel tiene sus propias técnicas para la implantación de los procedimientos de control, pero todos trabajan de acuerdo a una determinación del estado del proceso. Este estado puede ser físico como el de temperatura del producto y posición de las válvulas, se implantará tomando en cuenta las restricciones de tiempo y tenderán a ser periódicas; por lo general estos sistemas usan relojes para controlar el momento de ejecución de las tareas necesarias para realizar su labor de control. En el caso de los sistemas de coordinación, donde el estado del sistema es ante todo un estado lógico como sería la regla de control a ser usada en el dispositivo regulador de temperatura, o el nivel de temperatura deseado a la salida del intercambiador de calor; el estado cambia en instantes no predeterminados al avance, por lo que la implantación del mecanismo de control es asíncrono.

1.5 El Proceso de Automatización

1.5.1 Sistemas Continuos por regiones de operación

Del ejemplo que se muestra en la figura 1.7, describiremos un mecanismo sencillo para el control de la temperatura Ts del líquido **s** mediante un intercambiador. El sistema se controla mediante la apertura o cierre de la válvula de entrada de agua fría **r** o refrigerante, que viene de la torre de enfríamiento, con temperatura Tri . La temperatura de entrada del líquido Tsi, al igual que su flujo Fs, no son controlados por este sistema. La posición de la válvula es conocida en todo momento por el controlador y su posición varía de 90 grados (completamente abierta) a 0 grados

(completamente cerrada). Las ecuaciones que representan el modelo del sistema son:

$$\dot{T}s = (Fs/Vs) \times (Tsi - Ts) - (U \times A)/(Vs \times \rho_l \times Cps) \times (Ts - Tr)$$

$$\dot{T}c = (U\times)A)/(Vc \times \rho_r \times Cpc) \times (Ts - Tr) - (Fc/Vc) \times (Tr - Tri)$$

Para variar la temperatura, se aplica un controlador proporcional – integral de acuerdo a la función:

$$f_r = K_c \times \left(error + 1/T_i \int errordt \right)$$

donde:
f_r representa la acción de apertura o cierre de la válvula,
error representa la diferencia de la temperatura del líquido a la salida del intercambiador y una referencia.

El sistema puede alcanzar los valores de referencia si: la temperatura de entrada del refrigerante es inferior a un valor dado, o el flujo del líquido refrigerante es relativamente pequeño.

La simulación del sistema nos da los valores que se muestran en la figura 1.9 para una entrada con perturbaciones tanto en el flujo como en la temperatura y una temperatura del refrigerante constante. En este caso el flujo del refrigerante no hace posible que se alcance la temperatura de salida deseada.

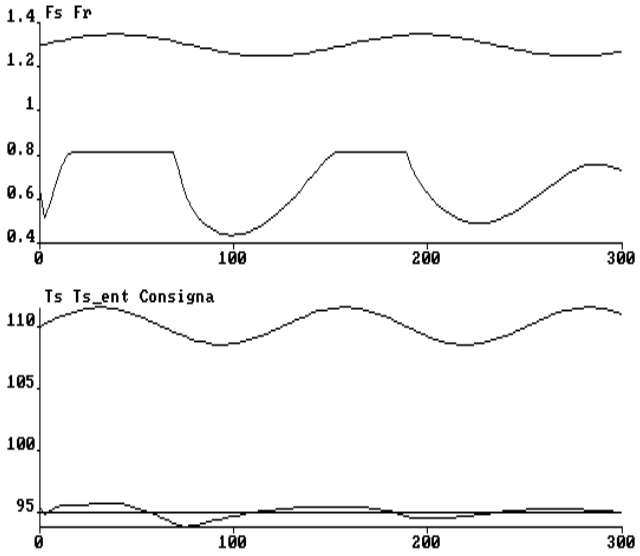


Figura 1.9: Temperatura de Salida del Intercambiador.

Si la temperatura no puede ser mantenida dentro de valores aceptables, el sistema de control debe detectar esas inconsistencias y enviar la información que permita a los otros sistemas actuar, para que de esa forma, el conjunto *intercambiador – torre de enfríamiento*, mantenga la temperatura deseada. Las acciones en este sentido pueden ser sobre la entrada, mediante una disminución en el caudal del líquido de entrada, o sobre el refrigerante mediante una disminución de su temperatura. El controlador de la torre de enfríamiento establece de manera adecuada, los

valores que permitan alcanzar al primer equipo los valores requeridos, mediante una disminución de la temperatura del refrigerante. En este caso se está realizando un control por zonas, donde a cada zona le puede corresponder una ley de control diferente. En la figura 1.10 se muestra como la consigna es alcanzada si se disminuye el flujo del líquido de entrada para las mismas condiciones

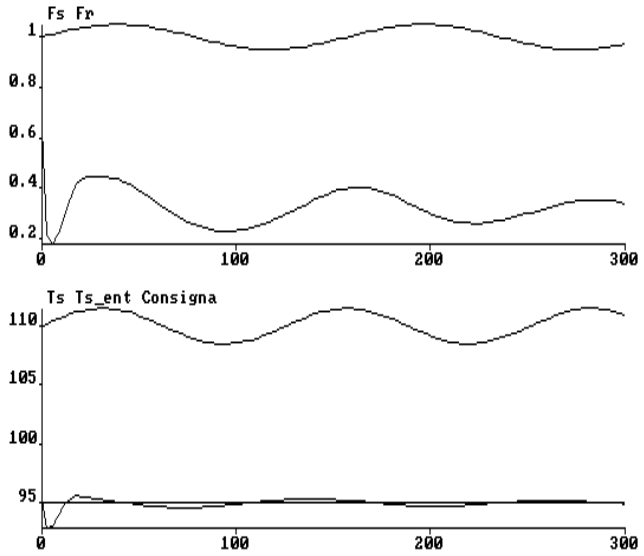


Figura 1.10: Temperatura de Salida del Intercambiador para otro flujo de entrada.

1.5.2 Sistemas con una Dinámica Discreta

Para el control de la torre de enfríamiento, el sistema descrito trabaja mediante un control a estados. Esto es, el flujo del aire es modificado mediante el funcionamiento de 2 ventiladores que pueden estar ambos apagados, ambos encendidos o uno solo de ellos encendido. El sistema tratará de regular la temperatura para las referencias establecidas a un nivel superior mediante el esquema siguiente; si la diferencia de temperatura es mayor a un cierto nivel de referencia, se enciende un ventilador, y si la diferencia de temperatura es aún mayor que un cierto límite, se enciende un nuevo ventilador. Las ecuaciones simplificadas que describen el proceso de enfríamiento son:

$$\dot{T}ri = (Frt/Vtr) \times (Tr - Tri) - (1 + B) \times \rho_{aire} \times (Trii - Taire)$$

donde B es una variable que incluye la masa de aire que está circulando en la torre de enfriamiento. Esta variable toma tres valores según el el número de ventiladores que esté funcionando.

La temperatura del agua a la salida de la torre depende de diversos factores como son: caudal del agua y temperatura de entrada del caudal, temperatura del medio ambiente. La variable de control es la cantidad de aire que pasa por la torre de refrigeración. Para incrementar la masa de aire en contacto con el agua, es necesario encender los ventiladores. Si el criterio es el de minimizar el consumo de energía, los ventiladores solo se encienden cuando la temperatura del agua alcance un cierto rango. Si dado un cierto tiempo la temperatura del agua continúa subiendo, se enciende el ventilador. En la figura 1.11 se muestra la evolución de la temperatura de salida

para una temperatura y volumen del refrigerante constantes, haciendo abstracción de los cambios de temperatura en el ambiente ante un cambio en el encendido de los ventiladores.

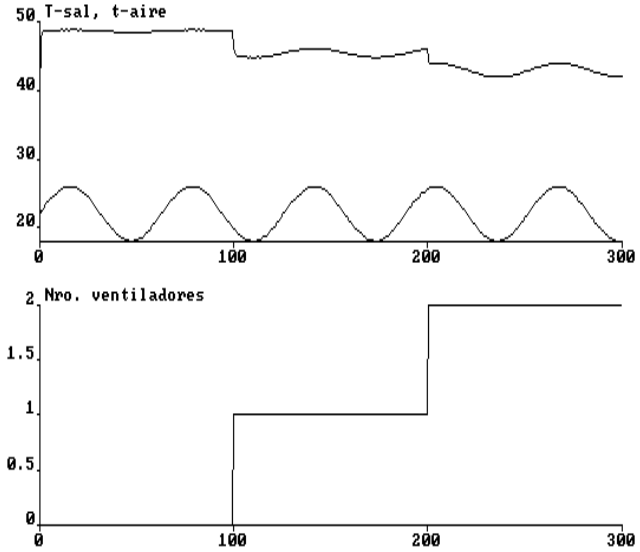


Figura 1.11: Evolución de la Temperatura en la Torre

El modelo que representa el comportamiento de la torre de refrigeración da como resultado un autómata con cuatro estados (Frío (s1), Normal (s2), Caliente (s3) y Muy Caliente (s4)). La evolución del mismo se da en la figura 1.12.

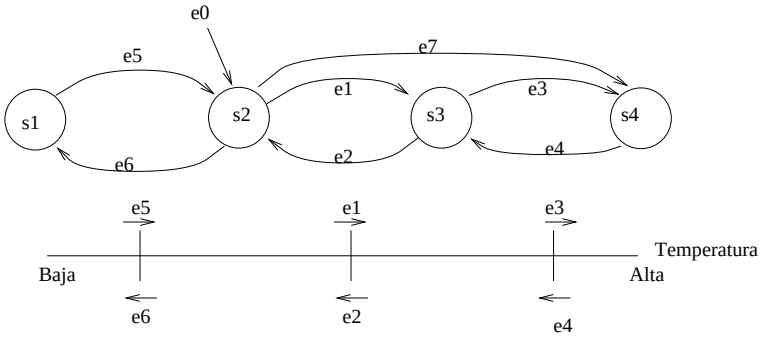


Figura 1.12: Auómata de Evolución de la Temperatura

El comportamiento del sistema de control que mantiene la temperatura de la torre dentro del rango aceptado para el intercambiador se muestra mediante el autómata de la figura 1.13 de acuerdo a la descripción dada anteriormente.

Se hace necesario tener un sistema de coordinación, que conozca el estado de cada uno de los dos sistemas y, mediante la generación de eventos, hacer que cada uno de los sistemas actúe de la manera adecuada, para que el conjunto cumpla con los objetivos. Los criterios de optimización establecidos pueden definir nuevos parámetros de operación como son:

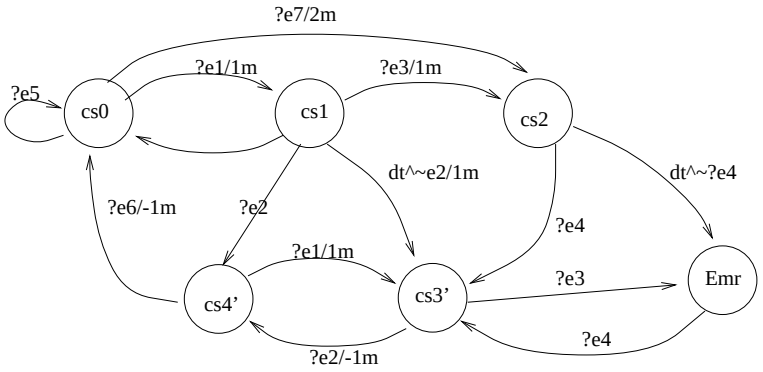


Figura 1.13: Comportamiento del Sistema de Regulación de Temperatura

- Disminución del volumen de entrada del producto a refrigerar.
- Nuevo modo de operación de la torre de refrigeración.
- Cambio en el procesamiento aguas abajo para permitir usar el producto a una temperatura más elevada.

Todos estos parámetros son fijados en otro nivel y los controladores directos solo tienen como función asegurar que la operación se desarrolle normalmente. Esta distribución de las tareas forma una *jerarquía en el control* que es modelada mediante una organización jerárquica de autómatas en el caso más sencillo. En un primer nivel nos encontramos que existen sistemas que tienen una parte continua, pero que sus puntos de consigna, parámetros de control o leyes de control son fijados de manera discreta; en este caso el sistema se trata mediante metodologías orientadas a los *Sistemas Híbridos*. En los siguientes capítulos, trataremos el caso de los sistemas discretos y los sistemas híbridos e iremos resolviendo el problema del control integrado del sistema, tomando en cuenta la parte continua, con los controles específicos para cada región de operación, como el conmutador que selecciona la ley de control adecuada en cada caso.

Parte II

Sistemas Dinámicos en Automatización

Capítulo 2

Sistemas Discretos

2.1 Introducción

La mayoría de los sistemas que se encuentran en la naturaleza, tienen una dinámica continua por lo cual pueden ser descritos mediante un sistema de ecuaciones diferenciales o en diferencias como es el caso de los cambios de temperatura a lo largo de un período de tiempo, el sistema gravitacional, etc. Algunos de los sistemas creados por el hombre también poseen una dinámica continua, como es el caso de los procesos químicos, algunos sistemas mecánicos, etc. Otros sistemas, la mayoría de ellos creados por el hombre, presentan una dinámica que debe ser descrita de otra manera, formando sistemas altamente no lineales, para los cuales se usan sistemas de transiciones. Los sistemas discretos tienen su aplicación en las comunicaciones, en los sistemas de control de tráfico, etc., teniendo como el caso más conocido el computador. Los modelos basados en transiciones describen el comportamiento de un sistema mediante la enumeración de los eventos desde un estado inicial, o mediante una historia de los cambios de estados ocurridos a partir de un estado inicial. Dentro de los sistemas de transiciones tenemos:

- *Los Autómatas de Estado Finito*
- *Las Redes de Petri y los Sistemas Grafcet*
- *Los sistemas de lenguajes de programación y la programación lógica*

Existen otros sistemas que incluyen, tanto una dinámica continua como una dinámica discreta y son los sistemas híbridos. Para estos sistemas, la dinámica sale de la ejecución de procedimientos para la ejecución de tareas de producción. Si bien la fundamentación teórica ha sido bien establecida para la descripción de sistemas discretos, sólo trabajos recientes han abordado el problema de la síntesis de controladores para sistemas procedimentales. A pesar de que los ingenieros de proceso han estado implementando los sistemas procedimentales, la formalización de los mismos en el desarrollo de aplicaciones no es de uso común.

Un sistema discreto puede ser controlado mediante el secuenciamiento adecuado de las entradas que hacen que ocurran las transiciones en el sistema. Los conceptos de *estado, controlabilidad, observabilidad* clásicos en la teoría de control, se extienden a los sistemas discretos y a los sistemas procedimentales.

2.2 Lenguajes

Un lenguaje resulta de la concatenación de un conjunto de símbolos o letras, pertenecientes a un alfabeto Σ. Las diferentes cadenas de símbolos que se pueden formar, definen las palabras del lenguaje. Por ejemplo, para $\Sigma = \{\alpha, \beta, \gamma, \delta\}$ se pueden definir los lenguajes siguientes:

$L_1 = \{\alpha, \alpha\beta, \alpha\gamma, \beta\delta\}$

$L_2 = \{$Todas las cadenas de longitud 3 comenzando por $\alpha\}$

$L_3 = \{$Todas las cadenas de longitud finita posibles y que comiencen por $\beta\}$.

Dentro de un lenguaje, una palabra que puede formar parte del lenguaje es la palabra vacía ϵ, que es una palabra de longitud cero; esto es, que no posee ningún simbolo.

Se denota como Σ^* al conjunto de todas las palabras posibles formadas con los símbolos pertenecientes a Σ, incluyendo ϵ, $card(\Sigma^*) = \infty$.

2.2.1 Operaciones con palabras

Concatenación

En un alfabeto se define una operación que es la operación de concatenación. La concatenación de símbolos genera una palabra, y la concatenación de palabras genera una nueva palabra. Sea $u = \alpha\beta\beta$ y $v = \gamma\beta$, la concatenación de las plabras u y v genera la palabra uv, donde $uv = \alpha\beta\beta\gamma\beta$.

Propiedades de la concatenación

1. *Operación cerrada*: La concatenación de dos palabras de Σ^* es una palabra de que pertenece a Σ^*.
$$u \in \Sigma^* \& v \in \Sigma^* \Rightarrow uv \in \Sigma^*$$
2. *Propiedad asociativa*
$$u(vw) = (uv)w$$
Luego, (Σ^*, *concatenación*) es un semigrupo.
3. *Existencia de un elemento neutro.* Para un lenguaje, se define el elemento neutro ϵ o palabra vacía, como el elemento que al concatenarse genera la misma palabra con la cual se concatenó.
$$\epsilon\alpha = \alpha\epsilon = \alpha$$
Luego, (Σ^*, *concatenación*) es un monoide.
4. La operación concatenación de palabras no es conmutativa, ya que en general $uv \neq vu$.
5. *Longitud de una palabra.* La longitud de una palabra viene dada por la cantidad de símbolos existentes en la palabra. Al concatenarse dos palabras, la longitud de la nueva palabra resulta de la suma de longitudes de las palabras concatenadas. Notación: Sea $u \in \Sigma^*$ y $u = u_1u_2 \dots u_n$, $|u| = n$.

2.2.2 Operaciones sobre lenguajes

Se conoce como lenguaje: L sobre un alfabeto Σ a todo subconjunto del lenguage universal de Σ^*

$$L(\Sigma) \subseteq \Sigma^*$$

En particular el conjunto vacío $\emptyset$ es un subconjunto de Σ^*, y se conoce como el lenguaje vacío. Para simplificar la notación, escribiremos $L(\Sigma) = L$.

Unión de lenguajes

Sean dos lenguajes definidos sobre el mismo alfabeto, $L_1 \subset \Sigma^*$, $L_2 \subset \Sigma^*$. Se conoce como unión de los dos lenguajes L_1, L_2 al lenguaje definido como:

$$L_1 \cup L_2 \;=\; \{u \mid u \in L_1 \text{ ó } u \in L_2\}$$

Propiedades de la operación unión de lenguajes

1. *Operación cerrada.* La unión de dos lenguajes sobre el mismo alfabeto, es también un lenguaje sobre dicho alfabeto.
$$\text{si } L_1 \subset \Sigma^* \text{ y } L_1 \subset \Sigma^* \Rightarrow L_1 \cup L_2 \subset \Sigma^*$$

2. *Propiedad asociativa.* Sean $L_1,\ L_2,\ L_3$ subconjuntos de Σ^*
$$(L_1 \cup L_2) \cup L_3 \;=\; L_1 \cup (L_2 \cup L_3)$$

3. *Existencia del elemento neutro.* Cualquiera que sea el lenguaje L, el lenguage vacío cumple que:
$$L \cup \emptyset \;=\; \emptyset \cup L \;=\; L$$

4. *Propiedad conmutativa.* Cualesquiera que sean L_1 y L_2, se verifica que:
$$L_1 \cup L_2 \;=\; L_2 \cup L_1$$

5. *Propiedad de idempotencia.* Cualquiera que sea L, se verifica que:
$$L \cup L \;=\; L$$

Concatenación de lenguajes

Sean dos lenguajes no vacíos, definidos sobre el mismo alfabeto, $L_1 \subset \Sigma^*, L_2 \subset \Sigma^*$. Se conoce como concatenación o producto de los dos lenguajes $L_1,\ L_2$ al lenguaje definido como:

$$L_1 L_2 \;=\; \{uv \mid u \in L_1 \;\&\; v \in L_2\}$$

Es decir: todas las palabras del lenguaje producto, se forman concatenando una palabra del primer lenguaje con otra palabra del segundo lenguaje. Esta definición solo se cumple si ambos lenguajes son no vacíos, ya que

$$\emptyset L \;=\; L\emptyset \;=\; \emptyset$$

Propiedades de la operación Concatenación de lenguajes

1. *Operación cerrada.* La concatenación de dos lenguajes sobre el mismo alfabeto, es otro lenguaje sobre el mismo alfabeto.

2. *Propiedad asociativa.* Sean $L_1,\ L_2,\ L_3$ subconjuntos de Σ^*
$$(L_1 L_2) L_3 \;=\; L_1 (L_2 L_3)$$

3. *Existencia del elemento neutro.* Sea un lenguaje $L \subseteq \Sigma^*$, se cumple
$$\{\epsilon\} L \;=\; L\{\epsilon\} \;=\; L$$

Binoide libre

Con las operaciones *unión* y *concatenación* el conjunto $\boldsymbol{L}$ de todos los lenguajes L que pueden definirse sobre un alfabeto Σ es un Binoide. Si se considera a las letras del alfabeto como conjuntos de lenguajes de una sola palabra; con las operaciones de unión y concatenación de lenguajes puede generarse cualquier lenguaje sobre dicho alfabeto, con excepción de $\emptyset$ y ϵ.

Potencia de un lenguaje

Se conoce como potencia i-esima de un lenguaje a la operación que consiste en concatenarlo consigo mismo i veces. Como la concatenación tiene la propiedad asociativa, no es preciso especificar el orden en el cual tiene que efectuarse las i operaciones.

$$L^i = LLL \dots L \text{ i veces}$$

Si $L^1 = L$ y $L^0 = \{\epsilon\}$, se verifica que: $L^{1+1} = L^1 L = LL^1$ y $L^i L^j = L^{i+j}$ para $i, j \geq 0$.

Clausura positiva de un lenguaje

Se define como clausura positiva de un lenguaje L a:

$$L^+ = \bigcup_{i=1}^{\infty} L^i$$

Clausura de un lenguaje

Se define como clausura de un lenguaje L al lenguaje L^* que resulta de la unión de L^+ y L_0. Es de remarcar que $\{\epsilon\} \neq \emptyset$.

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

Esta clausura se conoce como clausura de Kleene

Prefijo de una palabra

$u \in \Sigma^*$ es prefijo de una palabra $v \in \Sigma^*$, si para algún $w \in \Sigma^*$, $v = uw$. Denotamos $Pre(v)$ al conjunto de todos los prefijos de v.

Clausura de prefijos de un lenguaje L

$\overline{L}$ es el lenguaje compuesto por todos los prefijos de L:

$$\overline{L} = \{u / uv \in L \text{ para algún } v \in \Sigma^*\}$$

Lenguaje prefijo cerrado

Un lenguaje es prefijo cerrado, si el lenguaje L y su clausura de prefijos coinciden: $L = \overline{L}$. Esto es, si $v \in L$ y u es prefijo de v, entonces $u \in L$.

Lenguajes S-cerrados

Sea $S \subset \Sigma^*$ un semigrupo con respecto a la concatenación, entonces L es S–cerrado si $LS \subset L$.

Símbolos admisibles

Sea $\alpha \in \Sigma$, diremos que α es un símbolo admisible para la palabra $u \in L$ si la concatenación $u\alpha \in L$.

Denotemos $L(u) \subset \Sigma$ al subconjunto de los símbolos admisibles para la palabra u.

2.2.3 Expresiones Regulares

El concepto de "expresiones regulare" fue introducido por Kleene en 1956, como un meta lenguaje para expresar el conjunto de palabras aceptadas por los autómatas de estado finito. Su definición es la siguiente:

Dado un alfabeto Σ y los símbolos $\emptyset$ (conjunto vacío), ϵ (palabra vacía), $+$ (unión), $\cdot$ (concatenación), $*$ (cierre o clausura), se cumple que:

1. El símbolo $\emptyset$ es una expresión regular.
2. El símbolo ϵ es una expresión regular.
3. Cualquier símbolo $\sigma \in \Sigma$ es una expresión regular.
4. Si α y β son dos expresiones regulares, $\alpha + \beta$ y $\alpha \cdot \beta$ también son expresiones regulares.
5. Si α es una expresión regular, α^* también es una expresión regular. α^* se define como:

$$\alpha^* = \bigcup_{i=0}^{\infty} \alpha^i$$

   donde α^i es la concatenación de α consigo misma i-veces, y $\alpha^0 = \epsilon$
6. Sólo son expresiones regulares las que se pueden obtener aplicando las reglas anteriors un número finito de veces sobre los símbolos de Σ, la palabra vacía ϵ y el conjunto vacío $\emptyset$.

El orden de prioridad de las operaciones es el siguiente:

1. $*$ (cierre)
2. $\cdot$ (concatenación)
3. $+$ (unión)

El orden puede alterarse mediante paréntesis.

2.3 Autómatas de Estado Finito

Existen diferentes definiciones sobre las máquinas de estado finito o autómatas de estado finito. Por ejemplo, Salomaa en [41] define una máquina de estado finito como el modelo matemático para un dispositivo que procesa información, produciendo respuestas a entradas discretas. De una manera más restrictiva, se define como un dispositivo que acepta un lenguaje de acuerdo a unas reglas específicas. Usaremos una definición que incluye tanto al reconocimiento del lenguaje como a la generación de símbolos de salida, cuya concatenación generan también un lenguaje.

2.3.1 Definición de Autómata de Estado Finito

Un autómata tiene un conjunto finito de estados interno donde la evolución del autómata se realiza mediante cambio en su estado, al recibir una entrada que sea aceptada en el estado presente. Estos cambios en el estado se conocen como transiciones. El conjunto de entradas aceptadas por el autómata forman el alfabeto de entrada. La diferente secuencia de entradas que hacen evolucionar al autómata se conoce como el lenguaje reconocido por el autómata [23].

Una máquina de estados finitos M es una 5-tupla

$$M = (Q, \Sigma, f_t, q_0, F)$$

donde

Q es un conjunto finito de estados denotados por q
Σ, alfabeto de entrada, es un conjunto finito de símbolos de entrada, denotados por σ
$f_t : Q \times \Sigma \mapsto 2^Q$ es la función de transición o evolución del autómata
$q_0 \in Q$ es el estado inicial del autómata
$F \subset Q$ es el conjunto de estados finales de M

Representación gráfica de un autómata

De manera gráfica, un autómata es representado mediante un grafo, donde los nodos representan los estados y los arcos representan las transiciones. Cada transición tiene asociada una entrada para activar su ocurrencia, aunque la misma puede ser nula. En el autómata de la figura 2.1 el conjunto de estados es $Q = \{a,\ b,\ c,\ d,\ e\}$, el conjunto de símbolos de entrada es $\Sigma = \{\alpha,\ \beta,\ \gamma,\ \delta\}$, el estado inicial es a y el final es e. El estado inicial de un autómata se indica mediante una flecha sin origen en otro estado, y los estados finales mediante un doble círculo

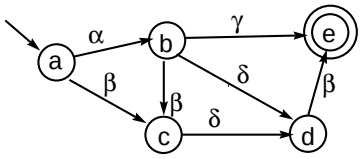


Figura 2.1: Autómata de Estado Finito

El lenguaje $L(M)$ reconcocido por el autómata M viene dado por el conjunto de las posibles concatenaciones de símbolos aceptadas por el autómata. $L(M) \subset \Sigma^*$, donde

$$\Sigma^* = \{\sigma_1, \sigma_2, \ldots \sigma_n, \sigma_1\sigma_2, \ldots, \sigma_1\sigma_n, \ldots \sigma_1 \ldots \sigma_n, \sigma_n\sigma_1 \ldots\}$$

En el ejemplo, el lenguaje reconocido viene dado por las cadenas $\alpha\gamma$, $\alpha\beta\delta\beta$, $\alpha\delta\beta$, y $\beta\delta\beta$.

Relación entre el lenguaje reconocido y el autómata

La relación entre un autómata y el lenguaje reconocido por el mismo viene dada por lo siguiente: Sea $(q, \omega) \in M$ un par que configura al autómata M, donde ω es una secuencia, (cadena de

símbolos de Σ^*), y las transiciones (cambio de estados relaciones binarias entre configuraciones) $f_t(q, \omega)$ donde $q \in Q$ y $\omega \in \Sigma^*$ entonces, se tiene que

$$f_t(q, \omega) \in Q$$

El conjunto $L(M) = \{\omega \in \Sigma^* \text{ tq } f_t(q_0, \omega) \in Q\}$ define el lenguaje aceptado por el autómata M.

Kleene en 1950 propone el teorema siguiente: Si un lenguaje es regular, entonces el puede ser aceptado por algún Autómata de Estado Finito; y si un lenguaje es aceptado por un Autómata de Estado Finito, entonces el lenguaje es regular.

Autómata determinístico

Se dice que un autómata es determinístico si $ft(q, \sigma)$ es única para cada $q \in Q$ y $\sigma \in \Sigma$, f_t, y además existe un solo estado inicial. Un autómata que no cumpla estas dos últimas propiedades se denomina un autómata no determinístico.

El autómata de la figura 2.2 es no determinístico, ya que para el estado b existen dos transiciones asociados al mismo símbolo.

$$\begin{array}{lll} f_t(a, \alpha) = \{b\} & & f_t(a, \beta) = \{c\} \\ f_t(b, \gamma) = \{c, d\} & & \\ f_t(c, \alpha) = \{d\} & & \end{array}$$

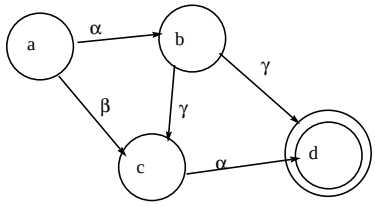


Figura 2.2: Autómata de Estado Finito no determinístico

Estado de bloqueo

Un autómata llega a un estado de bloqueo (estado sin salida) q_b, si q_b no es un estado final y si $\forall \sigma \in \Sigma,\ f_t(q_b, \sigma) = \emptyset$. En la figura 2.3, el estado c es un estado de bloqueo.

Autómata Reiniciable

Un autómata se dice que es *reiniciable*, si a partir de cualquier estado $q_i \in Q$ existe una secuencia ω tq. $f(q_i, \omega) = q_0$. El autómata mostrado en la figura 2.4.a es reiniciable, mientras que el autómata de la figura 2.4.b no lo es, ya que no existe ningún camino desde el estado e al estado a.

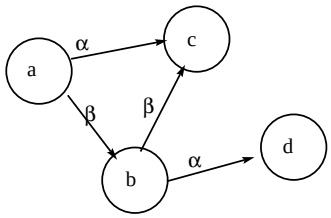


Figura 2.3: Autómata de Estado Finito con bloqueo

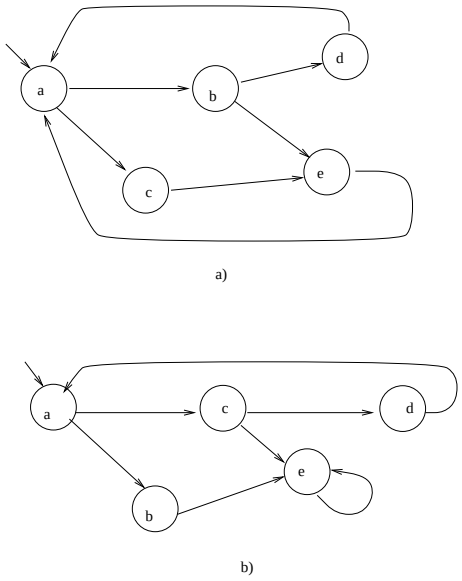


Figura 2.4: Reiniciabilidad en un autómata

Accesibilidad o Alcanzabilidad

Un estado $q_n \in Q$ se dice que es *accesible* si existe un ω tq. $f_t(q_0, \omega) = q_n$. En la figura 2.5 los estados $c,\ e,\ g$ no son accesibles ya que no existe ningún camino desde el estado inicial que lleve a uno de los estados $c,\ e,\ g$. Un autómata es accesible, si todos sus estados son accesibles. Un autómata puede transformarse en accesible eliminando los estados no accesibles. El lenguaje reconocido por el autómata accesible es el mismo.

Coaccesibilidad o Coalcanzabilidad

Un estado $q_n \in Q$ es *coaccesible*, si desde ese estado existe una secuencia de transciones que le permitan alcanzar un estado final. El automata es coaccesible si todos sus estados son coaccesibles. Un autómata puede transformarse en coaccesible si se eliminan los estados no coaccesibles.

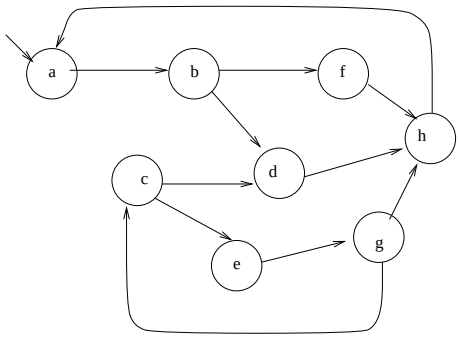


Figura 2.5: Accesibilidad en un autómata

Equivalencia de estados

Para un autómata finito determinista (Q, Σ, f, q_0, F), se dice que dos estados $p, q \in Q$ son equivalentes (pEq) si para toda palabra $\omega \in \Sigma^*$, se verifica que $f(p, \omega) \in F \Leftrightarrow f(q, \omega) \in F$. En la figura 2.6, el estado $s4$ es equivalente al estado $s5$.

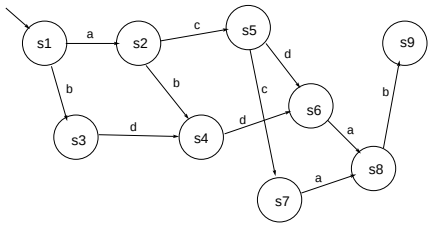


Figura 2.6: Equivalencia de Estados en un autómata

Autómatas equivalentes

Sean dos autómatas finitos determinísticos (Q, Σ, f_t, q_0, F) y $(Q', \Sigma, f'_t, q'_0, F')$. Se dice que dos estados $p \in Q$ y $q \in Q'$ son equivalentes si $f_t(p, \omega) \in F \Leftrightarrow f'_t(q, \omega) \in F'$, para todo $\omega \in \Sigma^*$.

Dos autómatas son equivalentes si reconocen el mismo lenguaje. Es decir, si $f(q_0, \omega) \in F \Leftrightarrow f'(q'_0, \omega) \in F'$, para todo $\omega \in \Sigma^*$. De esto deducimos que dos autómatas con diferente número de estados pueden ser equivalentes observacionalmente. Los dos autómatas descritos en la figura 2.7 son equivalentes, pues las dos secuencias reconocidas son idénticas.

2.3.2 Autómatas Productores de Lenguajes

Un autómata puede producir salidas asociadas a sus transiciones o a sus estados. Un autómata donde su salida es función de sus transiciones es conocido como una máquina Mealy, mientras que aquellos donde su salida es función del estado es conocido como una máquina de Moore. Un autómata generador de un lenguaje, donde los símbolos son generados en el momento de la

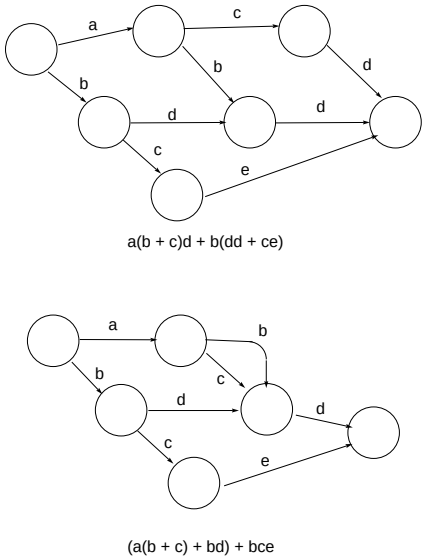


Figura 2.7: Equivalencia Observacional de autómatas

ocurrencia de sus transiciones, se define como una 7-tupla:

$$M = (Q, \Sigma, f_t, \Xi, f_s, q_0, F)$$

donde

Q es un conjunto finito de estados $q \in Q$
Σ o alfabeto de entrada es un conjunto finito de símbolos σ.
$f_t : Q \times \Sigma \mapsto Q$ es la función de transición o evolución
$\chi \in \Xi$ es el alfabeto de entrada $f_s : Q \times \Sigma \mapsto \Xi$ es la función de salida.
$q_0 \in Q$ es el estado inicial del autómata
$F \subset Q$ es el conjunto de estados finales de M

Gráficamente, en los autómatas productores reconocedores de lenguaje se representan los símbolos de salida mediante su nombre precedido por un signo de admiración **!** y los símbolos de entrada por el nombre del símbolo precedido por el signo de interrogación **?**

2.3.3 Generadores

Un generador es una quintupla $G = (Q, \Sigma, f_t, q_0, Q_F)$ donde $Q,\ \Sigma,\ , q_0,\ Q_F$ se definen de la misma manera que para un Autómata y f_t es una aplicación parcialmente definida $f_t : X \times \Sigma \mapsto X$. Un generador se denota como: $f_t(x, \sigma)!$

Lenguaje generado por G

El lenguaje generado por G; se define como el conjunto de todas las cadenas de eventos $u \in E^*$, E es el conjunto de eventos del generador G, admisibles a partir del estado inicial del generador:

$$L(G) = \{u \in E^* / f_t(x_0, u)!\}$$

El lenguaje marcado $L_m(G)$, corresponde a todas las cadenas de eventos admisibles, que llevan al sistema a un estado marcado.

$$L_m(G) = \{u \in E^* / f_t(x_0, u) \in F\}$$

Las propiedades que cumple un generador son:

1. $L(G)$ es prefijo-cerrado.
2. Un automáta A es un generador con un lenguaje $L(A) \subset \Sigma^*$
3. El lenguaje marcado de un generador satisface $L_m(G) \subseteq L(G)$
4. *Accesibilidad.* Un estado $x \in X$ es accesible si existe una cadena de eventos desde el estado inicial al estado x. Un generador es accesible si todos sus estados son accesibles, es decir si $\forall x,\ x = f_t(x_0, u)$, con $u \in \Sigma$.
5. *Co-accesibilidad.* Un generador es co-accesible (sin bloqueo) si para cualquier $u \in L(G)$ puede ser completado en una palabra que pertenezca a $L_m(G)$.
6. *Generador limpio.* Un generador es limpio si es accesible y co-accesible. Para que se cumpla ésto, el cierre del lenguaje marcado de G debe ser igual al lenguaje de G: $\bar{L}_m(G) = L(G)$

Operaciones con Generadores

Los generadores tienen un conjunto de operaciones como la proyección, proyección inversa y la composición.

- **Proyección**
  La proyección representa la generación de un nuevo lenguaje mediante el ocultamiento de un conjunto de eventos (símbolos). Sean Σ y Σ_i dos conjuntos de eventos, con $\Sigma_i \subset \Sigma$. La proyección $P_i : \Sigma^* \mapsto \Sigma_i^*$ se define como :

$$\begin{aligned} P_i(\epsilon) &= \epsilon \\ P_i(\sigma) &= \begin{cases} \epsilon \text{ si } \sigma / \in \Sigma_i \\ \sigma \text{ si } \sigma \in \Sigma_i \end{cases} \\ P_i(u\sigma) &= P_i(u)P_i(\sigma), \quad \text{para } u \in \Sigma^*; \sigma \in \Sigma \end{aligned}$$

  Para un lenguaje L la proyección del lenguaje corresponde a borrar todos los símbolos que no estén presente en el conjunto de proyección.

$$P_i L = L_i = \{u_i \in \Sigma^* / u_i = P_i u \text{ para cualquier } u \in L\}$$

- **Proyección inversa**
  La proyección inversa corresponde a la generación de todas las posibles palabras de las cuales se puede generar una palabra que ha sido proyectada.

$$P_i^{-1} L_i = \{u \in \Sigma^* / P_i u \in L_i\}$$

- **Producto Síncrono**
  En la composición síncrona, las transiciones en dos generadores ocurren solo si ambos generadores aceptan desde el estado actual el mismo evento común a ambos lenguajes. Formalmente: Sean $L_1 \subseteq \Sigma_1^*$ y $L_2 \subseteq \Sigma_2^*$, (con la posibilidad $\Sigma_1 \cap \Sigma_2 = \emptyset$). Sea $\Sigma = \Sigma_1 \cup \Sigma_2$. El producto síncrono $L_1//_sL_2 \subseteq \Sigma^*$ se define como :

  $$L_1//_sL_2 = P_1^{-1}L_1 \cap P_2^{-1}L_2$$

  Se nota que $u \in L_1//_sL_2$ si y sólo si $P_1(u) \in L_1$ y $P_2(u) \in L_2$

  Para los autómatas A con $\Sigma_a = \{\alpha, \beta\}$ y B con $\Sigma_b = \{\beta, \gamma\}$ de la figura 2.8, el producto síncrono genera el autómata C de la misma figura. $C = A//_sB$

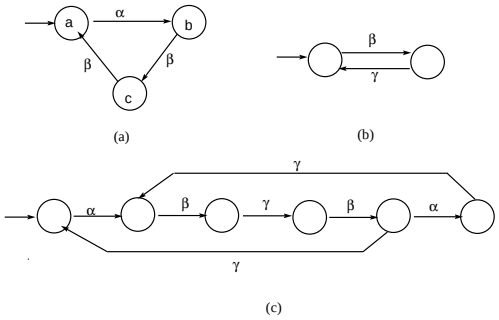


Figura 2.8: Producto síncrono de dos lenguajes

Es de notar que el producto de dos generadores limpios puede no ser limpio.

- **Intersección de lenguajes**
  La intersección de lenguajes, corresponde al producto síncrono de los generadores.

- **Lenguajes sin conflicto**
  Dos lenguajes se dicen que no entran en conflicto si L_1 no contiene ninguna trayectoria completa que no esté en L_2 y viceversa. Para ello, ambos lenguajes deben ser prefijo cerrado.

  Formalmente: Dados los lenguajes L_1 y $L_2 \subseteq \Sigma^*$ se tiene que:

  $$\overline{L_1 \cap L_2} = \bar{L}_1 \cap \bar{L}_2$$

- **Producto asíncrono**
  El producto asíncrono de dos generadores G_1 y G_2, representado $G_1||G_2$ resulta de todas las combinaciones posibles de ambos generadores. El producto asíncrono resulta de todas las transiciones posibles independientes en ambos generadores y el resultado es el producto cartesiano de los mismos.

2.4 Autómata de Estados Finitos Comunicantes

Dentro de los desarrollos de sistemas basados en autómatas, un caso especial es el orientado al desarrollo de sistemas comunicantes; por lo cual este punto es tratado en este capítulo. Un autómata

comunicante,es un autómata que reconoce un lenguaje, y que a su vez produce un lenguaje en su evolución, estableciéndose un coloquio (intercambios de mensajes) entre dos entidades. Como se ha dicho al comienzo, la evolución del autómata está determinada por la entrada que él recibe, pero si los mensajes producidos desde el autómata inicial alteran la secuencia de los mensaje que él va a recibir, ésto implica que su evolución es producto del intercambio de símbolos entre él mismo y su contexto. El contexto puede ser otro autómatas o un grupo de autómatas. Los símbolos intercambiados son conocidos como "mensajes". El sentido del término mensaje viene dado por que mediante el intercambios de símbolos, se está transmitiendo y recibiendo información desde/hacia el autómata. En la figura 2.9 se indica un conjunto de autómatas que intercambian mensajes a través de unos puertos con el fin de cumplir un servicio; en el ejemplo, los autómatas 1 y 2 utilizan los servicios suministrados por el par de autómatas 3 y 4. En la figura, los símbolos intercambiados se muestran junto a las líneas de salida de cada una de las máquinas; las máquinas tienen uno o más puertos para el intercambio de mensajes.

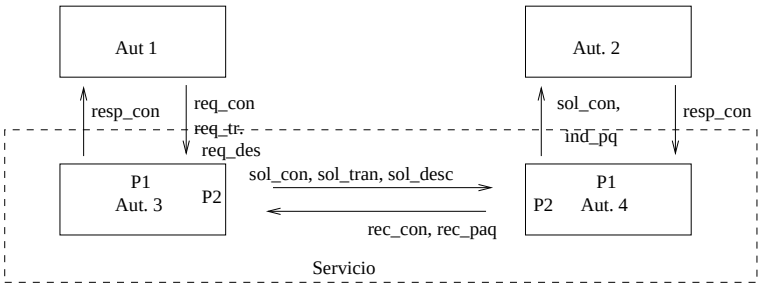


Figura 2.9: Máquinas comunicantes

Cada máquina tiene una evolución descrita mediante un autómata de estado finito. La evolución de cada una de las máquinas se muestra en la figura 2.10

El intercambio de mensajes trae como resultado un nuevo autómata formado por el producto de los autómatas que están intercambiando información. El tamaño del espacio de estados del nuevo autómata crece y es menor o igual al producto del número de estados de los autómatas que se están comunicando, ya que $Q_p \subseteq Q_1 \times Q_2$. En el ejemplo de la figura 2.9, los autómatas intercambian mensajes entre ellos para cumplir un servicio, generando un nuevo autómata que se muestra en la figura 2.11. El nuevo autómata puede ser sustituido por un autómata de menor tamaño que sea equivalente observacionalmente para un ente externo.

2.4.1 Composición de autómatas comunicantes

Un autómata puede poseer uno o más puertos de comunicación por medio de los cuales mantiene diálogos con diferentes autómatas o generadores de lenguaje. El diálogo mantenido con cada uno de ellos es función, no solamente del intercambio del otro autómata con el cual se comunica, sino también del diálogo mantenido con otros autómatas por los otros puertos. El autómata generado por la interacción entre dos autómatas es visto como un autómata único por cada uno de ellos tal como se definió en la subsección 2.4.

La composición puede ser *asíncrona* como en el caso de los autómatas comunicantes o *síncrona* cuando los mensajes emitidos por un autómata son captados por otro autómata en el mismo momento, esto es, las transiciones ocurren simultaneamente. En la composición asíncrona existe un tiempo entre una transición ocurrida en el autómata que genera el evento y el autómata que

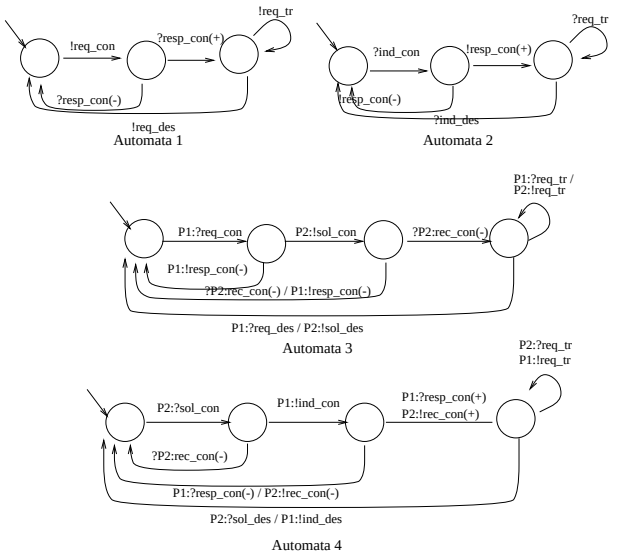


Figura 2.10: Máquinas comunicantes

consume el evento (mismo símbolo reconocido por ambos autómatas). En la figura 2.12 se muestra la composición asíncrona de dos autómatas, que es equivalente al caso del autómata generado por intercambio de información.

En la composición síncrona, existe una fusión de transiciones, ésto es, para que una transición asociada a un evento se produzca en el autómata *A1*, se debe producir la transición en el autómata *A2* asociada con el mismo evento. En la figura 2.13 se muestra la evolución síncrona los autómatas de la figura anterior. Cuando el evento está asociado a transiciones en un solo autómata, esta transición se produce como en el caso asíncrono.

El autómata resultante de esta composición, puede a su vez componerse con otros autómatas y establecer de esta manera una jerarquía. Su composición sigue exactamente las mismas reglas descritas anteriormente.

2.4.2 Máquinas de Estado Finito Extendidas. (MEF-E)

Asociemos al ejemplo mostrado en el capítulo 1.4 un estado de funcionamiento de los ventiladores de refrigeración. En este caso el estado puede ser ninguno encendido, uno encendido, o los dos encendidos; si como criterio adicional, tenemos que el próximo a encender debe ser el que tiene más tiempo apagado, el número de estados se incrementa, y el autómata se muestra en la figura 2.14.

Para simplificar el modelado y la descripción del sistema, se pueden asociar variables al autómata. En este caso, el autómata se conoce como una máquina de estado finito extendida (MEFE). Una MEFE es un autoómata que tiene un conjunto de variables asociadas que ayudan a caracterizar un estado. Los valores de las variables en el autómata dan información suplementaria al estado. La máquina de estado finito extendido que representa el autómata de la figura 2.14 se muestra en la figura 2.15.

Formalmente, una MEFE está dada por la octupla:

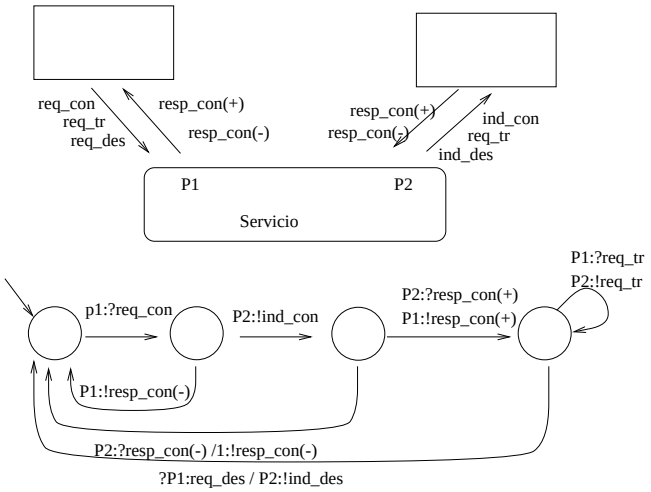


Figura 2.11: Autómata generado por el intercambio de mensajes entre dos autómatas

$$MEFE \ = \ (Q, \Sigma, V, \Xi, f_t, f_s, q_0, v_0, F)$$

donde Q es el conjunto de macro-estados del autómata,
Σ es el conjunto de símbolos de entrada,
V es el conjunto de variables asociadas,
Σ es el conjunto de símbolos de salida,
f_t esta dada por f_{t1} y f_{t2}, donde una transforma las variables y la otra los macroestados.
$f_{t1} : Q \times V \times \Sigma \mapsto Q$ y
$f_{t2} : Q \times V \times \Sigma \mapsto V$,
$f_s : Q \times V \times \Sigma \mapsto \Sigma$,
q_0 es el macro-estado inicial y v_0 es el vector inicial de valores.

2.4.3 Lenguajes de descripción de autómatas

Para la descripción de sistemas de transiciones basados en autómatas de estado finito extendidos, varios lenguajes han sido propuestos en el mercado. EL VDHL, orientado al modelado de sistemas discretos, en especial componentes en microelectrónica, el SDL, desarrollado para la especificación y validación de protocolos de comunicación. Posteriormente aparecen otros lenguajes orientados a la implementación de sistemas como el Esterel y el Lustre.

2.5 Redes de Petri

En la década de los 60 Karl von Petri escribe su trabajo “Kommunication mit Automaten”, donde describe una técnica para representar máquinas que se comunican, de una manera más completa que mediante MEF. Las MEF presentan dificultad al momento de describir “estados” que resultan de la combinación de varios predicados. Este trabajo tuvo un gran impacto en la computación,

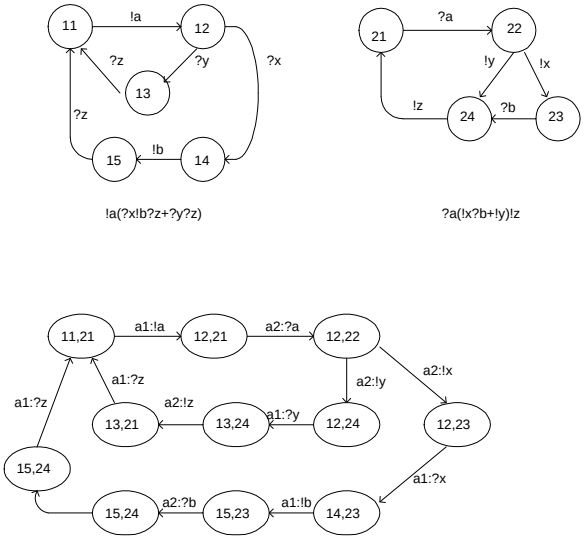


Figura 2.12: Composición asíncrona de Autómatas Comunicantes

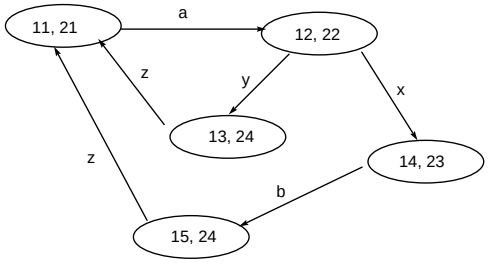


Figura 2.13: Composición Síncrona de Autómatas Comunicantes

siendo usado en el año 1969 en Carnagie Melon en el diseño de un nuevo computador; luego en el control de sistemas procedimentales resultó un herramienta muy útil y de ella se deriva el *GRAFCET* ver [15] el cual es utilizado en algunos *Controladores Lógicos Programables* como lenguaje de programación del controlador. Las Redes de Petri han sido ampliamente utilizadas en la descripción, construcción y validación de protocolos de comunicación. Finalmente, las Redes de Petri también se utilizan en la descripción de problemas industriales, ya que permiten realizar la abstracción de actividades complejas en un conjunto transición lugar transición [50].

Una red de Petri es un 6-tupla $RdP = (L, T, \Sigma, M, f_e, M_0)$, donde:

L es el conjunto de Nodos Lugares
T Es el Conjunto de Nodos Transición
Σ es un conjunto de eventos o condiciones de disparo asociados a una transición
M es un vector de marcación
$f_e : L \times T \times M \mapsto M$ es la función de evolución de la red
M_0 es la marcación inicial.

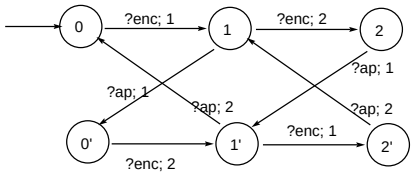


Figura 2.14: Autómata con memoria del último encendido

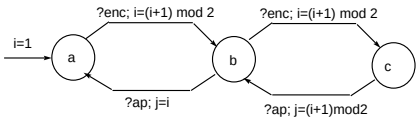


Figura 2.15: Autómata extendido con memoria del último encendido

Representación gráfica de una Red de Petri

Los principios en los cuales se basan las Redes de Petri son bastantes sencillos. De manera gráfica, una Red de Petri es un grafo bi-partito, con dos tipos de nodos: Los nodos "lugares" y los nodos "transición", con arcos que van desde los nodos lugares a los nodos transición y viceversa. No existen arcos que vayan de un nodo transición a otro nodo transición o de un nodo lugar a otro nodo lugar (son nodos bi-partitos). La evolución se logra mediante la actualización de las fichas en los lugares luego del disparo de una transición.

- **Nodos Lugar**
  Los nodos lugar, dependiendo de la orientación del sistema representados gráficamente mediante un círculo, están asociados al estado, o a los recursos disponibles en un sistema. Los nodos lugar tienen asociadas fichas que permiten determinar el estado de una red.

- **Nodos Transición**
  Los nodos transición representan el cambio del estado en la red, mediante una modificación del número de fichas en los nodos lugar. Los nodos transición tienen asociados eventos, que hacen que se realice el disparo de la transición si la misma se encuentra habilitada.

En la figura 2.16 se describe una Red de Petri.
Para la red mostrada en la figura 2.16 la marcación inicial es

$$M_0 = (1, 0, 0, 0, 0, 0, 0, 0, 0)$$

Las condiciones de disparo para la transición $e1$ está dada por la marcación inicial. El disparo de la transición $e1$ conduce a la marcación

$$M = (0, 1, 0, 0, 0, 0, 0, 0, 0)$$

Desde esta marcación, solo puede ocurrir la transición $e2$, que conduce a la marcación

$$M = (0, 0, 1, 0, 0, 0, 0, 0, 0)$$

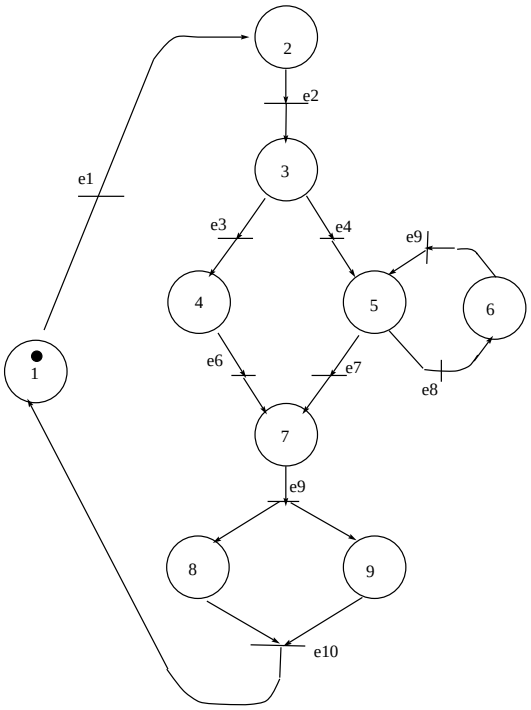


Figura 2.16: Representación gráfica de una Red de Petri

Desde esta marcación, dos transiciones pueden ocurrir, $e3$ y $e4$. Llevando a las marcaciones:

$$M = (0, 0, 0, 1, 0, 0, 0, 0, 0) \quad M = (0, 0, 0, 0, 1, 0, 0, 0, 0)$$

El disparo de la transición $e9$ lleva a la marcación

$$M = (0, 0, 0, 0, 0, 0, 0, 1, 1)$$

donde se puede disparar la transicion $e10$.

2.6 Verificación de Sistemas de Transiciones

La dinámica de un sistema de transiciones debe satisfacer con un conjunto de propiedades como: accesibilidad, coaccesibilidad, limpieza, reiniciabilidad, etc, que permiten determinar la validez de un diseño o un sistema en funcionamiento. Esta verificación parece simple en el caso de sistemas pequeños, y se puede lograr mediante una análisis del sistema gráfico, se vuelve inmanejable cuando se realiza la composición de sistemas. La composición de sistemas de transiciones como dos autómatas o dos redes de Petri, nos generan un nuevo sistema de transiciones resultado del intercambio de mensajes, (como en el caso de autómatas comunicantes y Redes de Petri) o del producto de lenguajes (como el caso de los Generadores).

Es necesario entonces disponer de herramientas que permitan la verificación de propiedades a partir de su descripción mediante lenguajes de computación, bases de datos, u otra herramienta de descripción de sistemas de transiciones. Las herramientas computacionales para la verificación de propiedades en los sistemas discretos son pocas y no existen herramientas de uso general que permitan verificar aplicaciones de protocolos de comunicaciones y aplicaciones de control y supervisión de sistemas discretos.

Herramientas como TCTW de la Universidad de Otawa están orientadas a la verificación de las especificaciones de supervisión y control, mientras que la mayoría de las herramientas fueron orientadas a la verificación de protocolos de comunicación como el VASID [11], desarrollado en la Universidad de Los Andes para la verificación de Sistemas Distribuidos. Algunas aplicaciones del mismo se dan en [14].

Capítulo 3

Modelado y Control de SED's

3.1 Introducción

En el capítulo anterior hemos visto como se describe un sistema discreto, y como su evolución es determinada por la presencia de eventos en ciertos estados. El modelado consiste entonces en extraer el comportamiento de sistemas procedimentales expresados en términos de MEF, Redes de Petri o generadores de lenguajes, para los cuales se pueda determinar la controlabilidad del sistema y posteriormente construir un supervisor o un controlador del sistema.

Como premisa de modelado en los Sistemas de Eventos Discretos (SED's), se tiene que la existencia de un evento en un estado del SED, implica la evolución del mismo; entonces, la dinámica del sistema está determinada por la secuencia de eventos a partir de un estado inicial. Si se logra controlar la presencia de los eventos para un sistema discreto entonces el mismo puede ser controlado mediante la imposición de ciertos eventos por parte de un **supervisor**, tal como lo describen los trabajos iniciales de Wonham y Ramadge [54, 38, 37] dedicados a la supervisión de sistemas discretos. Estos trabajos posteriormente han sido utilizados por diferentes autores, para el desarrollo de supervisores de sistemas de eventos discretos [8, 7, 49, 52] o en la síntesis de controles procedimentales para procesos [42]. Desarrollos paralelos aparecen orientados a la construcción de programas paralelos – concurrentes en tiempo real [3, 10, 33].

Hasta el momento se han desarrollado varios modelos para SED, en la tabla 3.1 se resumen los principales:

Modelos	**Temporizados**	**No-temporizados**
Lógicos	Ramadge-Wonham temporizados Redes de Petri temporizadas Lógica temporal de tiempo real (GRTTL)	Ramadge-Wonham Redes de Petri controladas Lógica temporal
Algebraicos	Álgebra Max-plus	Álgebra de procesos
Analíticos de rendimiento	Cadenas de Markov Teoría de colas Modelos estocásticos temporizados	

Tabla 3.1: Tabla comparitiva de modelos de SED.

Para el desarrollo de Sistemas a Eventos Discretos y su control, usaremos primordialmente las definiciones extraidas de los trabajos de Ramadge – Wonham y Sanchez. En este capítulo daremos una introducción para la comprensión del desarrollo de supervisores y controladores de eventos.

3.2 Modelado de SDED's

Formalmente, un Sistema Dinámico a Eventos Discretos (SDED's), es una 5-tupla

$$SDED = [Q, \Sigma, f_t, q_0, Q_m]$$

donde:

- Q es el conjunto de estados
- Σ es el conjunto de eventos, *controlables* y *no controlables*. $\Sigma = \Sigma_c \cup \Sigma_{nc}$.
- f_t es la función de transición parcialmente definida. $f_t : \Sigma \times Q \mapsto Q$.
- q_0 es el estado inicial.
- Q_m el conjunto de estados marcados. $Q_m \subseteq Q$.

El aspecto primordial a ser considerado en el control de sistemas discretos viene dado por el manejo de los eventos. Un evento se considera *controlable* si el mismo puede ser deshabilitado por un supervisor, con el fin de que no aparezca en el sistema, mientras que los eventos *no controlables (incontrolables)*, aparecen espontáneamente en el sistema discreto y la única forma de evitar de que éstos aparezcan es el de mantener el sistema en estados donde los mismos no pueden presentarse.

Los modelos construidos mediante MEF deben permitir representar la dinámica secuencial del sistema (*comportamiento*) en los aspectos de :

- El comportamiento del sistema, que representa todas las posibles trayectorias que se pueden generar a partir de un estado inicial para diferentes secuencias de eventos.
- El comportamiento marcado, que representa las trayectorias posibles desde un estado inicial a un estado deseado. Estos estados deseados pueden representar el cumplimiento de una tarea, un estado de operación, la capacidad de reinicialización del sistema, etc.

Este especificación es conocida como el *Comportamiento en lazo abierto.*

El lenguaje generado por la planta puede ser alterado por la presencia de un supervisor que restrinja la presencia de ciertos eventos en algunos estados; ésto significa la eliminación de palabras del lenguaje inicial.

El segundo paso consiste en definir la especificación del comportamiento deseado bajo la existencia de un supervisor que permita la deshabilitación de los eventos controlables, esta especificación es conocida como el *Comportamiento en lazo cerrado.* Por lo tanto, el supervisor debe conocer todos los símbolos que aparecen en la planta para poder seguir su comportamiento y de esta manera deshabilitar símbolos controlables que puedan permitir la existencia de caminos que lleven a situaciones no deseadas.

El último paso lleva a la síntesis del supervisor, ésto es la construcción de un supervisor que secuencie las trayectorias deseadas en la planta a partir del uso de los eventos controlables.

El comportamiento en lazo cerrado debe cumplir con las especificaciones esperadas en un sistema discreto, ésto es, ausencia de situaciones de bloqueo, ausencia de lazos infinitos, y asegurar que el sistema alcance los estados deseados en un número finito de transiciones.

Para encontrar el modelo que representa un sistema, se procede por representar el conjunto posible de transciones (y eventos asociados) que pueden aparecer en el sistema. En el caso de sistemas complejos, el modelo se logra mediante la composición de modelos elementales según muestra Sanchez en el capítulo 2 de [42].

Tomemos como ejemplo el caso del sistema mostrado en la figura 3.1, donde los elemntos componentes del sistema son: un tanque de almacenamiento con sus dos elementos sensores de nivel máximo y nivel mínimo; una válvula de entrada, una bomba de entrada, una válvula de salida y una bomba de salida. Este sistema forma parte de un sistema más completo que será analizado más adelante.

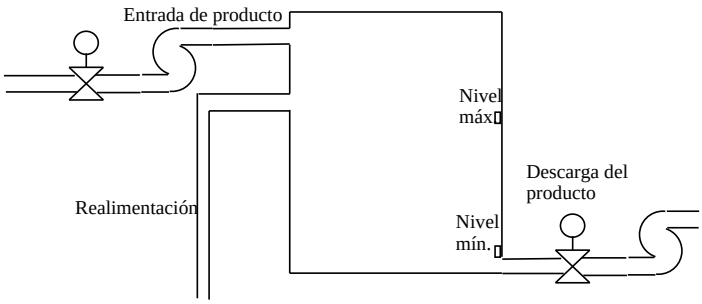


Figura 3.1: Tanque de carga y recirculación de producto

El tanque recibe un producto para ser tratado más adelante en un proceso aguas abajo, el cual debe repetirse hasta que el producto esté listo. El funcionamiento del tanque de almacenamiento es el siguiente:

- **Carga del producto**. Por medio del primer grupo de válvula – bomba, se introduce el producto sin que sobrepase el nivel máximo.

- **Recirculación del producto y descarga del producto**. Una vez introducido el producto, el grupo de carga es apagado y se procede con el grupo de salida del producto. Este grupo funcionará hasta que el producto haya sido descargado totalmente. Esto se indica mediante el sensor de nivel mínimo. El proceso que está aguas abajo, hará recircular el producto al tanque, mientras éste no esté limpio.

Modelado de los sistemas elementales

Válvula.
La válvula se modela mediante una MEF, la cual tiene dos estados **cerrada (0)** y **abierta (1)**. Las transiciones posibles son la de **abrir** y **cerrar**. El estado inicial es siempre cerrada, y su estado marcado es cerrada. La evolución de la misma se da en la figura 3.2(a). La evolución de la bomba se da de una manera similar, teniendo los estados **apagada (0)** y **encendida (1)**

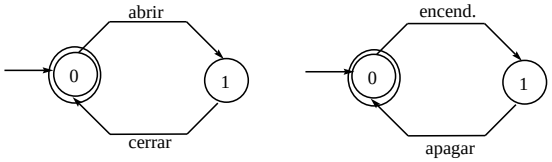


Figura 3.2: Modelos de válvula y bomba

La composición asíncrona de ambos elementos da la MEF se muestra en la figura 3.3(a). Existe un estado indeseable en esa composición, que esta dada por la condición de válvula cerrada y bomba encendida; este estado debe ser evitado. La evolución válida se da en la figura 3.3(b)

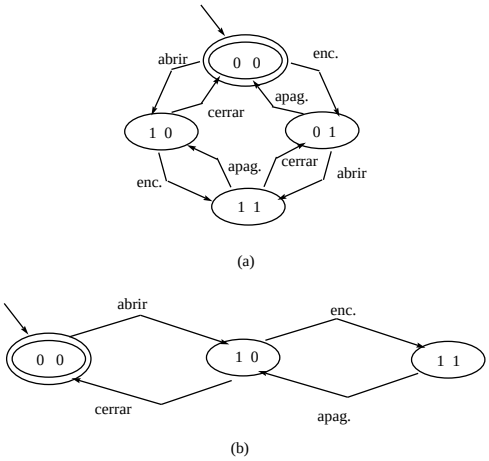


Figura 3.3: Composición de los modelos de válvula y bomba

En el caso de la válvula y la bomba, los eventos de *encender, apagar, abrir cerrar* son todos controlables, por lo cual el estado *no deseado (1, 1)* del grupo válvula bomba puede ser evitado si se da la secuencia correcta. El lenguaje reconocido por el grupo es:

$$(((ab{\circ}ce)^*+(en{\circ}ap)^*)^*{\circ}(ab+en+(ab{\circ}en+en{\circ}ab){\circ}(ce+ap+\ ((ce{\circ}ab)^*+(ap{\circ}en)^*)^*{\circ}(ce{\circ}ap+ap{\circ}ce))))^*$$

El lenguaje que nos lleva a la situación indeseada es:

$$((ab \circ ce)^* \circ (ab + ab \circ en \circ (ap \circ en)^* \circ ap + ap \circ ce))^*$$

El lenguaje supervisado es un subconjunto del lenguaje reconocido por la composición inicial de las dos máquinas.

El modelado del tanque se logra mediante la representación del nivel del tanque. Este tanque presenta tres niveles posibles que son: *vacío (0), no vacío – no lleno (1), lleno (2).* El estado

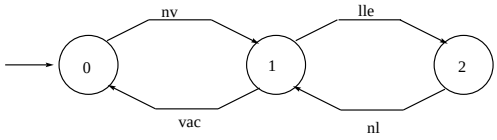


Figura 3.4: Modelo de funcionamiento del tanque en MEF

vacío corresponde al estado inicial. Los sensores tienen dos salidas posibles, 0 y 1. La evolución del tanque se da en la figura 3.4.

El conjunto de símbolos para el tanque son $\{nv,\ vac,\ lle,\ nl\}$. Ninguno de estos eventos es controlable, pues son generados por los sensores de la planta.

En la figura 3.5 se muestra la composición asíncrona de las bombas con el tanque para reducir el número de estados. El autómata que contiene los grupos bomba - válvula y tanque genera un autómata de 27 estados y 108 transiciones. El gráfico mostrado tiene 12 estados y 40 transiciones.

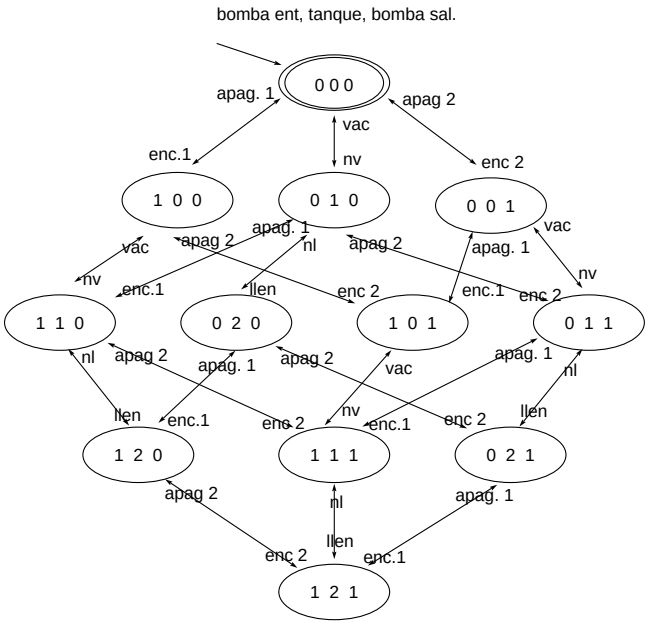


Figura 3.5: Modelo de funcionamiento del grupo bomba de entrada – tanque – bomba de salida

3.2.1 Eliminación de transiciones no posibles físicamente

La evolución del tanque está condicionada al funcionamiento de los grupos de bomba-válvula. El tanque no puede cambiar de estado si las bombas no están funcionando y las válvulas no están abiertas. Entonces es necesario eliminar del modelo compuesto las transiciones no factibles. Esto

se puede realizar mediante un estudio del autómata o por composición síncrona con un autómata que contenga las secuencias posibles.

El cambio en el tanque del estado 0 al estado 1, (transición nv) sólo puede ocurrir cuando la bomba de entrada ha sido encendida, al igual que la transición que cambia el estado 1 al estado 2 (transición lle). De la misma manera, el tanque cambia del estado 2 al estado 1 (transición nl) y al estado 0 (transición vac) cuando la bomba de salida está encendida.

La eliminación se logra mediante un análisis de las transiciones posibles o mediante la composición síncrona del autómata resultante de la composición asíncrona de los componentes elementales. La composición síncrona del autómata mostrado en la figura 3.5 con el autómata 3.6.a que elimina la posibilidad de que se den transiciones de llenado antes del funcionamiento de la bomba, nos da un autómata con 12 estados y 36 transiciones. Una nueva composición con el autómata 3.6.b que elimina las transiciones de vaciado cuando la bomba de salida no está en funcionamiento nos da un autómata de 12 estados pero con 32 transiciones.

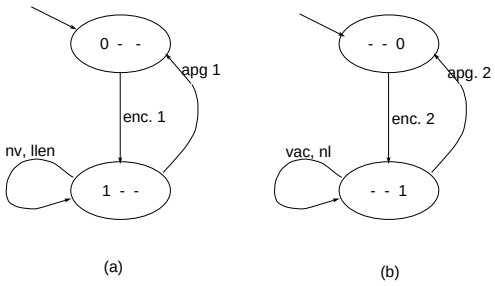


Figura 3.6: Modelado de transiciones posibles

El autómata que representa la evolución eliminando las transiciones no posibles se da en la figura 3.7.

Una vez obtenida el modelo de la planta a controlar, el siguiente paso consiste en la construcción de un supervisor según el enfoque de Ramadge, Wonham [38, 37, 54] o conseguir un controlador según Sanchez [42] que permitan asegurar que el sistema llegue a los estados deseados o que cumpla con unas secuencias que satisfagan los requerimientos impuestos por el diseñador.

3.3 Supervisores y Estructura de Control

El objetivo de control es el de definir un conjunto de secuencias de símbolos que eviten que la planta llegue a una situación indeseada, o lo que es mejor, que estas secuencias garanticen conducir al sistema a un estado deseado (marcado).

Partiendo de la partición del conjunto de símbolos Σ en dos subconjuntos Σ_c y Σ_{nc} que cumplen

$$\Sigma = \Sigma_c \cup \Sigma_{nc}$$

$$\Sigma_c \cap \Sigma_{nc} = \emptyset$$

se debe encontrar un conjunto $\gamma \subset \Sigma$ tal que:

- Si $\sigma \in \gamma$ entonces σ es permitido por γ, de otra manera σ es deshabilitado por γ.

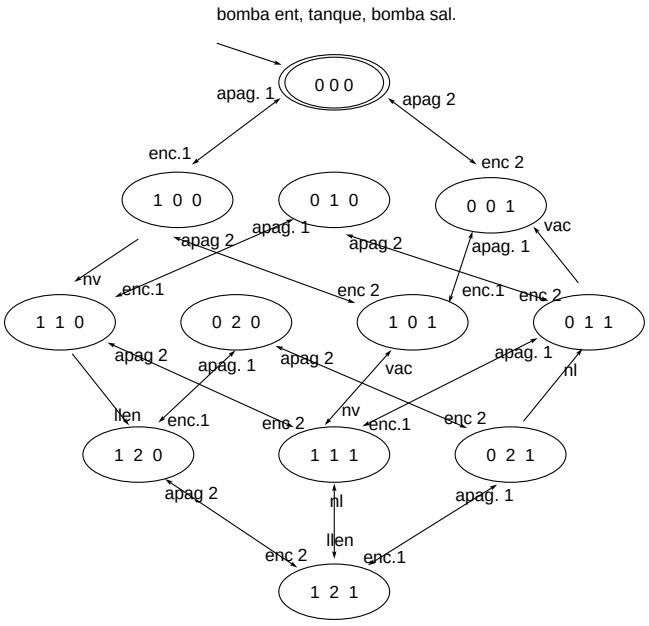


Figura 3.7: Modelo de funcionamiento posible del grupo bomba de entrada – tanque – bomba de salida

- Si $\sigma \in \Sigma_{nc}$ entonces $\sigma \in \gamma$

  El conjunto de las posibles entradas de control es $\Gamma \subset 2^{\Sigma}$.

Para el autómata mostrado en la figura 3.8, con $\Sigma = \Sigma_c \cup \Sigma_{nc}$; $\Sigma_c = \{\alpha, \beta\}$ y $\Sigma_{nc} = \{\xi, \omega\}$; $Q = \{a, b, c\}$; $q_0 = a$ y $Q_m = a$

$\Gamma = \{\gamma_1, \gamma_2, \gamma_3, \gamma_4\}$
$\gamma_1 = \{\alpha, \beta, \xi, \omega\}$
$\gamma_2 = \{\alpha, \xi, \omega\}$
$\gamma_3 = \{\beta, \xi, \omega\}$
$\gamma_4 = \{\xi, \omega\}$

3.3.1 Lenguaje controlado

Si las secuencias posibles de la planta pueden ser seleccionadas dentro del conjunto Γ mediante un supervisor, se tiene un lenguaje que puede ser controlado. La selección se realiza en función de los eventos que se han ido produciendo. El control es una aplicación

$$h : L \mapsto \Gamma$$

la cual asocia a cada cadena posible $\Omega \in L$ una secuencia de control $\gamma = h(\omega) \in \Gamma$.

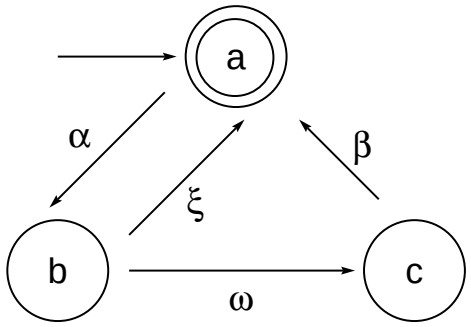


Figura 3.8: Máquina de tres estados

3.3.2 Supervisores de SED

El supervisor es un sistema discreto que sigue a la planta, mediante un reconocimiento de los símbolos que aparecen en la planta (eventos) y habilita dentro de los eventos controlables, aquellos que permitan conducir al sistema hacia un estado deseado. El sistema supervisado presenta una realimentación de control, al igual que en el caso clásico de los sistemas continuos bajo control.

El supervisor es el sistema:

$$S = (S_{svr}, \Psi)$$

donde S_{svr} es una máquina de estados finita dada por la 5-tupla:

$$S_{svr} = [X, \Sigma, f_e, x_0, X_m]$$

donde:

- X es el conjunto de estados del supervisor.
- Σ es el mismo conjunto de eventos del sistema supervisado.
- f_t que es la función de transición definida parcialmente. $f_e : \Sigma \times X \mapsto X$.
- x_0 es el estado inicial.
- X_m el conjunto de estados marcados. $Q_m \subseteq Q$.
- Ψ es una aplicación de deshabilitación definida como:

$$\Psi(\sigma, x) = \begin{cases} 0 \text{ o } 1 & \text{si}\sigma \in \Sigma_c \\ 1 & \text{si } \sigma \in \Sigma_{nc} \end{cases}$$

La función Ψ representa el mecanismo de deshabilitación y tiene el sentido de dejar pasar el evento σ en el estado x del supervisor hacia la planta. $\Psi(\sigma, x)$ siempre será igual a 1 si σ es no controlable y 0 o 1 en el caso de que σ sea controlable.

En un enfoque, la planta P y el supervisor S están sincronizados sobre el conjunto de eventos compartidos por ambos. Los estados de S son resultado de patrones secuencias conocidas por el

supervisor, el cual habilita eventos que hacen que la planta siga uno de los patrones que tiene el supervisor. Esto hace que el supervisor no genera comandos de control, por el contrario el inhibe la aparición de eventos controlables en la planta. La decisión final del disparo de la transción corresponde a la planta. En la figura 3.9 se muestra la estructura de la planta y su supervisor, de acuerdo a la noción de control realimentado, sabiendo que $\Sigma_c = \{\alpha, \delta\}$ y $\Sigma_{nc} = \{\beta, \gamma, \theta\}$

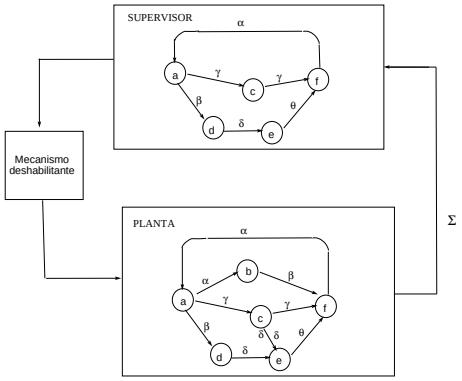


Figura 3.9: Sistema Dinámico a Eventos Discretos supervisado

La acción conjunta de la planta y el supervisor genera un un subconjunto de trayectorias dentro de las trayectorias iniciales del sistema, identificado como el lenguaje de lazo cerrado. Este lenguaje se identifica como $L(S/P)$: Lenguaje de P restringido por S. De la misma manera el lenguaje marcado de lazo cerrado se identifica como $L_m(S/P)$. Estos lenguajes cumplen con:

$$L(S/P) = L(S) \cap L(P) \text{ y}$$

$$L_m(S/P) = L_m(S) \cap L_m(P)$$

Para el comportamiento en lazo cerrado se tienen las siguientes condiciones:

- $\epsilon \in L(S/P)$
- $\sigma\sigma \in L(S/P)$ si y sólo si $\sigma \in L(S/P),\ \sigma\sigma \in L(P),\ \Psi(\sigma, f_t(x_0, \sigma) = 1$

El lenguaje en lazo cerrado tiene las propiedades siguientes:

- $\{\epsilon\} \subseteq L(S/P) \subseteq L(P)$
- $L(S/P)$ no es vacío y es prefijo cerrado.

3.3.3 Realización de un supervisor mediante un SDED

Un supervisor puede ser representado mediante un generador S que tiene acciones de control sobre P implícito en su estructura de transiciones. de la siguiente manera:

- Si $\omega \in L(h/P)$ entonces
  $\omega \in L(S)$,
  $\omega\sigma \in L(S)$ si y sólo si $\sigma \in h(\omega)$

- Si $\omega \in L(h/P),\ \omega\sigma \in L(P),\ \sigma \in h(\omega)$ entonces $\omega\sigma \in L(S)$

Esto quiere decir que las transiciones deshabilitadas no deben aparecer en la estructura de transciones del supervisor, y que las transciones habilitadas y, físicamente posibles, deben estar en la estructura del supervisor.

Para el generador $S = (Y, \Sigma, f_g, y_0, Y)$

- las transiciones son disparadas por los eventos que ocurren en la planta, de acuerdo a la función de transción f_g.
- Las acciones de control vienen dada por la deshabilitación de eventos en la planta P desde el supervisor cuando está en el estado y, tal que $\sigma \notin \Sigma(y)$

$$h(\omega) \cap \Sigma\left(f_t(x_0, \omega)\right) \subseteq \Sigma\left(f_g(y_0, \omega)\right) \subseteq h(\omega)$$

El conjunto *Planta – Supervisor* representado en la figura 3.10 se representa como un par de sistemas comunicantes descritos en el capítulo anterior.

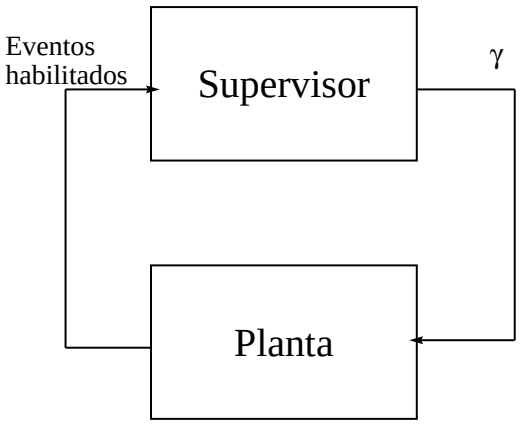


Figura 3.10: Supervisor - Planta

3.4 Controlabilidad y Existencia de Supervisores

La definición de controlabilidad usada en los sistemas de control continuos indica que un sistema es controlable, *si es posible encontrar una acción de control que lleve al sistema a un estado deseado desde cualquier estado en un lapso de tiempo finito.* Un sistema discreto es controlable si se puede construir un supervisor que pueda conducir el sistema hacia un estado deseado en una serie de pasos finitos. El controlador debe entonces conocer las trayectorias posibles de la planta, y los eventos no controlados deben formar parte del lenguaje del supervisor.

Un supervisor S es propio si y sólo si la composición de S/P no tiene bloqueo.

$$\overline{L_m}(S/P) = L(S/P),$$

esto es, cada trayectoria dentro del sistema de lazo cerrado puede ser continuada hasta alcanzar un estado marcado.

Para que todos los estados marcados puedan ser alcanzados por el lenguaje $L(S/P)$, $L_m(S)$ y $L_m(P)$ no deben tener conflictos; i.e:

$$\overline{L_m}(S/P) = \overline{L_m(s) \cap L_m(P)} = \overline{L_m}(S) \cap \overline{L_m}(P)$$

esto es: $L_m(S)$ y $L_m(P)$ son lenguajes sin conflicto y S/P es *limpio* debido a que todos sus estados son alcanzables y co-alcanzables. El teorema de Wonham demuestra la aseveración anterior.

Controlabilidad

Un lenguaje no vacío $K \subseteq L_m(P) \subseteq \Sigma^*$ se dice que es controlable con respecto a una máquina P si:

$$\overline{K}\Sigma_{nc} \cap L(P) \subseteq \overline{K}$$

En otras palabras, la controlabilidad de K implica que cualquier prefijo ω de una cadena en K, seguida de un evento no controlable σ, es tal que $\omega\sigma \in L$. Para el ejemplo de la figura 3.11 con el lenguaje $L(P) = (\alpha\beta\gamma + \alpha\delta\omega)^*(\epsilon + \alpha + \alpha\beta\alpha\delta)$ y un controlador $K = (\alpha\beta\gamma)^*\alpha\beta$ se tiene que $\overline{K} = (abc)^*(\epsilon + a + ab)$

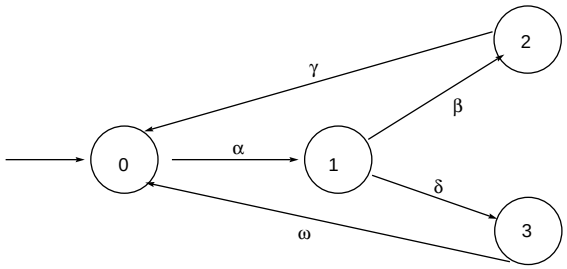


Figura 3.11: Autómata controlable

Si $\Sigma_{nc} = \{\alpha, \beta, \gamma, \omega, \delta\}$ y $\Sigma_c = \{\epsilon\}$, entonces K no es controlable, ya que $\overline{K}\Sigma_{nc} \cap L = (\alpha\beta\gamma)^*(\epsilon + \alpha + \alpha\beta + \alpha\delta)$.

Para $\Sigma_{nc} = \{\alpha, \beta, \gamma, \omega\}$ y $\Sigma_c = \{\delta\}$, entonces K es controlable ya que $\Sigma_{nc} \cap L = (\alpha\beta\gamma)^*(\epsilon + \alpha + \alpha\beta)$.

3.4.1 Cálculo de Lenguajes Controlables

Ramadge y Wonham en [38] definen la unión de todos los sublenguajes controlables de un lenguaje dado L como el *Lenguaje Controlado Supremo* $K^\uparrow$.

$$K^\uparrow = \cup\{K : \ K \subset L \text{ y } K \text{ sea controlable}\}$$

Además probaron que $K^\uparrow$ está bien definida y es única. Esto quiere decir que $K^\uparrow$ es cerrada bajo uniones arbitrarias y entonces contiene todos los sublenguajes controlables de un lenguaje dado L. Si K y L son lenguajes regulares, entonces $K^\uparrow$ también es regular. El supervisor que genera este lenguaje es conocido como el supervisor *mínimo restrictivo*.

En [54] Wonham y Ramadge proponen dos métodos para el cálculo de los lenguajes regulares controlables y sus MEF asociadas. El primero caracteriza el lenguaje supremo controlable $K^{\uparrow}$ como la mayor función de punto fijo $\Omega : \Gamma \rightarrow \Gamma$ definido como

$$\Omega(K) = L \cap sup\{T \subset \Sigma^*, T = \bar{T} \text{ y } T\Sigma_{nc} \cap L' \subseteq \bar{K}\}$$

donde $K \in \Gamma$
Γ es potencia de Σ^*
L y L' son lenguajes establecidos
T es un lenguaje que es cerrado y controlable en K

Si L es regular, el supervisor puede ser determinado como el límite de una secuencia de lenguajes K_j dado por

$$K_{j+1} = \Omega(K_j),\ j \geq 0,\ K_0 = L$$

Síntesis de $sup\mathbf{C}(K)$

En la síntesis de $sup\mathrm{C}(K)$ para:

- K: Un lenguaje regular
- P: Un planta con el comportamiento descrito mediante $L(P)$ y $L_m(P)$
- S: un generador limpio $L_m(S) = K \subset L(P)$

Algoritmo

1. Obtener P', un generador limpio tal que $L(P') = L_m(P')$=L(P). (Marcar todos los estados en P)
2. Calcular $C_0 = P'//S$; con $i = 0$. Nótese que $L_m(C_0) = L_m(P') \cap L_m(S) = L(P) \cap L_m(S) = K$.
3. Identificar los estados indeseados en C_i; estados (x, y) tal que $\Sigma_{nc}(P')(x) \not\subset \Sigma_{nc}(C_i)(x, y)$
4. Obtener C_i' mediante la eliminación de los estados indeseados en C_i y las transiciones asociadas. Hacer $C_{i+1} = Limpio(C_i')$
5. Si $C_{i+1} = C_i$ finalizar el proceso; de otra manera regresar al paso 3.

El segundo enfoque considera solo los lenguajes modelados mediante MEF. También es un proceso iterativo, en el cual una aproximación del lenguaje controlable y su MEF asociada. El procedimiento se basa en los siguientes principios.

Se asume la existencia de $P = \{Q, \Sigma, f_t, q_0, Q_m\}$ y $C_j = \{X, \Sigma, f_{xt}, x_0, X_m\}$ la MEF que realiza el lenguaje de la planta y la aproximación K_j al lenguaje controlable respectivamente.

Entonces se define lo siguiente:

- Una aplicación única $f : X \mapsto Q$, donde f indica la correspondencia entre los estados de la planta y la aproximación C_j.
- El conjunto de estados activos q, $\Sigma(q) \subset \Sigma$, es un subconjunto de las transciones para cada estado q, donde $f_t(\sigma, q)$ está definida.
- La restricción de actividad de eventos, $(\Sigma[f(x)] \cap \Sigma_{nc}) \subset \Sigma(x)$, para cada estado x en C_j.

La restricción de actividad de eventos es comprobada para cada estado en la aproximación C_j. Si el estado x en C_j no lo satisface, significa que existe un estado en la planta P que puede ejecutar una transción no controlable que no ha sido considerada en C_j, y por lo tanto, esa cadena no es controlable. En consecuencia, la construcción de una MEF que sea una realización thel m'aximo lenguaje controlable puede obtenerse eliminando esos estados de C_j que no satisfacen la restricción de eventos activos. Esta MEFes una realización del lenguaje supremo controlable $K^{\uparrow}$.

Algoritmo

1. Obtener P', un generador limpio tal que $L(P') = L_m(P')$=L(P). (Marcar todos los estados en P)
2. Calcular $C_0 = P'//S$; con $i = 0$. Nótese que $L_m(C_0) = L_m(P') \cap L_m(S) = L(P) \cap L_m(S) = K$.
3. Identificar en C_i; si no hay que ir al paso 7
4. Construir C_i' eliminando todas las transciones que corresponden a los eventos controlables en C_i.
5. Determinar otros estados indeseados como aquellos que tienen un arco in C_i' llevando a algún estado identificado como indeseado.
6. Modificar C_i eliminando los estados indeseados identificados en los pasos 3 y 5, y las transiciones respectivas. Hacer $C_{i+1} = limpio(C_i)$
7. Si $C_{i+1} = C_i$; finalizar el proceso ya que $sup\mathrm{C}(K) = L_m(C_i)$; de otra forma ir al paso 3.

3.5 Controlabilidad y existencia de supervisores

Un lenguaje no vacío $K \subseteq L_m(P) \subseteq \Sigma^*$ se dice que es controlable con respecto a una planta P si

$$\bar{K}\Sigma_{nc} \cap L(P) \subseteq \bar{K}$$

En otras palabras, un lenguaje es controlable si las transiciones no controlables nunca causan que una cadena salga del cierre de prefijos $\bar{K}$. La existencia de un supervisor propio S, que en conjunto con una planta P, genera un lenguaje K, el cual es controlable fue demostrado por Ramadge y Wonham en [38]. La propiedad de no conflicto entre $L_m(S)$ y $L_m(P)$ la incluyó Wonham [53] más tarde para garantizar la ausencia de bloqueo en la planta.

3.5.1 Síntesis de supervisores y Controlablidad Condicional

En el control supervisorio se dice que un lenguaje no vacío $K \subseteq L_m(P) \subseteq \Sigma^*$ se dice que es controlable con respecto a una máquina elemental P si $\bar{K}\Sigma_{nc} \cap L(P) \subseteq \bar{K}$. Esto es, si un evento no controlable nunca genera una cadena que salga del cierre $\bar{K}$. Si $K \subset L_m(P)$ es controlable y $L_m(P)$ y $L_m(S)$ no están en conflicto, entonces S es propio y $L_m(S/P) = K$.

La determinación del lenguaje controlable y su MEF de acuerdo a la definición de Wonham y Ramadge, se hace según uno de los algoritmos mostrados en la sección 3.4.1. El concepto de controlabilidad de un proceso que se ha estado usando en este trabajo, indica que si existe un evento que desde un *estado* dado lleve al sistema a tener un comportamiento no deseable, este *estado* debe ser eliminado del espacio de estados controlables. Esto quiere decir, que el supervisor debe tomar

en cuenta todos los eventos no controlables y la conducta que se desarrolla como consecuencia de dicho evento, para evitar la llegada del sistema a estados donde puedan aparecer eventos que lleven a situaciones no deseadas.

Controlabilidad Condicional

Tomado de Sanchez, [42]. En algunos casos se encuentra que el lenguaje máximo controlable es vacío o no conduce a una solución de interés, por lo tanto Sanchez [42] introduce la idea de tener un supervisor propio en el cual existen eventos controlables que en la planta son incontrolables: $\Sigma_{con} \subseteq \Sigma_{nc}$. En el modelado del sistema, debe definirse este subconjunto y se define la existencia de control condicional de acuerdo al parágrafo siguiente:

Un lenguaje $K_{cc} \subseteq L_m(P) \subseteq \Sigma^*$ se dice que es controlable condicionalmente con respecto a una MEF P, si $\bar{K}_{cc}\sigma_{nc} \cap L(P) \subseteq \bar{K}_{cc}\sigma_{con}$, $\sigma_{nc} \in \Sigma_{nc}$, $\sigma con \in \Sigma_{con}$.

En palabras, un lenguaje es *condicionalmente controlable* si un evento no controlable nunca genera una palabra que este fuera del cierre de prefijos seguido de un evento de Σ_{con}. Este enfoque permite la construcción de un supervisor propio condicional S_c, en el cual el lenguaje marcado es el lenguaje controlable condicional de interés.

Existencia de un Supervisor Condicional Si $L_1,\ L_2 \subseteq \Sigma^*$ y L_1 y L_2 no tienen conflicto y son controlables, entonces $L_1 \cap L_2$ es controlable.

Estas con solo condiciones suficientes. L_1 y L_2 puede que no sean controlables, pero su intersección si lo es. L_1 y L_2 pueden tener conflicto, pero su intersección es controlable. El supervisor que realiza $L_1 \cap L_2$ no es necesariamente libre de bloqueo.

Un supervisor propio S_c para una planta P existe si se cumple que $L_m(S_c/G) = K_{cc}$ y K_{cc} es controlable condicionalmente y $L_m(P)$ y $L_m(S_c)$ no están en conflicto. Este tipo de supervisor es conocido como *Supervisor propio.*

Prueba Asumase que

- S_c y P son limpios
- $L(S_c)$ y $L(P)$ son cerrados
- $K_{cc} = L_m(S/P)$.

Si K_{cc} es controlable condicionalmente entonces

$$\begin{array}{lcll} \bar{L}_m(S_c/P)\Sigma_{nc} \cap L(P) & \subseteq & \bar{L}_m(S_c/P)\Sigma_{con} & \text{por definición} \\ \bar{L}_m(S_c/P)\Sigma_{con} & = & [L_m(S_c) \cap L_m(P)]\Sigma_{con} & \text{por definición} \\ & = & [L_m(S_c) \cap L_m(P)]\Sigma_{con} & \text{sin conflicto} \\ & = & [L(S_c) \cap L(P)]\Sigma_{con} & \text{limpio} \\ & = & L(S_c/P)\Sigma_{con} & \text{por definición q.e.d} \end{array}$$

K_{cc} es calculado mediante los mismos métodos utilizados que para lenguajes controlables. Asumase que P y C_j son MEF correspondientes a la planta y a la aproximación de un lenguaje controlable. Si un estado $x \in X$ no cumple con la restricción de actividad de eventos, significa que existen eventos no controlables en el estado de P que no est'n en el conjunto activo $\Sigma(x)$ correspondiente al estado de C_j, que representa al supervisor en síntesis.

Si $\Sigma(x)p[\Sigma(x) \cap \Sigma_{nc}] \neq \emptyset$ en el estado $x \in X$ donde la restricción de actividad de eventos falla, entonces existen eventos controlables que pueden llevar al sistema a un lenguaje controlable condicional si se cumple que $\Sigma[f(x) \cap \Sigma_{nc} \in \Sigma_{con}$. Entonces, la construcción de la MEF que representa el supervisor condicional S_c se obtiene mediante la eliminación de los estados que no cumplen con la restricción de actividad si

1. No hay eventos no controlables en $\Sigma(x)$, o
2. Los eventos no controlables en el estado presente x no pertenecientes al conjunto activo no están en el conjunto Σ_{con}.

3.6 Síntesis modular de supervisores

Cuando existen múltiples especificaciones para un sistema dado, la s'ıntesis del supervisor puede ser realizada por partes, cada tarea de s'ıntesis de un supervisor está referida a una especificación. Las condiciones para la s'ıntesis de cada supervisor son las propuestas porWonham y Ramdge. La s'ıntesis modular simplifica la tarea de construcción del supervisor total.

El supervisor global se consigue mediante la operación de conjunción de las especificaciones individuales. Esta operación se conoce como *Conjunción de supervisores.* La *Conjunción de Supervisores* para dos superviores S y T se define como el producto síncrono de las dos máquinas encontradas, que es una realización de la intersección de los dos lenguajes marcados.

Conjunción de Supervisores ($\sqcap$)

S y T son supervisores completos para una planta dada P, entonces

1. $L(S \sqcap T/P) = L(S/P) \cap L(T/P)$
2. $L_m(S \sqcap T/P) = L_m(S/P) \cap L_m(T/P)$
3. $S \sqcap T$ es un supervisor completo para P

Es necesario notar que la operación de conjunciń no conserva la propiedad de no bloqueo. Sin embargo, si los supervisores S y T no bloquean a la planta P y $L_m(S/P)$ y $L_m(T/P)$ no están en conflicto, entonces $S \sqcap T$ no presentará bloqueo.

Controlabilidad en intersección de lenguajes

Si $L_1,\ L_2 \subseteq \Sigma^*$ y L_1 y L_2 no tienen conflicto y son controlables, entonces $L_1 \cap L_2$ es controlable.

Estas con solo condiciones suficientes. L_1 y L_2 puede que no sean controlables, pero su intersección si lo es. L_1 y L_2 pueden tener conflicto, pero su intersección es controlable. El supervisor que realiza $L_1 \cap L_2$ no es necesariamente libre de bloqueo.

3.7 Control Optimo para SDED

Segunpta y Lafortune en [45] proponen las bases para la optimización de Sistemas Dinámicos de Eventos Discretos.

3.8 Control Jerárquico de Sistemas Discretos

Un sistema complejo está compuesto por un grupo de sistemas elementales, los cuales pueden ser supervisados independientemente, y coordinados por un sistema de orden superior, formando una jerarquía de sistemas. Existen varios enfoques para establecer la jerarquía de los sistemas, dentro de los cuales podemos enumerar el propuesto por Brave y Heymman en [**?**], el utilizado por Wonham [**?**], las "state charts" en [19, 20]. Utilizaremos en este trabajo una composición basada en los sistemas comunicantes descritos en el capítulo 2.

Capítulo 4

Bases para el desarrollo de Sistemas Híbridos

4.1 Introducción

Este capíitulo trata sobre las bases conceptuales que permiten el desarrollo de sistemas híbridos a partir de una descripción parcial de un sistema bajo control sobre todo el espacio de estado del sistema.

En el enfoque clásico de sistemas híbridos, se hace la referencia a sistemas donde una dinámica discreta controla una dinámica continua. En nuestro caso, utilizamos una generalización de estos aspectos al caso donde una dinámica continua es representada por otra dinámica menos rica pero con la suficiente información, para que un supervisor pueda controlar el sistema mediante el envío de una secuencia de señales de paso. Una gran cantidad de aplicaciones se han encontrado para este tipo de sistemas, entre ellos tenemos: los sistemas de manufactura, el procesamiento por lotes, los sistemas de control de vuelo y los sistemas de control inteligente [1, 5, 30, 35, 40].

Los modelos clásicos los hemos clasificado en cuatro grupos básicos, autómatas, sistemas dinámicos, estructuras algebraicas y lenguajes programables.

4.1.1 Clasificación Básica de HyDS

1. Autómatas. Los principales exponentes de los HyDS como autómatas entre otros son: Sreenivas y Krogh (1.991), Alur 1.993, Antsaklis, Lemmon y Stiver 1.993, Desphande y Varaiya (1.995), Nerode y Kohn (1.993) Tittus y Egardt (1.994), Referidos en [47], [16], citeAl-Co-He,[32] y [25].

2. Sistemas Dinámicos. Los principales exponentes de los HyDS como sistemas dinámicos son: Branicky (1.995), Brockett (1.993), Passino (1.993). Referidos en [5], [9] y [34].

3. Estructuras Algebraicas. Los principales exponentes de los HyDS que utilizan estructuras algebraicas abstractas en sus descripciones son: Forsman (1.994), Grossman y Larson (1.995). Referidos en [17] y [18].

4. Lenguajes Programables. Los principales exponentes de los HyDS que utilizan lenguajes programables son: Hooman (1.993), Lamport (1.993), Benveniste Le-Borgne y Le-Guernic

(1.993). Referidos en [22] y [4].

Haremos a continuación una breve descripción de cada uno de éstos diferentes enfoques.

4.2 Autómatas

En el diseño actual de controladores para sistemas dinámicos complejos la ingeniería se ha visto en la necesidad de satisfacer altos grados de tecnología que anteriormente se resolvía con operadores humanos.

La creciente exigencia de automatización en procesos complejos ha desarrollado el estudio de estrategias de control inteligente.

En este tipo de modelo los controladores se representan como un autómata que interactúa con la planta (o proceso físico) a través de la interfaz.

La dificultad principal que surge en este tipo de enfoque, es la complejidad de los procesos de interfaz entre dos sistemas de naturalezas diversas analógica y digital.

Algunos autores como Sastry [44] utilizan representaciones formales de autómatas, en el contexto clásico utilizados en el capítulo anterior. Como vimos un autómata es un grafo cuyos vértices son los estados del sistema $(x, p) \in X^C \times X^E$ y los lados los eventos. El sistema evoluciona de un estado a otro por la ocurrencia de eventos $(u^c, u^e) \in U^c \times U^e$. Hay una ecuación diferencial asociada a cada vértice del grafo y el estado continuo x satisface dicha ecuación en el vértice.

4.2.1 Modelo de Krogh

De manera clásica, un paso importante en el caso de los sistemas híbridos está dado por la existencia de sistemas que interactúan y que poseen dinámicas diferentes tal como se muestra en la figura 4.1 tomada de [47].

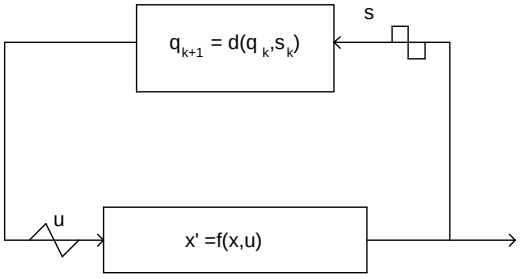


Figura 4.1: Sistema Híbrido según Srenivas - Krogh

En este caso, el sistema evoluciona de acuerdo a una secuencia de funciones constantes a trozos que llevan la planta desde un punto inicial x_0^k correspondiente al punto final del paso anterior x_f^{k-1} hasta un punto final en el paso presente. La dinámica continua de la planta está dada por la concatenación de dinámicas correspondientes a cada intervalo. Como ejemplo tomemos el tanque dado en la figura 4.2, para el cual sólo se controla la entrada mediante un autómata.

La evolución del autómata y la salida de la planta se dan en la figura 4.3. El autómata, que representa el comportamiento de la planta, resulta de la extracción de los eventos posibles obtenidos

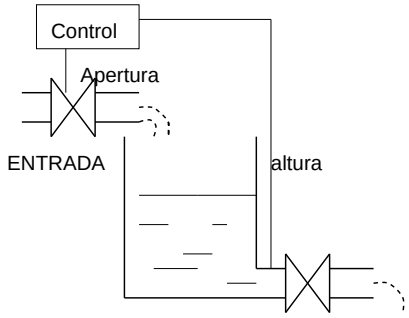


Figura 4.2: Sistema Tanque - Controlador

de los valores continuos del sistema bajo control; estos valores corresponden al paso de ciertas cotas dependiendo del estado en que se encuentra el controlador. La ecuación que rige el sistema es:

$$\dot{x} = f(x, u_k, t),$$

donde $u_k = cte$ durante un lapso de tiempo $(t_k, t_k + \tau)$.

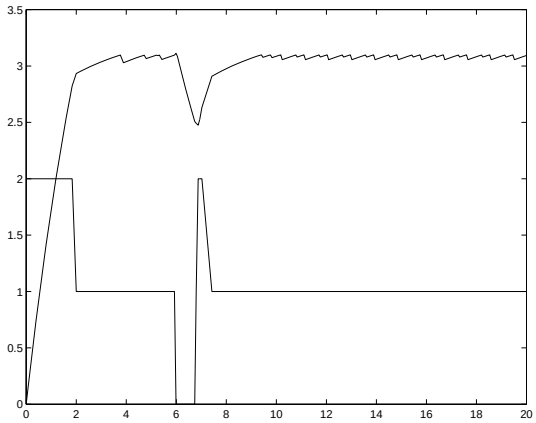


Figura 4.3: Relación entre la dinámica continua y la secuencia de pasos discretos

Este modelo simple permite realizar controles bastante limitados, sólo posible, si el comportamiento continuo del sistema es sencillo, o el secuenciamiento de controles u es muy fino, de tal manera que el control siga completamente al proceso físico.

4.2.2 El modelo de Antsaklis, Lemmon, Stiver

En [28], el sistema está compuesto por el sistema de control que es discreto, la planta y una interfaz, según se muestra en la figura 4.4. El comportamiento del sistema continuo o planta se describe mediante la $p+1$-tupla $P_c = \left(X, \Phi_t^{r_1}, \ldots, \Phi_t^{r_p}\right)$, donde $X \subset \Re^n$ y $\Phi_t^{r_i} : X \mapsto X$ para $i = 1, \ldots, p$.

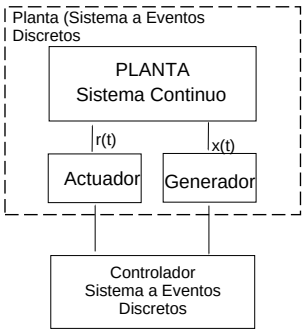


Figura 4.4: Modelo Lemmon Antsaklis para sistemas híbridos

$\Phi_t^{r_i}$ es generado por la ecuación diferencial $\dot{x}(t) = f(x(t), r_i)$. El control de la misma se logra por la conmutación de los operadores de transición $\Phi_t^{r_i}$ en el tiempo.

Para el ejemplo del control del llenado del tanque mostrado en la figura 4.2, el control mantenido mediante el mismo esquema de conmutación, pero con un control local del tipo PID adaptado a cada una de las condiciones de operación, y el resultado se muestra en la figura 4.5.

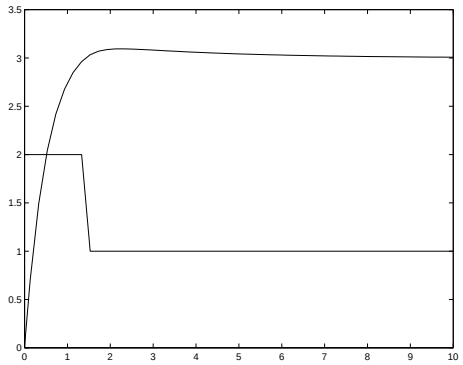


Figura 4.5: Tanque controlado mediante la commutación de la función

4.2.3 Viabilidad de Sistemas Híbridos. Khon, Nerode, Remmel y Yakhnis

Khon et alea en [25] trata sobre el control de sistemas continuos mediante un autómata que asegure que el sistema continuo siga una trayectoria definida. Para eso, el autómata genera funciones de control $c(t)$ las cuales forzan a que el estado $x(t)$ permanezca dentro de un determinado conjunto, denominado el conjunto de viabilidad (VS), para unas perturbaciones $d(t)$. Esto lo denominan como viabilidad en los sistemas híbridos. El modelo del sistema dinámico se describe como:

$$\dot{x}(t) = f(x(t), c(t), d(t))$$

donde x representa el estado del sistema, c una función de control y d las perturbaciones para el sistema. El sistema está definido en el intervalo $[0, \infty]$. El sistema puede descomponerse en intervalos de tiempo $[0, \Delta], [\Delta, 2\Delta], \ldots$, y para cada intervalo de tiempo, debe existir una función de control c_n y manejar una perturbación d_n. Esto lleva a que el sistema pueda ser descrito de manera general mediante.

$$\dot{x}(t) = f(x(t), \tilde{c}(t), \tilde{d}(t))$$

donde $\tilde{c}(t), \tilde{d}(t)$ son funciones parametrizables en el intervalo $[0, \infty]$.

La evolución en cada intervalo n está dada por:

$$\dot{x}(t) = f(x(t - n\Delta), \tilde{c}(t - n\Delta), \tilde{d}(t - n\Delta))$$

transformando el intervalo a $[0, \Delta]$, tenemos que el para espacio de estado S un conjunto de controles admisibles C en $[0, \Delta]$ y un conjunto de perturbaciones admisibles D en el intervalo $[0, \Delta]$. Asi la trayectoria $x(t)$ en el intervalo $[0, \Delta]$ es única para un control $c = c(t)$ y una perturbación $d = d(t)$.

La trayectoria total del sistema esta determinada por la concatenación de trayectorias parciales asociadas a cada intervalo.

El autómata que describe el comportamiento de la planta tiene como alfabeto de entrada perturbaciones d_n, y un control c_n lo cual determina un nuevo estado de la planta en el intervalo $n\Delta$. Este estado está asociado a la trayectoria continua $x(t)$ para un estado inicial $x(0) = s_0$, con una perturbación inicial d_0 y un control c_0.

Viabilidad local

Se define entonces al subconjunto VS de los estados de la planta como el subconjunto viable que cumple la condición de que para el intervalo $[0, \Delta]$ $x(t)$ pertenece al subconjunto VS. A partir de los subconjuntos viables se definen los nodos que son viables. Si existe transición entre los diferentes nodos, se dice que el sistema es viable.

4.2.4 El enfoque de implantación de Lennartson, Tittus, et alea

Un modelo más general para el control de sistemas híbridos está dado por Lennartson, Tittus y otros en [29]. Este modelo describe dos submodelos para efectuar la supervisión en lazo cerrado.

Modelo incluyendo una imagen de la planta

En el primer modelo, la planta es controlada de manera híbrida, y mediante sensores (generadores de eventos), se extrae información de la misma para deducir el estado y de esa manera ajustar el controlador, tal como se muestra en la figura 4.6. La supervisión se realiza a través de un modelo discreto de la planta, el cual es actualizado a partir de los valores del supervisor y de un generador de eventos de la planta.

Los elementos que componen el sistema son una parte discreta, resultado de la discretización del sistema continuo mediante la captura de eventos que indican el arrivo del sistema continuo a una *región* o la imposición de una región mediante un control discreto, la parte continua comandada mediante una secuencia de controles continuos y una interfaz.

- **Parte Discreta**

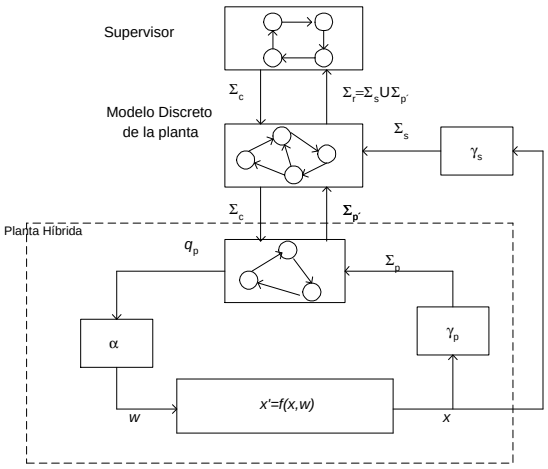


Figura 4.6: Supervisión en lazo cerrado de Sistemas Híbridos.

- **Las regiones de operación**. Resulta en un mapa del estado continuo en eventos discretos de la transición entre regiones continuas en un espacio de estados. A cada región se le asigna un único evento de entrada, resultando en un cambio de estado en el controlador y en la imagen de la planta cuando se produce un evento.

  Una región se especifica generalmente mediante una región convexa $\rho(x) = Ax + b > 0$ que satisface la relación: $\rho(x) > 0$ donde $\rho : X \mapsto \Re^k$. El estado x se considera en una región si la relación es satisfecha.

- **El convertidor de eventos en continuo.** Transforma un símbolo de salida del controlador en una señal de control continua.

- **La imagen de la planta**. La imagen de la planta es capturada mediante la detección de eventos no controlables que aparecen en la planta y del conjunto de eventos controlables generados por un supervisor. Los eventos no controlables son medidos directamente por γ_p o a través del controlador directo. El sistema híbrido evoluciona acorde a las entradas :$\Sigma_h = \Sigma_p \cup \Sigma_c$, conocidos como eventos físicos y eventos de control. Esta función está parcialmente definida para un conjunto de eventos: $\sigma \in \Sigma$, en cada estado.

  La dinámica del sistema está dada por

$$\delta_h : Q_h \times \Sigma_h \mapsto Q_h$$

- **Parte Continua**. Describe la evolución continua del sistema físico a través de la evolución de un sistema dinámico de variables continuas.

  - **El espacio de estados**. $X \subseteq \Re^n$ es el conjunto formado por los estados x de la planta. El vector de estado es denotado $x(t) \in X$. Las restricciones físicas del sistema determinan valores posibles de $x(t)$ dentro de $\Re^n$.

  - **Las señales de entrada**. $W \subseteq \Re^m$ es un subconjunto de dimensión $\leq$ m definido para las señales de entrada $w(t) \in W$. Esta señal depende de los cambios de estado en el controlador directo.

– **La dinámica continua**. La dinámica continua viene dada por la función $f : X \times W \mapsto X$. La ecuación que rige el sistema es:

$$\dot{x}(t) = f\left(x(t), w(t)\right)$$

La ecuación anterior puede expresarse como

$$\dot{x}(t) = f_0\left(x(t)\right) + \sum_{i=1}^{m} w_i(t) f_i\left(x(t)\right)$$

Esta ecuación es equivalente a la propuesta por Lemmon y Antsaklis en [28].

– **El controlador discreto**. Es un autómata que sigue al proceso físico controlándolo bajo de un esquema de funcionamiento. El controlador discreto tiene una evolución que es determinada por $\delta : Q \times \Sigma \mapsto Q$.

- **La interfaz**

– **El generador de eventos de la planta**. Este sistema detecta los cambios que aparecen en la planta para que el control discreto actúe. La interfaz está dada por la aplicación siguiente: $\gamma_p : X \mapsto \Sigma_p$. Estos eventos son no controlables en el sentido definido por Ramadge, Wonham [38]

– **El convertidor de estados en señales continuas**. Está representado por la aplicación $\alpha : Q \mapsto W$, α transforma el estado del controlador en un vector de entradas de control a la planta.

Modelo sin imagen de la planta

Un segundo modelo de sistemas híbridos viene dado por la composición de la planta y su controlador, donde el controlador es un sistema continuo, y el supervisor secuencia cambios en el controlador. El supervisor contempla en su especificación el modelo de la planta híbrida, no necesitando construirse la imagen de la planta para efectuar el control. Este modelo se muestra en la figura 4.7.

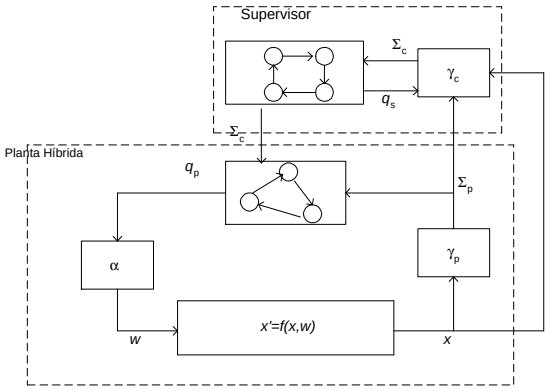


Figura 4.7: Supervisión en lazo cerrado de Sistemas Híbridos sin imagen de la planta

4.3 Sistemas Dinámicos.

Un sistema dinámico es un modelo abstracto matemático de un fenómeno natural o artificial llamado sistema físico. Estos modelos contemplan dos tipos de variables externas (entrada-salida) e internas (estados) y dos transformaciones: una que representa la evolución de los estados y la otra el valor de observación o salida. Las variables del sistema físico pueden ser representadas por funciones que varían continuamente con el tiempo o funciones que varían por ocurrencias de eventos discretos. En el primer caso hablamos de sistemas dinámicos continuos y en el segundo de sistemas dinámicos discretos

En este enfoque un HyDS es presentado como la interacción de dos subsistemas uno discreto y el otro continuo con una regla de conmutación entre ellos (Switching Systems).

Los modelos utilizados para describir HyDS como un sistema dinámico consideran una extensión de la ecuación (1) $\dot{x} = F(x,p)$ donde el campo vectorial $F : \Re^n \times \Re^k \rightarrow \Re^n,\ x \in X,\ p \in P$ (numerable), describe la evolución de los estados (x,p) del sistema. El dominio de F se representa como la unión de componentes de una variedad diferenciable $X = \bigcup X_i$ donde el campo vectorial puede variar discontinuamente de componente a componente X_i. La partición de X en regiones X_i está dada por los cambios discretos del campo vectorial.

Una solución (x,p) de (1), es tal que x es continua y p constante a trozos y la ecuación (1) se satisface en el sentido usual entre saltos de $p \in P$. Luego la evolución del sistema híbrido es visto como una secuencia de segmentos de trayectoria donde el punto final de un segmento está conectado con el punto inicial del siguiente por una transformación, que actúa de acuerdo a la variable discreta.

La dificultad que surge en considerar un HyDS en la teoría de sistemas dinámicos, es la caracterización de los flujos del sistema debido a los cambios frecuentes que sufren las trayectorias, éste puede ser un problema muy difícil de atacar con las herramientas clásicas que disponemos en el presente.

En la figura 4.8 se visualiza, una solución de un sistema híbrido autónomo en el enfoque anteriormente expuesto donde $card(P) = 3$

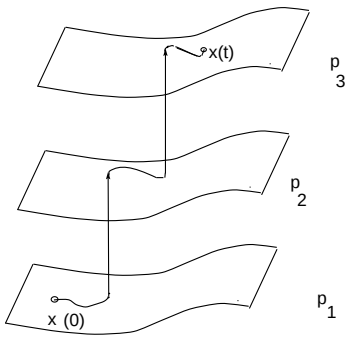


Figura 4.8: Representación de una solución

El campo vectorial cruza las componentes X_i de X transversalmente.

Los ejemplos clásicos para modelar un HyDS en el enfoque de sistemas dinámicos, surgen de la mecánica y biología básicamente. Estas aplicaciones casi siempre requieren del uso de ecuaciones diferenciales en general muy difíciles de resolver por la complejidad del modelo real a representar.

El modelo de Back, Guckenheimer y Myers implementa un simulador dinámico para facilitar el estudio de los HyDS como sistemas dinámicos, pero a pesar de su utilidad la imágen que se representa puede estar alejada de la realidad de la planta física.

En este campo falta mucho por desarrollar y las herramientas matemáticas, con que se cuenta, están todavía en gran dependencia de la teoría clásica de sistemas dinámicos de variable continua, y la intervención del componente discreto aun no se considera en estrecha correspondencia con la parte continua.

4.3.1 El Modelo Dinámico propuesto por Branicky

El modelo propuesto por Branicky en su tesis doctoral [5] y posteriormente descrito en [6] presenta a los Sistemas H´íbridos como una colección de sistemas dinámicos evolucionando en espacios de variables continuas y sujetos a controles continuos y transciones discretas.

Formalmente un sistema dinámico híbrido controlado (CGHDS) es:

$$H_c = [Q, \Sigma, A, G, V, C, F]$$

donde:

- Q es el conjunto de *estados discretos* o *estados índices*
- $\Sigma = \{\Sigma_q\}_{q \in Q}$ es la colección de sistemas dinámicos controlados, donde cada $\Sigma_q = [X_q, \Lambda_q, f_q, U_q]$ es un sistema dinámico controlado. X_q son espacios de estado continuos; U_q es el conjunto de controles continuos y f_q es la dinámica continua del sistema.
- $A = \{A_q\}_{q \in Q}$, $A_q \subset X_q$ para cada $q \in Q$, es la *colección de saltos propios.*
- $G = \{G_q\}_{q \in Q}$ donde $G_q : A_q \times V_q \mapsto S$ es la *aplicación de transiciones de los saltos propios*, parametrizados por el conjunto de control de transiciones V_q, que es un subconjunto de la colección $V = \{V_q\}_{q \in Q}$. Estos representan la dinámica discreta y su control.
- $C = \{C_q\}_{q \in Q}$, $C_q \subset X_q$, es la colección del conjunto de saltos controlados.
- $F = \{F_q\}_{q \in Q}$, donde $F_q : C_q \mapsto 2^S$ es la colección de aplicaciones de destino de saltos controlados.

4.4 Estructuras algebraicas.

La dificultad que surge de considerar HyDS en el enfoque de sistemas dinámicos, es la de caracterizar los flujos del sistema por los cambios frecuentes de modos de operación. Para algunos autores como Robert Grossman y R. Larson la caracterización de los flujos de los sistemas dinámicos, puede reducirse a un problema algebraico más sencillo. El espacio de estado X puede ser representado por el álgebra de funciones $R = Fun(X)$. La evolución de un estado $x \in X$ en el tiempo puede ser considerado como una acción en el álgebra R. En el caso de sistemas continuos está acción es considerada como una derivación en R, en el caso de sistemas discretos esta acción es un endomorfismo de R. Más precisamente, para sistemas continuos, un vector tangente al espacio de estados induce una derivación E en el álgebra, o sea, E es una aplicación lineal $E : R \to R$ que satisface:

$$E(fg) = fE(g) + E(f)g \qquad\qquad f, g \in R$$

Para espacios discretos la acción de una entrada (evento) $w \in W$ sobre el estado $X \times W \to X, \quad (x, w) \to x' = x.w$ induce una acción en el álgebra

$$w \times R \to R \quad (w, f) \to (w.f)(x) = f'(x) = f(x.w)$$

la cual es un endomorfismo.

Este punto de vista algebraico, da origen a estructuras abstractas que requieren un desarrollo teórico más general de los HyDS, del que se ha desarrollado hasta el momento. Nuestro enfoque algebraico se desarrolla sin intersectar este punto de vista. Aquí consideran las dinámicas de los subsistemas como acciones en unas álgebras abstractas, nosotros las consideramos en su estructura real.

4.5 Lenguajes de programación.

Un programa de cumputación es un DES, por lo que puede ser descrito por máquinas secuenciales. Los lenguajes de programación, son la herramienta para construirlos. De acuerdo a la forma de ejecución, éstos se pueden clasificar en síncronos y asíncronos. Por programas síncronos entendemos aquellos programas cuya ejecución se realiza de acuerdo a un reloj, mientras que los programas asíncronos se refieren a aquellos que son disparados por eventos. Para la construcción de cada tipo de programa existen lenguajes adecuados, en el caso de programas síncronos tenemos Pascal, C, entre otros y en programas para lenguajes asíncronos están Lustre, Signal, Esterel. Hoare en [21] desarolla una forma de probar programas secuenciales conocida como CSP (Communicating Sequential Processes) para validar programas que se ejecutaban en paralelo. Estos trabajos son extendidos en la verificación de los sistemas por Manna y Pneulli. Posteriormente, Milner [31] desarolla un álgebra para la verificación de sistemas comunicantes llamada CCS. todas estas técnicas de prueba transforman los programas en sistemas de transiciones o autómatas.

Para el desarrollo de Protocolos de Comunicación, fué necesario la construcción de lenguajes y paquetes de verificación de sistemas basados en sistemas de transiciones y mas especialmente en Autómatas, y así aparecen lenguajes como Estelle, SDL. Éstos últimos dan origen a lenguajes de aplicación en sistemas de control en tiempo real como Lustre y Signal.

Para el desarrollo de Sistemas de Control en base a los lenguajes antes descritos, se establece la idea de un controlador de sistemas continuos basados en el uso de lenguajes síncronos, que modelan al sistema en base a ecuaciones en diferencias, y controladores para sistemas discretos basados en el uso de lenguajes asíncronos. Ésto se conoce también como programación reactiva. Si el sistema continuo se puede controlar mediante lenguajes asíncronos se tiene un sistema híbrido.

4.6 Un enfoque algebraico abstracto para SDH

En esta sección presentamos un enfoque algebraico abstracto en el concepto de Sistemas Dinámicos Híbridos (SDH).

Si tomamos como modelo de comparación la representación dada de estos sistemas por un autómata, y que en resumen consiste en expresar el subsistema continuo (planta) como un sistema de ecuaciones diferenciales y el subsistema discreto como un SDED encargado de definir las transiciones discretas del proceso continuo, la interfaz, o proceso que transforma informaciones lógicas en analógicas y viceversa, en general presenta una descripción muy compleja, debido a las diferencias de las estructuras de los subsistemas del SDH consideradas. En nuestra construcción

primero unificamos la descripción de los subsistemas, desarrollando el concepto de Ps-DS (continuos o discretos). Tomando en cuenta que las dinámicas no actúan sobre todo el espacio considerado, definimos la noción de pseudo-dinámicas y los semigrupos de acción no son los clásicos.

Nuestra concepción no limita la naturaleza de los subsistemas (continuo o discreto) que interactúan, a pesar que la motivación de la definición es tomada del enfoque tradicional de HyDS, que en nuestro lenguaje será: la planta repesentada por un Ps^c-DS (realización del sistema entrada salida) y considerando un diagrama conmutativo la imagen de la planta como un Ps^e-DS.

Es de observar que consideramos el término híbrido en un contexto más general del clásico, como la interacción de dos subsistemas ambos continuos o ambos discretos. La estructura del diagrama limita solo las riquezas de las dinámicas, no su naturaleza. Obviamente la imagen de la planta (nivel inf.) del diagrama tendrá una dinámica menos rica que la planta (nivel sup.)

A continuación daremos algunas definiciones con sus respectivos ejemplos, ya que son parte de la construcción de nuestra teoría y no constituyen un material clásico de HyDS considerado en la literatura. El contenido de esta sección está desarrollado principalmente en nuestros trabajos publicados sobre HyDS, para mas detalles ver [48] y [12].

4.6.1 Sistemas Dinámicos

La tripleta (X, S, Φ) denotará un sistema dinámico (SD), donde X es el espacio de estado, S un semigrupo y $\Phi : X \times S \longrightarrow X$ la función de transición, que satisface las siguientes propiedades:

a) $\Phi_u(x) = \Phi(x, u)$ es una transformación del espacio de estado X (morfismo) para todo $u \in S$.

b) $\Phi_{uv} = \Phi_u \circ \Phi_v$ (covariante) o
$\Phi_{uv} = \Phi_v \circ \Phi_u$ (contravariante)

c) Si S es un monoide y ϵ es un elemento neutro, entonces $\Phi_\epsilon = Id_X$.

En los sistemas dinámicos a tiempo continuo S es considerado $\Re^+$, y los estados evolucionan con el tiempo de acuerdo a un sistema de ecuaciones diferenciales.

En el trabajo desarrollado por Ramadge y Wonham, un SDED puede ser modelado como SD sobre un alfabeto Σ y la dinámica se define por la concatenación de elementos de Σ, y es considerada una dinámica discreta.

En ambos casos (dinámica continua o discreta) el semigrupo de acción no está siempre definido sobre todo el conjunto $X \times S$. Por ésta razón deseamos considerar la acción de una aplicación parcialmente definida Φ con dominio un subconjunto de $X \times S$. Introducimos por ello el concepto de sistema pseudo-dinámico, y que será utilizado en lo sucesivo.

4.6.2 Sistemas pseudo-dinámicos

Un sistema pseudo-dinámico $(Ps - DS)$ es un SD, (X, S, Φ) donde Φ es un semigrupo de acción parcialmente definido, sobre $X \times S$. El dominio de Φ es el subconjunto $\bigcup_{x \in X} \{x\} \times S_x \subset X \times S$, donde $S_x = \{u \in S / \Phi(x, u)$ está definido$\}$. Un $u \in S_x$ es llamado x-admisible. Cuando los estados $x \in X$ evolucionan continuamente en el tiempo, al $Ps - DS$ lo denotaremos $Ps - D^cS$, si los cambios de estado corresponden a una ocurrencia de un evento, entonces el sistema tendrá la notación $Ps - D^eS$, diferenciando de esta manera la naturaleza continua o discreta de las dinámicas de los sistemas.

Ejemplo de un sistema pseudo-dinámico continuo

Sea U un conjunto arbitrario, y $S(U)$ denota a la función constante a trozos con valores en U, $u : [0, T_u) \to U$, con $T_u = \sum_{i=1}^{k} t_i$, $t_i \geq 0$, $u(t) = u_i \in U$, $t \in [t_1 + \cdots + t_{i-1},\ t_1 + \cdots + t_{i-1} + t_i)$, $i = 1, 2, \ldots, k$. Denotaremos u mediante $S(\underline{t}, \underline{u}) = S((t_1, \ldots, t_k), (u_1, \ldots, u_k)) = (u_1 t_1, \ldots, u_k t_k)$.

La concatenación de funciones constante a trozos u, v en $S(U)$ está definida por:

$$(u \cdot v)(t) = \begin{cases} u(t) & 0 \leq t < T_u \\ v(t - T_u) & T_u \leq t < T_u + T_v \end{cases}$$

$S(U)$ es un semigrupo con respecto a la concatenación de las funciones constantes a trozos. La función vacía $\epsilon : [0, 0) \to U$ es el elemento neutro de $S(U)$. Las siguientes propiedades son necesarias para $t_i, t_j \in \Re_+$:

a $(ut_i)(ut_j) = u(t_i + t_j)$

b $(u0) = \epsilon$

Sea $X^c \subset \Re^n$ un dominio y $f_{u_i} : X^c \mapsto \Re^n$, $u_i \in U$ un campo vectorial. Consideremos el problema de valor inicial

$$\dot{x} = f_{u_i}(x(t)),\ x(0) = x_0,\ t \in \Re^+ \tag{4.1}$$

Sea $t \mapsto \Phi_{u_i}^{x_0}(t) = \Phi_{u_i}(x_0, t)$ la solución completa a la ecuación (4.1) sobre el intervalo maximal $[0, T_{x_0})$, entonces $\Phi_{u_i} : X^c \times \Re_+ \to X^c$ es la pseudo-dinámica correspondiente a $f_{u_i} : X^c \to \Re^n$. Dado un estado inicial $x_0 \in X^c$, una función constante a trozos $u = (u_1 t_1, \ldots, u_k t_k) \in S(U)$, es admisible para x_0, o $u \in S_{x_0}$ si:

1. $t \mapsto \Phi_{u_1}(x_0, t)$ puede ser definida sobre $[0, t_1]$; donde $x_1 = \Phi_{u_1}(x_0, t_1)$.
2. $t \mapsto \Phi_{u_2}(x_1, t)$ puede ser definida sobre $[t_1, t_1 + t_2]$; donde $x_2 = \Phi_{u_2}(x_1, t_2)$.

   $\vdots$

k. $t \mapsto \Phi_{u_k}(x_{k-1}, t)$ puede ser definida sobre $[t_1 + \cdots + t_{k-1}, t_1 + \cdots + t_{k-1} + t_k)$; donde $x_{k-1} = \Phi_{u_{k-1}}(x_{k-2}, t_{k-1})$.

En otras palabras, el sistema de control dado por la ecuación (4.2) tiene solución sobre el dominio $[0, T_u)$ de el control u.

$$\dot{x} = F\left(x(t), u(t)\right) = f_u\left(x(t)\right),\ x(0) = x_0,\ t \in \Re^+ \tag{4.2}$$

Entonces la pseudo-dinámica $\Phi^c : X^c \times S(U) \mapsto X^c$ dada por la ecuación

$$\Phi^c\left(x_0, u\right) = \Phi_{u_k}\left(\Phi_{u_{k-1}}\left(\cdots\left(\Phi_{u_1}\left(x_0, t_1\right), t_2\right)\cdots\right), t_k\right) \tag{4.3}$$

está bien definida sobre $S_{x_0} \subset S(U)$. Luego $(X^c, S(U), \Phi^c)$ es un $Ps - D^c S$.

Ejemplo de sistema pseudo-dinámico discreto

Sea Σ^* el conjunto de todas las cadenas de elementos del conjunto Σ, incluyendo la palabra vacía ϵ, $\Sigma^* = S^e$ con la operación de concatenación es un monoide. Considérese $X^e \subset \Sigma^*$, un lenguaje prefijo cerrado, X^e será el espacio de estados en nuestro caso discreto. Para $u \in X^e$, el conjunto de cadenas u-admisible es $S^e_u = \{v \in \Sigma^* / uv \in X^e\}$, ($uv$ es la concatenación). El semigrupo de acción parcialmente definido es $\Phi^e : X^e \times S^e \mapsto X^e$, dado por $\Phi^e(u, v) = \Phi^e_u(v) = uv$, $(v \in S^e_u)$, donde $\Phi^e_u : \bigcup_{u \in X^e} \{u\} \times S^e_u \mapsto X^e$. Nótese que $\bigcup_{u \in X^e} \{u\} \times S^e_u \subset X^e \times S^e$. Entonces (X^e, S^e, Φ^e) is a $Ps - D^e S$.

$Ps - DS$ con aplicación de salida

Supóngase que un $Ps - DS$ (X, S, Φ) está equipado con una aplicación de salida (OM) $\varphi : X \mapsto Y$ y un conjunto de salida Y. Entonces se dice que la quintupla (X, Y, S, Φ, φ) es un Sistema Pseudo-Dinámico con aplicación de salida $(Ps - DS\&OM)$.

Supongamos que un $Ps - DS$ (X, S, Φ) está equipado con una aplicación de salida (OM) $\varphi : X \mapsto Y$ un conjunto de salida Y, y x_0 un punto dado en X. Entonces decimos que la sextupla $(X, x_0, Y, S, \Phi, \varphi)$ es un $Ps - DS\&OM$ inicializado.

4.6.3 Mapa con preservación de la dinámica

Considerese un par de $Ps - DS$, $\left(X^1, S^1, \Phi^1, \right)$ y $\left(X^2, S^2, \Phi^2, \right)$ entonces el par (f, h) de mapas $f : X^1 \mapsto X^2$, $h : S^1 \mapsto S^2$ conserva la dinámica si:

a. h es un semigrupo (monoide) con homomorfismo.

b. El diagrama 4.4 conmuta.

$$\begin{array}{ccc} X^1 \times S^1 & \stackrel{\Phi^1}{\rightarrow} & X^1 \\ (f,h) \downarrow & & \downarrow f \\ X^2 \times S^2 & \stackrel{\Phi^2}{\rightarrow} & X^2 \end{array} \tag{4.4}$$

La conmutatividad implica la condición $h(S_{1x}) \subset S_{2f(x)}$.

4.6.4 Equivalencia entre sistemas E/S y $Ps - DS\&OM$

Considérese una aplicación de entrada - salida $P : S \mapsto Y$, S es un monoide con la operación $\circ : S \times S \mapsto S$ $(u, v) \mapsto u \circ v$, θ denota al elemento neutro en S.

Usando la equivalencia de Nerode, en [48] se construyó una realización de la aplicación de entrada salida P de la siguiente manera: si $u, v \in S$ entonces $u \sim_p v$ si y solo si $S_u = S_v$ y $P(u \circ w) = P(v \circ w)$ para todo $w \in S_u = S_v$.

Denotaremos por X el conjunto de todas las clases $[u]$, $u \in S$, el estado inicial es $x_0 = [\theta]$. Sea $S_{[u]} = S_u$, es obvio que $S_{[u]}$ depende de la clase de u. La dinámica parcialmente definida $\Phi : X \times S_u \mapsto X$ es dada por $\Phi_{[u]} : S_{[u]} \mapsto X$; $\Phi_{[u]}(v) = \Phi([u], v) = [u \circ v]$, $\varphi : X \mapsto Y$ es dada por $\varphi([u]) = P(u)$. Es claro que $\varphi\left(\Phi_{[\theta]}(u)\right) = P(u)$.

Aquí el $Ps - DS\&OM$ inicializado $(X, x_0, Y, S, \Phi, \varphi)$ realiza una aplicación de entrada - salida $P : S \mapsto Y$. Esta construcción conduce a una realización canónica minima, ver [46].

Por otra parte, dado un $Ps - DS\&OM$ inicializado $(X, x_0, Y, S, \Phi, \varphi)$, entonces la aplicación parcial definida sobre el conjunto S_{x_0} por la formula $P_{x_0}(u) = \varphi(\Phi(x_0, u))$ $(u \in S)$, es una aplicación de entrada – salida. Entonces la descripción del sistema mediante una aplicación de entrada – salida y $Ps - DS\&OM$ son equivalentes.

4.6.5 Una clase de Sistemas de Control Dinámicos como un $Ps-D^cS\&OM$

Considérese el problema clásico discutido en 4.6.2. El sistema de control y su función de salida están dados por:

$$\dot{x} = f_u(x(t)),\ y(t) = \varphi(x(t)),\ x_0 = x(0),\ t \in \Re^+ \tag{4.5}$$

donde $f : X^c \times U \to \Re^m$, y $\varphi : X^c \to \Re^k$ son funciones. Si se admite controles integrables $u : [0, T_u) \to U \subset \Re^n$, entonces es conocido que se puede aproximar el control u por una secuencia (u_l) de controles tales que:

a. $u_l : [0, T_u] \to U$ sean funciones constantes a trozos $u_l \to u$ definidas en casi todas partes.

b. La trayectoria x_l correspondiente a u_l con la condición inicial $x_l(0) = x(0)$, converge uniformemente a la trayectoria x, correspondiente al control original u.

Entonces la secuencia de salida es $y_l = \varphi \circ x_l$, por lo cual se puede considerar la secuencia de controles escalera en vez de un control integrable. De acuerdo a lo expuesto en el ejemplo 4.6.2, $(X^c, S(U), \Phi^c)$ es un $Ps - DS$, con Φ^c definido de la manera expuesta anteriormente. Entonces por 4.6.4 se puede definir un $Ps - DS\&OM$ inicializado $(X^c, x_0, \Re^n, S(U), \Phi^c, \varphi)$ y un sistema de entrada salida

$$P_{x_0}(u) = \varphi\left(\Phi^c(x_0, u)\right).$$

El problema inverso, de realizar P por un sistema clásico de control se plantea utilizando el teorema de Jakubczyk [24].

Vimos en 4.6.4, que una aplicación de entrada y salida P puede ser realizada por un $Ps - DS\&OM$ abstracto. El punto difícil es equipar la realización abstracta con una estructura topológica diferenciable. B. Jakubczyk [24], ha definido la suavidad de una aplicación de entrada salida, de tal manera que la suavidad en este caso significa, que las funciones

$$\underline{t} = (t_1, \ldots, t_l) \mapsto p\left(\underline{t}, \underline{u}\right) \in \Re^k \text{ son suaves para todo } \underline{u} \in X_{i=1}^{l} U,\ l = 1, 2, \ldots$$

Por lo cual tiene sentido definir el rango máximo de estas aplicaciones. El teorema de Jakubczyk establece, que si el rango máximo es n, existe una realización mínima sobre una variedad X diferenciable n-dimensional donde las transiciones de estados $\Phi(x, u)$ están definidas por un sistema 4.5.

Este hecho es soporte teórico fundamental para el enfoque linguístico aquí utilizado. La descripción linguística abstracta de sistemas continuos, preserva la riqueza de las estructuras diferenciales, las cuales pueden ser reconstruidas según el teorema de Jakubczyk.

4.7 Definición general para los sistemas híbridos

Un Sistema Dinámico Híbrido SDH es una tripleta $(\tilde{X}^1, \tilde{X}^2, F)$ donde

$$\tilde{X}^1 = (X^1, S^1, \Phi^1) \qquad \tilde{X}^2 = (X^2, S^2, \Phi^2)$$

son $Ps - DS$ y $F = (f, h)$ es un mapa de conservación de las dinamicas tal que $f(x_0) = \epsilon$, donde $x_0 \in X^1$ es el punto inicial en $\tilde{X}^1$ y $\epsilon \in X^2$ es el punto inicial en $\tilde{X}^2$.

Notese que los sistemas $\tilde{X}^1$ y $\tilde{X}^2$ pueden ser considerados ambos $PS - D^cS$ (continuos) o $Ps - D^eS$ (discretos). Cuando $\tilde{X}^1$ es un $PS - D^cS$ y $\tilde{X}^2$ es un $Ps - D^eS$ esta definición es equivalenete con el concepto clásico de sistemas hbridos considerado en la literatura [1, 5, 30, 35].

SDH con aplicación de salida

Un sistema dinámico híbrido con aplicación de salida ($SDH\&MS$) es un SDH (X^1, X^2, F) equipado con un par de aplicaciones de salida $(\varphi^1,\ \varphi^2)$, $\varphi^1 : X^1 \mapsto Y^1$ y $\varphi^2 : X^c \mapsto Y^2$. Donde, Y^1 and Y^2 son dos conjuntos de salida. Como anteriormente se dijo, el diagrama siguiente describe la conservación de la dinámica $(F, g) = (f, h, g)$ entre X^1 y X^2 y los mapas de salida, donde g es dado en el diagrama 4.6

$$\begin{array}{ccccc} X^1 \times S^1 & \overset{\Phi^1}{\rightarrow} & X^1 & \overset{\varphi^1}{\rightarrow} & Y^1 \\ (f,h)\downarrow & & \downarrow f & & \downarrow g \\ X^2 \times S^2 & \overset{\Phi^2}{\rightarrow} & X^2 & \overset{\varphi^2}{\rightarrow} & Y^2 \end{array} \tag{4.6}$$

4.8 Ejemplos de sistemas híbridos

Tomemos dos ejemplos de aplicación. El primero se refiere al caso clásico de dos sistemas, donde uno mantiene una dinámica discreta y otro una dinámica continua. El segundo caso corresponde a dos sistemas que poseen dinámicas discretas.

- **Aplicación Continua - Discreta** Esta aplicación está tomada de un análisis de un problema académico tratado en la literatura que es el problema de los tres tanques.
- **Aplicación Discreta - Discreta** que es el caso utilizado a menudo en la supervisión de sistemas de manufactura modelados mediante Redes de Petri.

4.8.1 El sistema Tres Tanques en términos de $HyDS$

Consideremos un ejemplo simple tomado de la literatura que describe el control de un sistema de tres tanques para mantener su nivel estable, sabiendo que el flujo de salida de los mismos es igual al flujo de entrada. Un solo servidor llena los tanques y el flujo de ellos es diferente en cada uno, la suma de los tres es igual al flujo de entrada. Este sistema es dsarrollado en [26] y se muestra en la figura 4.9

El servidor S suministra un flujo $\rho(t)$ por unidad de tiempo al tanque que está siendo servido; $\rho(t) = 1$. Cada tanque pierde ρ_1, ρ_2, ρ_3 unidades de líquido por unidad de tiempo. $\rho_1 + \rho_2 + \rho_3 = 1$. El volumen total del líquido se mantiene constante.

Asumamos que el sistema es autónomo, esto es, que se controla internamente para asegurar que el nivel de los tanques se mantiene dentro de un rango dado. $(x_i^{\min}, x_i^{\max})$.

Si el valor del nivel del líquido puede ser medido por medio de un sensor y el nivel es alterado por la conmutación del servidor S. El modelo del flujo del sistema es dado por la tripleta $(\tilde{X}^1, \tilde{X}^2, F)$ de acuerdo a la definición dada en (4.7).

$$\tilde{X}^1 = (X^1, S^1, \Phi^1) \text{ es un } PS - D^cS$$

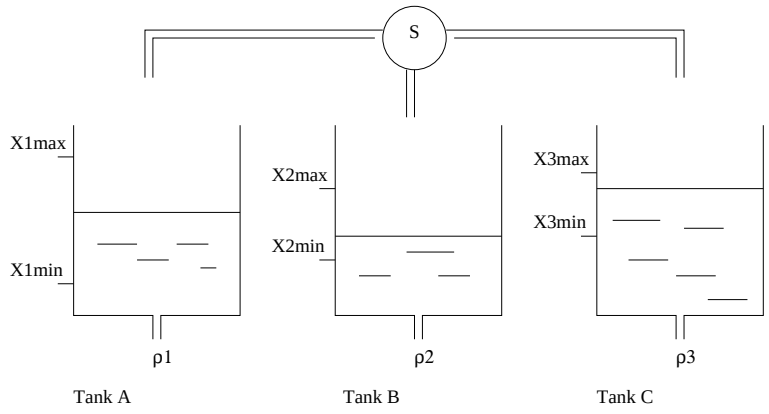


Figura 4.9: El sistema Tres Tanques

y

$$\tilde{X}^2 = (X^2, S^2, \Phi^2) \text{ es un } PS - D^e S$$

Las siguientes asignaciones se hacen:

1. *Espacio de estados continuo* X^1 es considerado como el espacio de estado continuo formado por el espacio vectorial $(x_1(t), x_2(t), x_3(t))$ donde cada $x_i(t)$ representa el nivel del líquido en cada tanque $i = 1, 2, 3$.

2. *Semigrupo de acción* X^1 $S^1 = [0, \infty)$, $\Phi^1 : X^1 \times S^1 \to X^1$.

3. *Dinámica continua* Φ^1 es la dinámica continua dada por el sistema.

$$F_a = \begin{bmatrix} -\rho_1 \\ -\rho_2 \\ -\rho_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} F_b = \begin{bmatrix} -\rho_1 \\ -\rho_2 \\ -\rho_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} F_c = \begin{bmatrix} -\rho_1 \\ -\rho_2 \\ -\rho_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad (4)$$

4. *Espacio de estado discreto* $X^2 = \{a, b, c\}$ corresponde a la posición del servidor en el tanque 1, 2 o 3.

5. Semigrupo de acción sobre X^2 S^2 es el monoide de todas las cadenas finitas de elementos en X^2 incluyendo la cadena vacía $\tilde{\epsilon}$.

6. *Dinámica discreta* La dinámica discreta Φ^2, es un semigrupo de acci'on parcialmente definido y es dado por la secuencia de cambios dados por los eventos $\{\epsilon, 1, 2, 3, 4, 5, 6\}$ considerado en la tabla (4.1) dados en [26], donde ϵ (evento nulo) se identifica con la cadena vacía ϵ aparece si no hay cambio en la posición del servidor. De manera gráfica, la evolución del sistema discreto se muestra mediante el autómata de la figura 4.10.

   No se consideran otros eventos, por lo cual, el resto de cadenas en S^2 son inactivas o no admisibles. Por esta identificación se tiene:

   $S_a = \{\epsilon, 1, 2\}$ $S_b = \{\epsilon, 3, 4\}$ $S_c = \{\epsilon, 5, 6\}$.

   Aquí el dominio de Φ^2 es: $\{a\} \times S_a \bigcup \{b\} \times S_b \bigcup \{c\} \times S_c \subset X^2 \times S^2$.

x^2	Región de Operación	X^1	Evento en el continuo	$x^{2\prime}$
a	F_a	$x_2(t) \leq x_2^{\min} \vee$ $x_1(t) \geq x_1^{\max}$	1	b
		$x_1(t) < x_1^{\max} \wedge$ $x_2(t) > x_2^{\min} \wedge$ $x_3(t) > x_3^{\min}$	ϵ	a
		$x_3(t) \leq x_3^{\min}$	2	c
b	F_b	$x_1(t) \leq x_1^{\min} \vee$ $x_2(t) \geq x_2^{\max}$	3	a
		$x_1(t) > x_1^{\min} \wedge$ $x_2(t) < x_2^{\max} \wedge$ $x_3(t) > x_3^{\min}$	ϵ	b
		$x_3(t) \leq x_3^{\min}$	4	c
c	F_c	$x_1(t) \leq x_1^{\min} \vee$ $x_3(t) \geq x_3^{\max}$	5	a
		$x_1(t) > x_1^{\min} \wedge$ $x_2(t) > x_2^{\min} \wedge$ $x_3(t) < x_3^{\max}$	ϵ	a
		$x_2(t) \leq x_2^{\min}$	6	b

Tabla 4.1: Automata de Deshpande y Varaiya

7. *La conservación de las dinámicas* está dada por el par $F = (f, h)$. De acuerdo con la definición de regiones $F_a,\ F_b,\ F_c$ dada por las condiciones de borde $(x_i^{\min}, x_i^{\max})$, y que se extraen de la tabla 4.1.

$$f(x) = \begin{cases} F_a \longrightarrow a \\ F_b \longrightarrow b \\ f_c \longrightarrow c \end{cases}$$

y $h(t) \in \{\epsilon, 1, 2, 3, 4, 5, 6\}$ es el evento moviendose en el tiempo t, de acuerdo a la tabla 4.1, $h(0) = \epsilon$.

Con estas consideraciones es fácil verificar la conmutatividad del diagrama 4.4, y de como el concepto general de sistema híbrido es compatible con otros modelos.

Los resultados de la simulación del sistema se ven en la figura 4.11.

El sistema puede cambiarse a un modelo más complicado, donde cada tanque genera dos eventos particulares no controlables que son vacío y lleno y existen dos eventos controlables para cada tanque que habilitan o deshabilitan el llenado del mismo. La dinámica de un tanque en particular se da en la figura **??**.

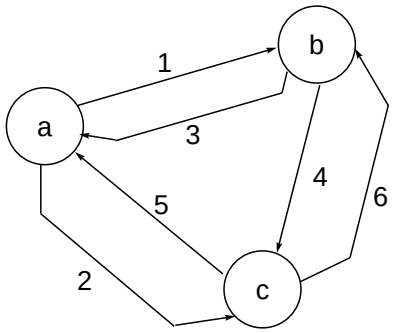
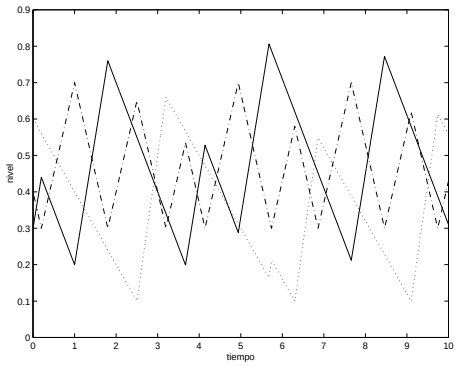
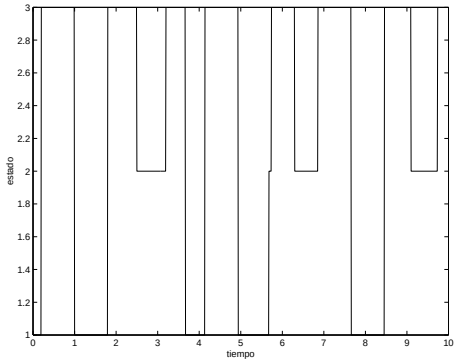


Figura 4.10: Autómata para el sistema tres tanques.



(a). Tanque 1 –, Tanque 2 ., tanque 3 -.



(b)

Figura 4.11: Dinámica en el sistema tres tanques. (a) Dinámica continua, (b) Comportamiento de los estados

Capítulo 5

Modelado y Control de Sistemas Híbridos

5.1 Algunas ideas sobre el modelado

El mundo está compuesto por un conjunto de entes que poseen un conjunto de propiedades o atríbutos y que mantienen una cierta relación entre ellos formando sistemas con nuevas propiedades o atríbutos. Mediante un modelo tratamos de representar un concepto acerca de una realidad física o lógica. Un modelo es, entonces, una abstracción de la realidad representada de alguna forma. Un objeto, ya sea físico o lógico, puede tener diferentes funciones las cuales deben ser modeladas en su totalidad, es por esto que uno o varios modelos cubren los diferentes aspectos del objeto. Tomemos el caso del ejemplo de la planta mostrada como ejemplo en el capítulo 1.4; el intercambiador de calor se modela mediante un conjunto de ecuaciones que describen el proceso de intercambio de calor, para el cuál los parámetros van cambiar dependiendo de los productos que circulan a través de el mismo; otro modelo representa su capacidad de producción con los otros elementos componentes del sistema, al igual que otros modelos son utilizados para indicar cuando el intercambiador debe entrar en mantenimiento. Se puede modelar también su envejecimiento en base a como su rendimiento varía en el tiempo. Para disminuir la complejidad, es mejor utilizar diferentes modelos que cubran los diferentes aspectos de la planta, y luego integrarlos mediante macro-modelos que usen los resultados de modelos parciales anteriormente desarrollados.

Ya que un sistema está compuesto de uno o más elementos que pueden tener relaciones o ser independientes entre sí, las interacciones entre los mismos forman parte de un agregado de los elementos componentes. A partir de las abstracciones hechas de los elementos, es menos complicado el tener modelos de las interacciones que un modelo total del sistema total. En capítulos anteriores nos hemos referido a los sistemas híbridos como aquellos sistemas que presentan dinámicas continuas y discretas simultáneamente, y donde la dinámica continua es controlado mediante un sistema discreto. Aplicando estos conceptos se puede conseguir un modelar el sistema global mediante el modelado de los procesos físicos por un conjunto de ecuaciones diferenciales y controlar el mismo mediante controladores continuos, derivando un conjunto de regiones de operación que se representan de manera discreta y sobre las cuales se puede viajar. Este viaje se modela de manera discreta por un sistema de transiciones. La dinámica total del sistema es entonces descrita mediante una jerarquía de sistemas de transiciones, los cuales se controlan por sistema supervisores

Particularmente, en este capítulo describiremos un proceso para la construcción de un sistema

de automatización integral a partir del modelado total de la planta, mediante el modelado parcial de cada uno de los subsistemas componentes del complejo estableciendo las leyes de control referidas al modelo. La abstracción de la composición planta – control mediante una dinámica discreta que represente el comportamiento del subsistema, nos permite simplificar el modelado, al obviar información no precisa para la construcción del modelo global.

Una vez realizados los procesos de abstracción, se prosigue con la construcción de los sistemas supervisorios que secuencien los cambios de estado en los subsistemas. Los supervisores así encontrados conforman un nuevo sistema el cual será a su vez supervisado por un sistema de mayor nivel. Este proceso lleva a la construcción de un sistema integrado verticalmente mediante el establecimiento de supervisores, horizontalmente mediante el establecimiento de las interacciones entre los subsistemas. Es necesario entender que un sistema posee un conjunto de dinámicas de acuerdo a diferentes observadores, una dinámica está relacionada directamente con el control, mientras que otra se puede referir a la disponibilidad de la planta; en ambos modelos, se tiene una información parcial y a otro nivel, estas dos dinámicas permiten tomar decisiones más complejas.

A partir del segundo nivel, cada subsistema debe ser considerado en diferentes aspectos que cubran modos de operación de la planta: capacidad, rendimiento esperado vs. rendimiento obtenido, disponibilidad (fallas en los equipos, políticas de mantenimiento) y las interacciones entre los subsistemas. Cada subsistema es modelado de acuerdo a sus funciones y sus interacciones.

5.2 Metodología para la construcción del modelo

Para lograr el control global del sistema, es necesario tener un representación del sistema a controlar, que contemple los diferentes aspectos como son:

- **Dinámica interna del proceso.**
  La dinámica interna está asociada al conjunto de leyes físicas que describen adecuadamente la planta en cada una de sus regiones de operación. Si la dinámica para una región de operación es muy difícil de obtener, o el algoritmo de control asociado a dicha región de operación no es la suficientemente robusto, se descompone esta región de operación y se desarrollan modelos parciales para cada una de las subregiones (Puntos de operación).

- **Dinámicas asociadas a condiciones de la planta.**
  En este caso se hace referencia a la disponibilidad de la planta dada por detección de fallas incipientes, fallas en el sistema, políticas de mantenimiento, que determinan cambios en la planta y que determinan el uso de la misma, imponiendo cambios en su operación.

5.2.1 Modelado de las Dinámicas internas

Esta idea se basa en la consideración de que la dinámica de un sistema puede ser representada mediante la concatenación de un conjunto de dinámicas parciales, donde el punto final en una de las dinámicas en el punto inicial en otra en la cual se solapa tal como se ve en la figura 5.1.

La concatenación de regiones transformada en un sistema discreto la llamaremos comportamiento del sistema físico y puede ser representada como un autómata cuyas transiciones solo se dan entre regiones que se solapan tal como se muestra en la figura 5.2.

La dinámica dentro de cada una de las regiones anteriormente mencionadas, depende entonces de las características del sistema físico y del controlador asociado. El controlador condiciona la evolución del sistema internamente. La dinámica en cada uno de las regiones es dada por:

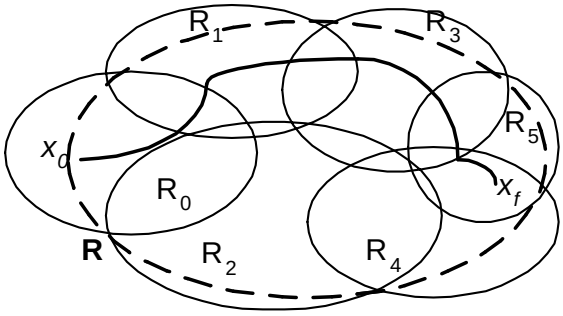


Figura 5.1: Comportamiento de un sistema físico

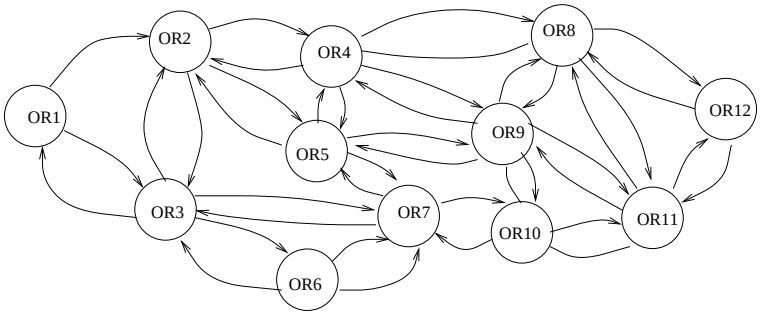


Figura 5.2: Comportamiento de un sistema físico

$$\dot{x^*} = f_{\omega_i}(x^*, u_{\omega_i}),\ i \in I \subset N, \tag{5.1}$$

donde

$\omega_i in \Omega$ es un símbolo del alfabeto que describe la dinámica del sistema
$x \in \Re$ representa el espacio de estados del sistema
$u_i \in U$ son la secuencia de controles aplicados al sistema

De la dinámica interna se establece el conjunto de regiones de operación, tal como se hace referencia en el capítulo 4, y cuales son los eventos que hace salir un sistema de una región a otra región tal como se muestra en la figura 5.3

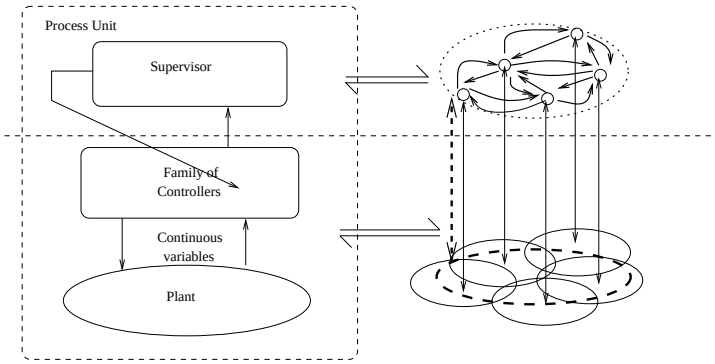


Figura 5.3: Relación entre el modelo de comportamiento y su control

La composición del sistema con su controlador genera los cambios entre las regiones de operación, y a su vez modifica la estructura del controlador

Ejemplo

Utilizando el ejemplo de los tres tanques mostrado en el capítulo anterior, vamos a construir un modelo total del sistema y su controlador. Para el modelado de la parte continua asumiremos que el vaciado del tanque asumiremos que el flujo de salida es constante. Tomemos cada uno de los tanques y lo modelamos separadamente; cada tanque se considera que tiene 10 estados discretos que son: *vacío – llenando, vacío – vaciando, bajo – llenando, bajo – vaciando, normal – lenando, normal – vaciando, alto – llenando, alto – vaciando, lleno – llenando, lleno –vaciando*, con situaciones anormales como *vacío y lleno*. La dinámica discreta se muestra en la figura 5.4.

El sistema que describe la dinámica de un tanque es el siguiente

$$\dot{x} = \begin{cases} 0 & \text{si } x = 0 \text{ y vaciandose} \\ -a_1 & \text{si } 0 < x < 1 \text{ y vaciandose} \\ 1 - a_1 & \text{si } 0 < x < 1 \text{ y llenandose} \\ 0 & \text{si } x = 1 \text{ y llenandose} \end{cases}$$

Los eventos que aparecen en el sistema son $a, c, 1, 2, 3, 4, 5, 6, 7, 8$, de los cuales son controlables a, c. Los estados marcados son $b-v,\ b-l,\ n-v,\ n-l,\ a-v,\ a-l$, los estados no aceptables son $v-v,\ l-l$. El supervisor controla la conducta del sistema abriendo y cerrando la alimentación al tanque dependiendo del evento que sea detectado. Para un solo tanque, el funcionamiento del supervisor se da en la figura

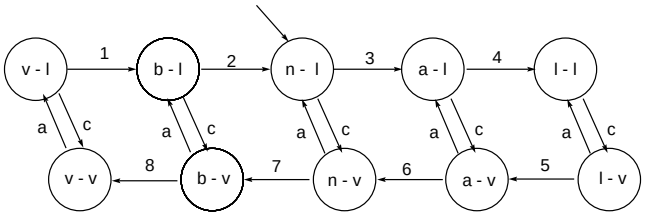


Figura 5.4: Discretización de la dinámica de un tanque

Capítulo 6

Conclusiones

El esquema de control jerárquico en sistemas híbridos permite la optimización de procesos sin la necesidad de desarrollar y mantener complejos modelos matemáticos que exigen grandes capacidades computacionales para su operación tanto fuera de líinea como en línea.

El uso de la infraestructura existente y de los esquemas de control convencionales, tipo PID, facilita la interacción del esquema de control propuesto con el operador, quien utiliza los parámetros e indicadores funcionales tradicionales en la industria.

La utilización de un sistema de control híbrido permite aprovechar al máximo las capacidades de los sistemas de control existentes en las instalaciones de procesos continuos extendiendo su vida útil y maximizando la rentabilidad de las inversiones realizadas, incluyendo la ejecución de estrategias de optimización para mejorar el rendimiento y la eficiencia de la Planta.

El nivel de automatización alcanzado abarca inclusive la predicción de los estados de operación de la Planta, dada una condición de operación, de forma de tomar previsiones en las unidades funcionales del proceso, así como para advertir a otros procesos con las cuales exista interacción, con el objetivo de reducir las perdidas por reproceso o producto fuera de especificación.

Las mayores dificultades encontradas están en la verificación del modelo global del sistema. Paquetes computacionales utilizados en la verificación de protocolos pueden ser utilizados con este propósito. Otra vía se logra mediante el uso de paquetes de simulación que utilicen tanto la parte continua como la parte discreta. El SIMNON es uno de estos paquetes al igual que el MATLAB.

Bibliografía

[1] P. Antsaklis, M. Lemmon, and J. Stiver. Modeling and design of hybrid control systems. In *IEEE Mediterranean Symposium on New Directions in Control & Automation*, pages 440 – 447, Malele-Chania, Crete, Greece, June 19-22 1994.

[2] Pannos Antsaklis and Kevin Passino, editors. *An Introduction to Inteligent and Autonomous Control*. Kluwer, Norwell, MA, 1993.

[3] Albert Benbeniste. Real–time systems design and synchronous programming. In *Proc. ECC 91. European Control Conference*, pages 512–521, Grenoble, France, July 2-5 1991.

[4] A. Benveniste, M. Le Borgne, and Le Guernic. *The Signal Approach.*, pages 230–254. Lecture notes in Computer Science. Vol 736. Springer Verlag, Berlin, New York, 1993.

[5] M. S. Branicky. *Studies in Hibrid Systems: Modelling, Analysis and Control*. PhD thesis, MIT, 1995.

[6] Michael S. Branicky, Vivel S. Borkar, and Sanjoy K. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE Transactions on Automatic Control*, 43(1):31–45, 1998.

[7] Y. Brave and M. Heymann. On optimal attraction in discrete event processes. *Information Sciences*, 67:245–276, 1990.

[8] Y. Brave and M. Heymann. On stabilization of discrete event processes. *International Journal of Control*, 51(5):1101–1117, 1990.

[9] R. Brockett. Hybrid systems in classical mechanics. In IFAC., editor, 13^{TH} *Triennial World Congress, S. Francisco USA. IFAC.*, pages 473–476, 1996.

[10] P Caspi. Model of discrete event systems in computer science. In *Proc. ECC 91. European Control Conference*, pages 503–511, Grenoble, France, July 2-5 1991.

[11] E. Chacón, E. Dapena, and F. Szigeti. Industrial automation by use of vasid environment. In *ISAS'95*, pages 6 – 12, Baden-Baden, Alemania, August 1996.

[12] Edgar Chacón, Gisela De Sarrazin, and Ferenc Szigeti. Pseudo dynamics hybrid systems. *Nonlinear Analysis, Theory, Methods & Applications*, 30(4):2533–2537, 1997.

[13] Edgar Chacón, Ferenc Szigeti, and Oscar Camacho. Integral automation of industrial complexes based on hybrid systems. *ISA Transactions*, 35(4):305 – 319, 1996.

[14] E. Dapena, C. Perdomo, and E. Chacón. Planning batch processes sequences in hierarchical hybrid systems. In *ISAS'96*, pages 759 – 765, Orlando, Florida, July 1996.

[15] René David. Grafcet: A powerful tool for specification of logic controllers. *IEEE Trans. on Control Systems Technology*, 3(3):253 – 268, 1995.

[16] A. Desphande and P.Varaiya, editors. *Viable Control in Hybrid Systems.* Lecture Notes in Computer Science, Vol 999 Springer-Verlag, New York, 1995.

[17] K. Forsman. Hybrid control systems and comprehensive grobner bases. In Lake Buena Vista., editor, *Proc. of the 33^{RD} Conf. on Decision And Control.*, pages 4092–4097, 1994.

[18] R. Grossman and R. Larson. An algebraic approach to hybrid systems. *Theoretical Computer Science*, 138:101–112, 1995.

[19] D. Harel. Statecharts: a visual formalism for complex systems. *Journal of Science of Computer Programming*, 8(1):231–274, 1987.

[20] D. Harel, A. Pnuely, J.P. Schmidt, and R. Sherman. On the formal semantics of statecharts. *IEEE LICS, Logic in computer science*, pages 54–64, 1987.

[21] C. Hoare. *Communicating Sequential Processes.* Englewood Cliffs. Prentice Hall, NJ, 1985.

[22] J. Hooman. *A Compositional Approach to the Design of Hybrid Systems.*, pages 121–148. Lecture Notes in Computer Science. Vol 736. Springer-Verlag., Berlin, New York, 1993.

[23] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison Wesley, Reading, MA, USA, 1977.

[24] B. Jakubzyk. Existence and uniqueness of realizations of non lynear systems. *SIAM J, on Control and Optimization*, 18:455–471, 1980.

[25] Wolf Kohn, Anil Nerode, Jeffrey B. Remmel, and ALexander Yakhnis. Viability in hybrid systems. *Theoretical Computer Science*, 138:141–168, 1995.

[26] M. M. Labinaz and R. Bayoumik. Modeling and control of hybrid system: A survey. In *Triennial World Congress IFAC*, pages 293–303, San Francisco, USA, July 1996.

[27] E. Lapalus, S. G. Fang, C. Rang, and R. van Gerwen. Manufacturing integration. *Computers in Industry*, 27:155–165, 1995.

[28] Michel Lemmon and Panos J. Antsaklis. Inductively inferring valid logical models of continuous-state dynamical systems. *Theoretical Computer Science*, 138:201–210, 1995.

[29] Bengy Lennartson, Michael Tittus, Bo Egardt, and Stefan Petterson. Hybrid systems in process control. *IEEE Control Systems*, pages 45–56, October 1996.

[30] A. Michel and L. Hou. Modeling and qualitative theory for general hybrid dynamical and systems. In *Preprints of IFAC Conference on Control of Industrial Systems.*, volume 1, pages 173–182, Belfort, France, 1997.

[31] R. Milner. *A Calculus of Communicating System*, volume 92. Lecture Notes in Computer Science, New York, 1980.

[32] Anil Nerode. Linear automaton transformations. *Proc. Amer. Math. Soc.*, (9):541–544, 1958.

[33] J. S. Ostroff. Systematic development of controllers for real–time discrete event systems. In *Proc. ECC 91. European Control Conference*, pages 522–531, Grenoble, France, July 2-5 1991.

[34] K. Passino. Bridging the gap between conventional and intelligent control. *IEEE Control Systems Magazine*, pages 12–18, June 1993.

[35] Ph. Peleties and R. Decarlo. Analysis of a hybrid system using symbolic dynamics and petri nets. *Automatica*, 30:1421–1427, 1994.

[36] Juan Pimentel. *Communications Networks for Manufacturing.* Prentice Hall, Englewood Clifs., 1990.

[37] P. Ramadge and W. M. Wonham. The control of discrete event systems. *Proc of IEEE: Especial Issue on Discrete Event Systems*, 77(1):81–97, Jan 1989.

[38] P. J. Ramadge and W. M. Wonham. Supervisory control of a class of discrete-event processes. *SIAM Journal of Control and Optimization*, 25(5):206–230, 1987.

[39] H. Roesler and L. Pierson. Open communications based on international standards. *Control Engineering*, 39(5), 1992.

[40] T. Roska. *Analog Events and Dual Computing Structure Using Analog and Digital Operators*, pages 225–238. Number 103 in Lecture Notes in Control & Information Sciences. Springer Verlag, Berlin, New York.

[41] Salomaa. *Formal Languages.* Academic Press, Boston, 1973.

[42] Arturo Sanchez. *Formal Specification and Synthesis of Procedural Controllers for Process Systems.* Number 212 in Lecture Notes in Control and Information Sciences. Springer Verlag, New York, 1996.

[43] G. Saridis. On the theory of intelligent machines: A comprehensive analysis. *International Journal of Intelligent Control and Systems*, 1(1):3–14, 1996.

[44] S. Sastry, G. Meyer, J. Lygeros, and D. Godbole. Hybrid system in air traffic control. In *IEEE Control and Decision Conference*, pages 1478–1483, 1995.

[45] R. Sengupta and S. Lafortune. An optimal control theory for discrete event systems. *SIAM Journal on Control and Optimization*, 36(2):488–541, 1998.

[46] Luz Solé and Ferenc Szigeti. Reachability of a class of coupled discrete event systems. In *Proc IFAC/IFORS Symposium on Large Scale Systems: Theory and Applications*, pages 386–389, Beijing, China, 1992.

[47] R. S. Sreenivas and B. H. Krogh. On condition/event system with discrete state realizations. *Discrete Event Dynamic Systems: Theory and Applications*, pages 209–236, 1991.

[48] Ferenc Szigeti, Edgar Chacón, and Gisela De Sarrazin. Hybrid dynamic systems: An application to integral automation of complex systems. In Universidad del Zulia, editor, *Segundo Coloquio sobre Ecuaciones Diferenciales y sus Aplicaciones*, pages 63–83, 1995.

[49] J.Ñ. Tsitsiklis. On the control of discrete-evnt dynamical systems. *Mathematics of Control, Signals, and Systems*, 2(1):95–107, 1989.

[50] W. M. P. van der Aalst. Putting high-level petri nets to work in industry. *Computers in Industry*, 25:45–54, 1994.

[51] T. J. Williams, P. Bernus, J. Brosvic, D. Chen, G. Doumenigts, L.Ñemes, J. L. Nevis, B. Vallespir, J. Vliestra, and D. Zoetekouw. Architectures for integrating manufacturing activities and enterprises. *Computers in Industry*, 32:111–139, 1994.

[52] Y. M. Willner and M. Heymann. Language convergence in controlled discrete-event systems. *IEEE Transactions on AUTOMATIC CONTROL*, 40(4):616–627, 1995.

[53] W. M. Wonham. *A control theory for discrete-event systems*, pages 129–169. Advanced Computing Concepts and Techniques in COntrol Engineering. Springer Verlag, 1988.

[54] W. M. Wonham and P. J. Ramadge. On the supremal controllable sublanguage of a given language. *SIAM Journal of Control and Optimization*, 25(3):635–659, 1987.