# Modelado y Simulación del Control Supervisorio para Sistemas Holónicos de Producción Continua

Carlos Arturo Parra Ortega

Tesis Doctoral

presentada a la ilustre

Universidad de Los Andes

para optar al título

de Doctor en Ciencias Aplicadas

Director: Dr. Edgar Alfonso Chacón Ramírez

17 Abril de 2009

# Resumen

En este documento presentamos una propuesta del esquema para validar el enfoque holónico aplicado a Sistemas de Producción Continua y por lotes en cuanto a su estructura, comportamiento dinámico y supervisión interna, mediante la comprobación de su comportamiento utilizando diversas técnicas de modelado y simulación que representan sistemas de eventos discretos. Para llevar a cabo esta validación se requiere describir el comportamiento dinámico de una Unidad de Producción cuya arquitectura de automatización se basa en el modelo de referencia holónico PROSA, en forma de sistemas dinámicos discretos y continuos acoplados. Este comportamiento dinámico proporciona las pautas para sintetizar un Sistema de Control Supervisorio utilizando Sistemas a Eventos Discretos (DES), que modificará la configuración del sistema a supervisar de acuerdo a una conducta predeterminada por la misión principal de producción. Para simular la gestión de holones autónomos y su sistema supervisor se recurrió a la tecnología de agentes, cuyo comportamiento se especifica utilizando lógica de primer orden y las reglas de comportamiento se obtienen a partir de modelos multi-resolucionales extendidos. El sistema multi-agente representa al componente de razonamiento de los holones, e interactúa con un modelo discreto y continuo que representa al proceso industrial y las propiedades de los recursos utilizados, los cuales a su vez representan el componente físico de los holones.

Como resultado de esta investigación se logró validar el comportamiento de la unidad de producción continua concebida como un sistema holónico, con lo que se puede avanzar en aspectos relacionados con supervisión de unidades de producción. Además obtuvimos una metodología de modelado y simulación que combina aspectos de sistemas híbridos, síntesis de sistemas supervisores, sistemas multiagente, y el formalismo de simulación de eventos discretos (DEVS). Otro aporte que se obtuvo de este trabajo consistió en la arquitectura de implementación basada en el Lenguaje Unificado de Modelado, con el fin de avanzar en dirección a una implementación directa de sistemas

holónicos supervisados, al igual que la extensión de modelos multi-resolucionales que permiten proyectar reglas de supervisión a reglas de comportamiento de agentes.

# Abstract

In this document we present a proposal for validate the holonic approach applied to Continuous and Batch Production Systems into structure, dynamical behavior and internal supervision, by means of the behavior testing using some modeling and simultion techniques that represent discrete-event systems. In order to carry out this testing it is needed to describe the dynamical behaviour of a Production Unit whose automation architecture is based on holonic reference model named PROSA, by way of coupled discrete and continuous dynamical systems. This dynamical behavior provides the model to synthesize a Supervisory Control System using Discrete-Event Systems (DES), that will modify the configuration of the system being controlled in accordance with a behavior predetermined by the main production mission. In order to simulate the management of autonomous holons and its supervisor system was made use of agent technology, whose behaviour is specified using first-order logic and the behaviour rules are obtained starting from extended multi-resolutional models. The multi-agent system represents the reasoning component of the holons, and works with a continuous and discrete model that represents the industrial process and the properties of the resources being utilized, which represent the physical component of the holons.

As result of this research work we achieved to test the behaviour of the continuous production unit conceived as holonic system, which allows us to advance in topics related with supervising production units. Moreover we obtained a modeling and simulation methodology that combines features of hybrid systems, supervisory systems synthesis, multi-agent systems and discrete-event system formalism (DEVS). Another contribution obtained from this work consisted in the implementation architecture based on Unified Modeling Language, in order to advance toward a direct implementation of supervised holonic systems, like the extension of multi-resolutional models that allow project supervision rules to agent behavioral rules.

# Agradecimientos

Expreso mi más sincera gratitud al profesor Edgar Chacón Ramírez, tutor de mi tesis, quien fue la guía permanente que tuve desde que manifesté el interés en estudiar el Doctorado en Ciencias Aplicadas. El profesor Edgar colaboró en la elaboración de la propuesta inicial, supervisó mi paso en la etapa de escolaridad, en la revisión de mis documentos para presentar en seminarios, presentación en eventos y publicaciones, al igual que en la elaboración de la tesis doctoral, así como el apoyo que me expresó en los momentos más difíciles de mis estudios doctorales, y ante todo por su sincera amistad profesada en todos estos años.

También agradezco sobremanera al profesor Eliézer Colina Morles, quien fue un apoyo importantísimo en la etapa de publicación de artículos, brindando consejos para la experimentación del problema a publicar, redactando en inglés las versiones finales de los artículos, así como sus guías e indicaciones para sortear la etapa más difícil del doctorado, como lo es la publicación en revista indexada.

Otros agradecimientos van para el grupo de estudio del área de automatización industrial, computación y simulación: mis compañeros Miguel Indriago, Lenín Becerra, José Andrickson, Fanny Rodríguez, Alfredo Ramos, Juan Fréitez, Alexy Sánchez, Oscar Amaury y Germán Zapata. Sus discusiones y acompañamiento sobre automatización, sistemas a eventos discretos, agentes, simulación y otros aspectos fue un apoyo a los pasos dados durante mis estudios. Especialmente agradezco a Germán y Óscar por la revisión del documento de la tesis y sus consejos para la redacción, así como a Pascal Blanc, por sus aportes en la discusión sobre modelado conceptual y UML. De la misma manera, agradezco al profesor Juan Cardillo por sus consejos para presentar el examen de candidatura.

También expreso mis agradecimientos a la Universidad de Pamplona y su honorable Consejo Superior, por su apoyo institucional a mi proyecto profesional, y en particular

a los profesores Oscar Fiallo y Antonio Gan, quienes me brindaron su apoyo permitiéndome desplazar desde Pamplona a Mérida para recibir las asesorías en la elaboración del documento de tesis doctoral después que me reintegré a labores académicas.

A mi esposa María Liliana y a mi hijo Carlos Daniel muchas gracias por la paciencia y comprensión que han tenido conmigo al permitirme que sacrificara parte del tiempo que debiera compartir con ellos, para dedicarlo en culminar estos estudios.

Finalmente, expreso mi infinito agradecimiento al Ser Supremo.

# Índice general

# Índice de figuras

# Índice de tablas

# Capítulo 1

# Introducción

Las Organizaciones Industriales han aprovechado los aportes tecnológicos de las tecnologías de información, automatización y microelectrónica, entre otras, logrando implantar esquemas organizativos, mantener sus mercados e integrar sistemas de información y control para satisfacer a una demanda de bienes y servicios cada vez más exigente. El entorno de comercio globalizado que rodea a estas organizaciones las obliga a ser más competitivas, debido a los cambios en las tendencias del consumidor y en normatividad industrial principalmente. Lo anterior implica que estas organizaciones tengan ciertas capacidades necesarias para mantener sus mercados, como por ejemplo la adaptabilidad a cambios en las especificaciones de los productos, optimización del uso de sus recursos para obtener a tiempo su producto final, entre otras. La tendencia general es hacia una sincronización entre los procesos productivos y la cadena logística de la empresa, para reaccionar con rapidez ante los cambios en los factores externos que rodean a la organización, como lo son las necesidades del cliente y las regulaciones estatales, así como en factores internos a la organización, como la ocurrencia de fallas en el equipamiento o cambios en la disponibilidad de insumos, por ejemplo. Existen tres tipos de procesos de producción en manufactura industrial, de acuerdo a la clasificación de Fisher [39], según la naturaleza en que desarrollan sus

actividades, y complementado luego por Chacón y DeSarrazin [19], cuyas definiciones se establecen a continuación:

- Proceso de Manufactura, donde varios insumos se ensamblan para obtener un producto terminado. La función principal de la producción es la maquinación de piezas, y el ensamble de ellas para obtener un producto. Por medio de la manipulación de las piezas se obtienen productos intermedios, los cuales a su vez son ensamblados en un proceso subsiguiente. Por ejemplo, fabricación de autos ensamblando sus piezas, partes automotrices.

- Proceso por Lotes, es un proceso que se realiza para una cantidad de producto, en un tiempo determinado. Al inicio del proceso se recibe materia prima, y luego de aplicar energía y operaciones de mezclado se obtiene el producto terminado, luego se reinicia el proceso o se inicia un proceso diferente con el mismo equipamiento. Ejemplos de estos procesos se encuentran en la producción de lácteos, petroquímica, azúcar, industria siderúrgica, industria del papel, entre otros. Charlotta Johnsson [57] considera el proceso por lotes como el más complejo puesto que combina características tanto continuas como discretas.

- Sistemas de Producción Continua. Estos procesos de transformación funcionan de manera permanente. Así que se producen bienes o servicios continuamente. Como ejemplo están las plantas de tratamiento de agua, el gas, la generación de electricidad, la extracción de petróleo, entre otros procesos. La mayoría de Sistemas de Producción Continua deben mantenerse en funcionamiento a pesar de que hayan cambios en el flujo de entrada de insumos, o funcionamiento degradado del equipamiento. Por tal motivo, algunos autores lo consideran como procesos de alta complejidad.

Sincronizar el proceso productivo y la cadena logística implica que el proceso industrial se debe *monitorear, controlar y supervisar*, a la vez que se aseguran insumos,

gestionan equipos y personas, y se analizan los resultados de producción y gestión. Factores externos a la Organización Industrial influyen sobre el proceso productivo, por ejemplo las condiciones del mercado que establecen nuevos requerimientos de producción, lo que trae como consecuencia cambios en el proceso productivo, en los recursos a utilizar, en los métodos de producción, y en otros aspectos. No solo factores externos afectan la programación de la producción, también influyen factores internos como la disponibilidad de insumos, cambios en recursos humanos, restricciones en los costos de producción, cambio en las capacidades y condiciones del equipamiento industrial, entre otros. La *coordinación* de los elementos y procesos de una organización industrial exigen que se cumplan los requerimientos de producción, además de que el proceso productivo sea seguro y óptimo bajo las nuevas condiciones. Se requiere por lo tanto de *flexibilidad y adaptación* en las empresas industriales.

## 1.1. Sistemas de Producción Continua

La aplicación de la investigación propuesta en este trabajo doctoral está orientada hacia una clase especial de Sistemas Industriales: los Sistemas de Producción Continua (SPC, de aquí en adelante). Estos SPC tienen como característica fundamental que su funcionamiento no puede deternerse, salvo aquellas situaciones donde se presentan fallas en su equipamiento, o exista un mantenimiento programado de recursos físicos. Su funcionamiento en general es de la siguiente manera: dado un flujo continuo de material, se les aplican procesos de transformación según un método determinado, y luego el producto obtenido se envía inmediatamente al cliente utilizando una red de distribución, o se almacena en depósitos para distribuirse posteriormente. El material de insumos también puede tomarse de un depósito por medio de otra red de distribución. La logística de la empresa debe facilitar que siempre se pueda aplicar el método de producción a los insumos. Este SPC se utiliza cuando la demanda es

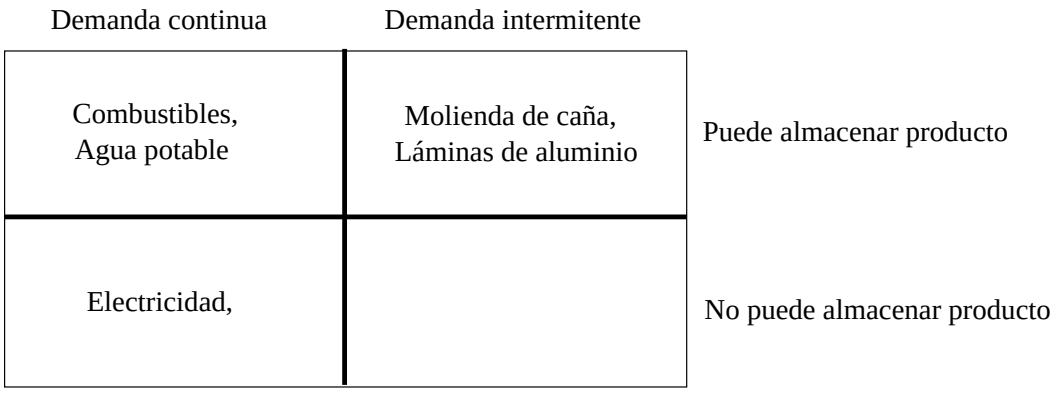


Figura 1.1: Clasificación de los Sistemas de Producción Continua

sostenida (continua) y previsible en el corto plazo. La planta de producción puede dedicarse a producir sin pausa, a menos que ocurra una condición de falla grave en el equipamiento industrial, o un cambio súbito en la especificación del producto, como consecuencia de un cambio en la demanda. Estos cambios en la especificación del producto ocasionan a su vez cambios en el método de producción y por lo tanto cambios en la ejecución de este método.

Para clasificar a los SPC pueden aplicarse dos criterios: uno de ellos consiste en la manera en que se hace disposición del producto final, y el otro está relacionado con la naturaleza de la demanda. En la figura 1.1 se propone una matriz de clasificación de los SPC según estos criterios. El primer criterio clasifica la disposición del producto final en aquellos SPC que pueden almacenar su producto final en alguna clase de depósitos para luego distribuirlo, y aquellos que deben entregar inmediatamente lo que han producido por medio de alguna red de distribución. En el primer grupo se ubica a la industria petroquímica, extracción y distribución de crudo, tratamiento de agua, entre otras, mientras que en el segundo grupo están las empresas generadoras y transportadoras de energía eléctrica. Los SPC también se pueden clasificar de acuerdo a la naturaleza de la demanda: la demanda debe satisfacerse siempre, como lo es la electricidad, o una demanda intermitente, como lo es el jugo de caña, el proceso de petróleo o la petroquímica.

Las condiciones del mercado, que exigen competitividad a las empresas, y el cambio continuo en los requerimientos de los clientes ocasionan cambios en los estilos de producción y configuración de las organizaciones de producción. Para el caso de los SPC, los cambios se observan en las especificaciones del producto, algunas veces emitidas por reglamentación (como lo son las características del agua tratada, por ejemplo), o por necesidades del cliente (rendimiento de un combustible, por ejemplo). Como consecuencia, la planificación de la producción, la asignación de recursos, programación de actividades (scheduling) y los mecanismos de control secuenciales y centralizados no son lo suficientemente flexibles para responder a estos nuevos estilos de producción que cambian continuamente y a las variaciones que se producen continuamente en los requerimientos del producto. Por tal motivo, una cualidad importante en los SPC y que se requiere de implementar es la *Reconfigurabilidad*, que permita una fácil extensión y reconfiguración de los sistemas de producción para atender demandas cambiantes. Otra cualidad es la *Reactividad*, la cual permite que un sistema reaccione ante cambios externos e internos adaptando su funcionamiento para las nuevas condiciones. Además, para ser competitivos, los SPC deben adoptar arquitecturas abiertas que integren sus actividades con proveedores y clientes en amplias redes de cadena de suministros, y que permitan tal interacción de una forma rápida y económica. Así mismo, esta característica se debe presentar al interior de sus *Unidades de producción*, de tal manera, se requiere que el SPC tenga cualidades de *cooperación*, tanto entre sus elementos de producción interna, como los que interactúan con el exterior de la organización.

El flujo de producto terminado está condicionado por la capacidad de la planta de producción y sus características; las especificaciones del producto; la calidad de la materia prima; la demanda, que se puede considerar continua; y algunos eventos, como fallos en elementos de la planta. A pesar de que cambien todos los aspectos mencionados anteriormente, la planta debe cumplir con su objetivo de producción,

sin que importen las nuevas condiciones. Esta cualidad se denomina *Estabilidad*. El objetivo del control de estos sistemas consiste en asegurar el comportamiento estable de ciertas variables en el tiempo, introduciendo cambios en la configuración de las unidades de producción, y proporcionar la flexibilidad necesaria para atender cambios en la demanda, o fallas en los recursos sin tener que detener el proceso. El punto de partida para esta investigación consiste en la necesidad de establecer la validez de un mecanismo de supervisión que proporcione autonomía a un sistema industrial concebido bajo una arquitectura de automatización que exhiba las cualidades antes mencionadas, y de establecer si este mecanismo es factible de implementar bajo las arquitecturas de computación y telecomunicaciones actuales.

## 1.2. Motivación y Objetivos

Actualmente, existe un gran interés académico el área de los Sistemas de Manufactura Inteligente (IMS) y en particular los Sistemas Holónicos de Manufactura, y se han desarrollado numerosos proyectos de investigación aplicados a arquitectura, modelado, control e integración con tecnologías emergentes. Morel *et al* [78] elaboraron en el año 2007 un estudio sobre los aspectos relevantes en los sistemas de manufactura de próxima generación, donde dos de estos aspectos están relacionados con esta propuesta doctoral: estos son los IMS y Sistemas de Manufactura Confiable *DMS*. Los problemas identificados y pronósticos sobre tendencias en investigación elaborado por estos autores con respecto a IMS y DMS están relacionados con la carencia de herramientas o plataformas para probar y validar desarrollos de IMS aplicados a problemas reales de producción, así como obtener aplicaciones más reactivas en el piso de planta, capaces de detectar fallas a tiempo. Además estos autores establecen a los sistemas de manufactura holónica y los sistemas multi-agente como paradigma de modelado para añadir inteligencia a un sistema de manufactura. De hecho, los sistemas multi-agente

se han utilizado en manejo de cadenas de suministros, programación de actividades, integración de aplicaciones, entre otros tópicos [108]. Estos paradigmas de modelado son parte central de este trabajo doctoral, donde se valida el diseño de mecanismos de supervisión de sistemas de manufactura holónica para otorgarles mayor autonomía.

Sin embargo, puesto que los sistemas de manufactura son a gran escala y complejos, es difícil abordar el problema de control de procesos distribuidos y complejos de una manera integral. Por tal motivo, este trabajo aborda el control supervisorio de un sistema de manufactura holónica desde el punto de vista del modelado, de la implementación con tecnología de agentes y la validación utilizando la simulación multi-agente de un sistema a eventos discretos, con lo que establece una solución parcial al problema de *confiabilidad* de los sistemas de manufactura (DMS).

El problema que se quiere resolver por medio de esta investigación consiste en establecer un método para validar el funcionamiento del control supervisorio para un sistema de producción continua, concebido bajo el paradigma holónico. Este SPC puede consistir de varias unidades de producción "autónomas" conectadas entre sí por medio de redes que distribuyen material y redes de telecomunicaciones para enviar y recibir información, de manera que la naturaleza del proceso a controlar además de continua es distribuida. La naturaleza distribuida de los sistemas de producción actuales necesita de sistemas de control y toma de decisiones distribuida, por lo tanto, las arquitecturas de sistemas basados en agentes y Sistemas Holónicos propuestas por Bongaerts, Wyns, Shen *et al* [8, 110, 72] deben de tenerse en cuenta para el diseño del entorno de control adecuado de la planta. La interacción distribuida significa que los sujetos internos de la empresa pueden intercambiar información entre sí y con otros sujetos externos, y tomar decisiones operativas sin tener que recurrir a la dirección de la organización para planificar un proceso de producción. Una propiedad importante de los sistemas holónicos consiste en su capacidad de ser auto-similares por medio de la composición. Es decir, un holón complejo que represente a una organización industrial

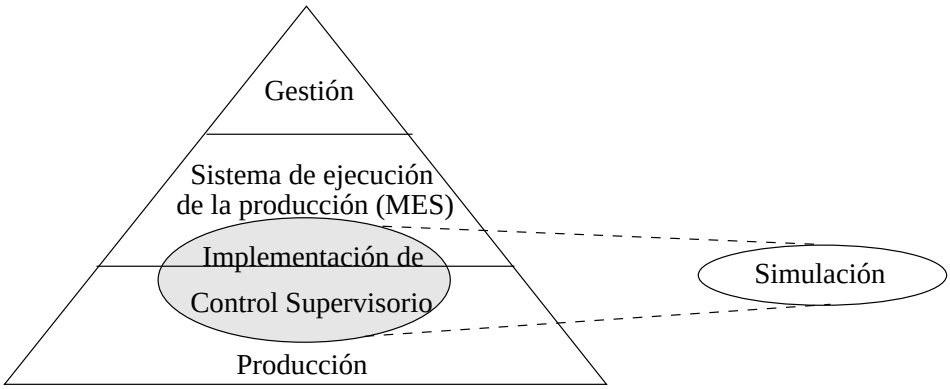


Figura 1.2: Ubicación de la investigación propuesta

se puede obtener a partir de la composición de holones más simples, que representan a unidades de producción, y estos a su vez se componen de otros holones más básicos. Esta última propiedad es más difícil de reproducir en los sistemas multi-agente [45].

Según Jennings [55], la tecnología de agentes es la forma más natural para desarrollar este entorno de fabricación, y por lo tanto se requiere de integrar esta tecnología a las unidades de producción autónoma, para brindarles capacidades de gestión y de cooperación. Por otro lado, se necesita una metodología para validar el diseño de estas unidades de producción que son autónomas y gestionadas por agentes. Esta metodología debe basarse en el modelado y simulación de sistemas de producción, ya que una implantación directa es muy costosa. Dentro de la estructura de automatización industrial, esta investigación estaría ubicada como una implementación de control supervisorio, tal y como aparece en la figura 1.2, y se evaluará la validez de esta arquitectura por medio de la simulación de varios ejemplos académicos, los cuales representarán a una o varias unidades de producción interactuando para lograr una meta común de producción.

### 1.2.1. Hipótesis de trabajo

Lo hipótesis que se plantea en esta tesis doctoral consiste en determinar si es factible validar la conducta de un sistema de producción continua compuesto por unidades

autónomas denominadas holones, las cuales están supervisadas. Para ello se requiere derivar un modelo de simulación a partir de la descripción de la organización y objetivos de producción que reproduzca la conducta anteriormente mencionada y que incorpore elementos conceptuales de sistemas holónicos, sistemas discretos y continuos, supervisión de procesos y sistemas multi-agente. Además se desea determinar cómo sería la implementación de este esquema de supervisión utilizando tecnologías adecuadas para implementar en un sistema distribuido. En la figura 1.3 se puede apreciar el esquema que guía los pasos de este trabajo: las teorías y técnicas que son punto de partida están en letra normal, mientras que los aportes a obtener en esta investigación están resaltados en **negrita**. Las flechas indican de qué manera se van a utilizar los aportes teóricos y tecnológicos, para obtener el resultado esperado y comprobar así la hipótesis de trabajo.

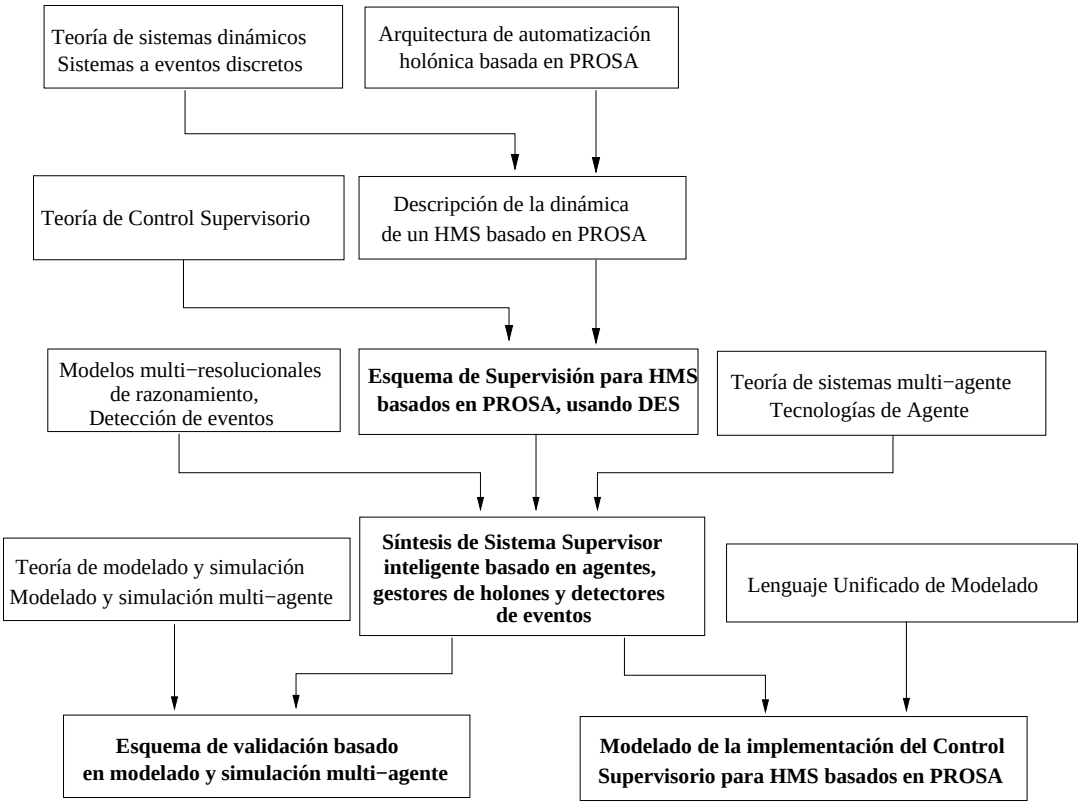


Figura 1.3: Insumos, actividades y resultados esperados de esta investigación

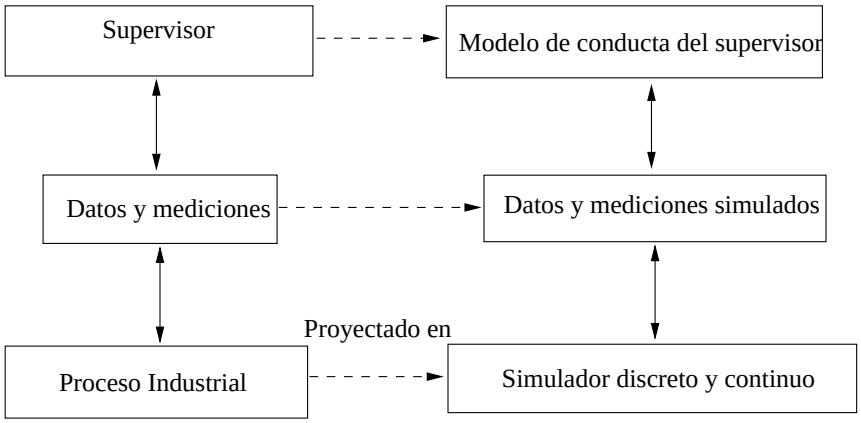


Figura 1.4: Esquema general de la metodología de validación

A partir de una arquitectura de automatización basada en sistemas holónicos, y la especificación de un proceso industrial como un sistema dinámico, se obtiene la descripción de la dinámica de un sistema holónico inspirado en la filosofía PROSA. Esta descripción, junto con las teorías de Control Supervisorio son el punto de partida para establecer un esquema de supervisión de sistemas holónicos. Para brindarle mayor inteligencia y capacidad de operar en un entorno distribuído, se ha pensado implementar un Sistema de Supervisión y Control a la manera de un sistema multi-agente, utilizando tecnologías de agente, con esquemas de razonamiento basado en modelos multi-resolucionales y a su vez se extenderán estos modelos para disponer funcionalidades de detección de eventos y razonamiento a varios niveles. Una vez se haya obtenido el Sistema de Supervisión inteligente, se va a proponer un modelado conceptual para implementar el Control Supervisorio en una arquitectura orientada a objetos que interactúe con las clases basadas en el estándar PROSA. Además, y este es uno de los fundamentos de este trabajo doctoral, se va a proponer un esquema de validación del Sistema de Supervisión basado en el modelado y simulación multi-agente.

En la figura 1.4 se puede apreciar como se plantea esta validación, donde el proceso industrial se proyecta a un simulador discreto y continuo (híbrido), y el supervisor lo

controla, proyectando un modelo de conducta basado en sistemas a eventos discretos y reglas, que modifica el comportamiento del proceso industrial, representado también con sistemas a eventos discretos. Este esquema general será desarrollado en el capítulo siete de este documento.

### 1.2.2. Objetivos específicos

- Expresar el comportamiento de un Sistema de Manufactura Holónica (HMS) inspirado en la arquitectura de automatización PROSA, como un Sistema Dinámico Acoplado (SDA), que combina tanto la dinámica continua del proceso físico de transformación con la dinámica discreta del estado de los recursos y el proceso. Así se podrán aplicar las teorías de control supervisorio para sistemas discretos, y la detección de eventos para proyectar de variable continua a discreta.

- Especificar una estructura de información que obtenga el estado de un recurso y de un proceso, dada una estructura de Holón Básica. La conducta de este Holón se considerará como una composición de Sistemas de Eventos Discretos (DES). Cada vez que hay un cambio de estados en el sistema discreto, se generan órdenes para que el sistema físico cambie alguno de sus parámetros bajo los cuales opera.

- Para controlar la conducta del Holón mencionado anteriormente, sintetizar un Supervisor para que controle la ocurrencia de eventos que lleven a estados no deseados del proceso a controlar. El mencionado Supervisor es otro sistema de eventos discretos recibe los eventos de la imagen del proceso del Holón básico de producción, y envía los comandos de habilitación/deshabilitación de eventos para esa imagen del proceso.

- Combinar la teoría de los Agente Racional y modelos de razonamiento para

sintetizar un Sistema que implemente al Supervisor, detectores de eventos, que gestione a los Holones básicos de la Unidad de Producción. Para poder supervisar un proceso de producción continuo se requiere proyectar los valores de las variables de estado hacia un sistema de eventos discreto que representa al estado de la planta, y un mecanismo que permita la detección de eventos.

- Definir una metodología basada en *modelado y simulación* que permita estudiar la supervisión de procesos industriales que tienen lugar en unidades de producción a partir de un modelo que represente a una planta industrial en sus tres aspectos fundamentales según la clasificación de van Gigch: sujetos, objetos y conceptos [43]. Donde los sujetos serían los actores que intervienen en la gestión de la unidad de producción, los agentes acoplados a cada supervisor, los detectores de eventos, entre otros; los conceptos serían la representación del estado del proceso, las metas de producción, la especificación del producto, las reglas de supervisión del proceso, las leyes de control y el modelo físico del proceso de producción; y los objetos físicos son los insumos utilizados, equipamiento industrial, energía y controladores. Esta metodología debe tener en cuenta la interacción de varios tipos de representación de sistemas: el modelo físico del proceso industrial, el sistema discreto que representa el estado de la planta y del supervisor, con la secuencia de eventos resultante que se va a comparar con un plan establecido por la coordinación de unidades de producción, en forma de secuencia de eventos y tiempos; y el sistema de decisiones, a cargo de los actores que gestionan la unidad de producción.

- Validar por medio de simulación a un grupo de unidades de producción concebidas bajo la arquitectura de automatización holónica, interactuando entre sí para ejecutar un proceso de producción real.

### 1.2.3. Aportes esperados

Como contribución de este trabajo doctoral se pueden esperar los siguientes aportes: a) Obtener la descripción de la dinámica híbrida de un conjunto de unidades de producción continua descritas por medio de la arquitectura de referencia PROSA. Esta dinámica consta de un sistema a eventos discretos cuyos cambios de estado se deben a la interacción de variables continuas, como las que se presentan en un proceso industrial. b) Proporcionar autonomía a una unidad de producción holónica por medio del mecanismo de supervisión de su conducta, y gestión basada en su misión global de producción. c) Establecer una especificación basada en el lenguaje de modelado UML que proporcione las bases de una posterior implantación en una arquitectura computacional y de comunicaciones. d) Obtener una metodología de validación que no solo permita evaluar la consistencia del mecanismo de supervisión del holón, sino también generar una secuencia de trayectorias que luego se pueda llevar a los gestores y controladores de la unidad de producción.

## 1.3. Estado del arte

Para una mejor ubicación de este trabajo dentro del área de automatización industrial y sistemas holónicos, el estado del arte se dividió en cuatro grandes grupos: modelado de sistemas de manufactura basados en holones, mecanismos de síntesis de sistemas supervisores, aplicados a los sistemas de manufactura, implementación de sistemas de manufactura con agentes inteligentes y metodologías de modelado y simulación de sistemas industriales. Los trabajos mencionados de aquí en adelante cubrirán alguno de estos aspectos, y algunos de ellos abarcarán dos o más tópicos. Como ejemplo, este trabajo doctoral abarca tres tópicos: la síntesis de un sistema de control supervisorio aplicada a un sistema holónico de producción continua, su implementación con tecnología de agentes, y su validación por medio del modelado y simulación a eventos

discretos.

En cuanto a la concepción de un sistema productivo se han desarrollado varias investigaciones sobre sistemas de manufactura holónica, donde uno de los trabajos más exitosos es el de una arquitectura de referencia basada en holones denominada **PROSA**, desarrollada en la Universidad de Lovaina [10]. Esta arquitectura está compuesta por tres holones principales: Holón de producto, Holón de recursos, y Holón de órdenes, además de un Holón de staff para proporcionar planificación jerárquica. Otra arquitectura para sistemas holónicos, basada en agentes y bloques funcionales se propuso por la universidad de Keele por Deen y Fletcher [63]. En esta arquitectura, un Holón de fabricación se establece como el acoplamiento de un control inteligente que contiene a los agentes y actúa como la "cabeza" y el sistema de procesamiento, compuesto por bloques funcionales los cuales actúan como la "base". Una planta de producción puede verse así como un conjunto de holones de fabricación. Otra arquitectura de sistemas holónicos es la propuesta de Maturana y Norrie de la Universidad de Calgary, denominada Metamorph II [72]. Esta propuesta consiste en una arquitectura holónica basada en agentes, cuya estructura es cambiante. Una arquitectura multicapa la proponen el Centro Germano de Automatización junto con la Universidad de Rumania, y se denomina INTERRAP [75].

En cuanto a la supervisión de un sistema holónico, existen varios trabajos que describen la dinámica de la conducta de un HMS, como el de Gouyòn en 2004 [48], que establece un esquema basado en autómatas para describir el comportamiento dinámico de un Sistema Holónico, y sintetiza luego un controlador para un proceso, escogiendo una trayectoria determinada. Igual descripción la establecen Chacón, Besembel y Hennet en 2003 [18], aunque plantean la descripción de la dinámica de un sistema holónico utilizando redes de Petri, y plantean el desarrollo de un mecanismo de supervisión basado en sistemas a eventos discretos. Leitão [67] propone una arquitectura holónica adaptativa denominada ADACOR, para el control de procesos de

manufactura, basándose en redes de Petri temporizadas. En resumen, la Supervisión de sistemas holónicos en su totalidad no se ha propuesto, sino la del Holón de Control, quien actúa como un Holón tipo Staff, considerando una arquitectura derivada de PROSA. Otros trabajos están orientados hacia el control supervisorio, sin importar la arquitectura de automatización utilizada. Entre los más recientes están el de Hellgren [52], orientado a la implementación de control supervisorio basado en eventos discretos, enfocado a sistemas de manufactura flexible.

### 1.3.1. Comparación con propuestas similares

Para comparar esta tesis doctoral con otros trabajos y propuestas similares, se establecen los siguientes aspectos de comparación:

- Segmento de aplicación: consiste en indicar hacia que tipo industria está orientada la investigación, o si es de tipo general. Este trabajo está dirigido principalmente hacia los Sistemas de Producción Continua.

- Concepción del sistema de producción: es decir, si se adopta un enfoque jerárquico de automatización, o un plano, o un enfoque holónico o una mezcla de arquitecturas de diseño. Para este caso, el SPC se considerará bajo la filosofía de Sistemas Holónicos.

- Orientación de la descripción de la conducta: indica si la propuesta se centra hacia el producto, hacia el recurso, o hacia que otro aspecto del proceso productivo.

- Mecanismos para describir la conducta del Sistema de Manufactura: indica la forma en que se considera la descripción de la conducta dinámica del Sistema de Producción, y el mecanismo de representación de esta. Para ello este trabajo estará orientado hacia la dinámica de cada recurso del sistema productivo, asi

como la evolución del proceso, vista como la interacción entre un método de producción y la ejecución de dicho método. En este documento la descripción del proceso físico a controlar se realiza por medio de sistemas híbridos, proyectando a sistemas de eventos discretos para integrar con la descripción del sistema holónico.

- Implementación del razonamiento: este aspecto consiste en describir la manera como se van a implementar las reglas para la gestión de los holones que intervienen en el proceso, el razonamiento del supervisor, detector y actuador. La tecnología de agentes, y los sistemas multiagente hacen parte de esta implementación.

- Descripción del proceso supervisado: indica como se va a supervisar la dinámica de la conducta, a partir de su descripción utilizando Sistemas de Eventos Discretos (DES). Además, describe el proceso de síntesis del Supervisor, así como su proyección hacia la lógica de los agentes que son el componente de información de cada Holón.

- Evaluación: indica como se va a validar el modelo diseñado. En este caso se realizará el modelado y simulación basado en DEVS, para medir el desempeño del esquema de supervisión de un proceso continuo.

- Implementación computacional: como implementar este diseño en un sistema real, utilizando Tecnologías de Información y Comunicación. Diseño en UML, programación en java, sistema multiagente utilizando la librería JADE, interfaz con PLC, interfaz HMI.

Entre los trabajos similares desarrollados en esta área se tiene la tesis de control sobre sistemas de manufactura de Gouyon [48], el control orientado a agentes propuesto por Simão para proporcionar agilidad a los sistemas holónicos [103], ADACOR, la

arquitectura adaptable de control propuesta por Leitão [67], sistemas holónicos propuestos por Adam [2], Arquitectura de Referencia PROSA, por Jo Wyns, Bongaerts con sus scheduling para sistemas holónicos [8, 110], un sistema de monitoreo basado en PROSA y desarrollado por Blanc [5], la metodología ANEMONA, para aplicar sistemas multi-agente a sistemas holónicos desarrollado por Giret [45], entre otras aplicaciones. A continuación la comparación con estos trabajos, aspecto por aspecto.

**Segmento de Aplicación.** Los trabajos desarrollados por Wyns, Gouyon, Simao y Leitao están orientados a Sistemas de Manufactura Discreta, como aquellas industrias donde se reciben piezas, se ensamblan y operan, y a partir de ahí sale el producto terminado, previa orden de trabajo. El trabajo de Adam está orientado hacia Sistemas de Información Distribuido, La propuesta de Bongaerts está orientada a la programación de la producción (scheduling) y control en línea de Sistemas Holónicos, al igual que el trabajo de Blanc [5]. La propuesta de Giret está orientada hacia la programación de actividades en un sistema holónico. La propuesta del trabajo doctoral de este documento se orienta hacia sistemas de producción continua.

**Concepción del Sistema de Producción.** Varios de los trabajos ya mencionados conciben al Holón como la unidad de producción fundamental, y basados en la arquitectura PROSA, como lo son los de Bongaerts, Wyns, Gouyon y Blanc. La disertación de Simao se basa en PROSA también, pero adiciona un Holón de Control, para proporcionar conocimiento experto sobre control de procesos. Además considera a los atributos y métodos de los holones como sub-holones. Leitao propone una organización holárquica propia, Adam se centra en el efecto Janus para organizar Holones, Blanc extiende la definición de Holones PROSA, mientras que Giret se basa en PROSA, pero define un agente abstracto como base del razonamiento de un holón para implementar la capacidad de auto-similaridad en sistemas multi-agente. Esta tesis doctoral está orientada hacia una filosofía holónica inspirada en la arquitectura

de referencia PROSA, definiendo un holón básico (indivisible) y luego la conformación de unidades de producción a partir de la composición de holones básicos. No se considera al holón de Staff.

**Descripción de la Conducta del Sistema de Producción.** La conducta del Sistema Holónico de Manufactura puede analizarse desde varias perspectivas: desde el punto de vista de los recursos, del producto terminado, del proceso en ejecución, o una combinación de cada uno de ellos. También es criterio para analizar esta conducta el método utilizado para representarla, como las redes de Petri, las máquinas de estado finito, esquemas basados en lógica, entre otros. Las propuestas que analizan la conducta del sistema de producción orientándose hacia el producto son las de Wyns y Gouyon, este último proponiendo una supervisión del producto, al igual que los recursos utilizados. Simao establece una propuesta orientada al recurso, en especial al equipamiento, y propone una supervisión del Holón Recurso. El trabajo propio describe la conducta del sistema de producción orientándose en el proceso y el estado de los recursos. El método para representar esta conducta se basa en sistemas a eventos discretos, bien sea redes de Petri o máquinas de estado finito, donde algunos de sus estados corresponden a la proyección de valores de variables continuas asociados a una condición de operación.

**Síntesis del Supervisor.** Los trabajos que proponen explícitamente la supervisión o el control en HMS son los de Bongaerts, el cual propone un mecanismo basado en redes de Petri temporizadas, estocásticas, condicionales y controladas. Adam también utiliza el enfoque de redes de Petri, en particular las redes paramétricas, para representar interrupciones. Leitao propone una conducta dinámica del holón producto, modelada con redes de petri temporizadas y de perturbaciones. Simao establece el uso de un modelo de conducta de cada equipamiento específico generado por grafcet, y generando a su vez código C o C++. Por el contrario, Gouyon propone un es-

quema basado en autómatas. Vale la pena destacar que la mayoría de propuestas mencionadas no conciben la supervisión como un control de alto nivel, sino que se orientan directamente a controlar el comportamiento de cada holón. El método de síntesis depende de la manera en que se considere la descripción de la conducta de la unidad de producción. Este y los anteriores criterios se pueden apreciar en la tabla 1.1.

Tabla 1.1: Comparación de esta propuesta con otras similares en los aspectos de aplicación, concepción, conducta y supervisión

| Propuesta | Aplicación | Concepción | Conducta | Supervisión |
|---|---|---|---|---|
| Bongaerts, 1998 | Scheduling y control de HMS | Holones PROSA | | Supervisor basado en redes de Petri temporizadas, estocásticas, condicionales y controladas |
| PROSA (Wyns), 1999 | Sistemas de manufactura | Holones PROSA | Orientada al producto | |
| Adam, 2000 | Sistemas de Información distribuidos | Se centra en el "efecto Janus", para organizar holones | | Redes de Petri paramétricas, para representar interrupciones |
| Gouyon, 2004 | Sistemas de manufactura | Holones PROSA | Orientada al producto | Utiliza autómatas y algoritmos de síntesis |
| Blanc, 2006 | Programación de actividades en HMS | Holones PROSA extendidos | | |
| Giret, 2005 | Programación de actividades y control de HMS | Basado en PROSA | | |
| Leitao, 2004 | Sistemas de manufactura | Propone una holarquía propia para organizar los Holones. | Orientada al holón Producto | Conducta dinámica modelada con redes de petri temporizadas y de perturbaciones |
| Simao, 2005 | Sistemas de manufactura | Holones PROSA mas un holón de Control, que es de tipo Staff. Atributos y métodos son sub-holones | Orientada al recurso | Uso de un modelo de conducta de cada equipamiento específico generado por grafcet, y código C o C++ |
| Hellgren, 2002 | Sistemas de manufactura flexible | Independiente de la arquitectura | Orientada a los recursos y actividades. Utiliza redes de Petri modulares | Se basa en restricciones en vector de marcación, utilizando la exclusión mutua generalizada. Introduce monitores, asumiendo que todas las transiciones son controlables |
| Koffman, 2006 | Control digital de Sistemas continuos | Independiente de la arquitectura | Ecuaciones diferenciales y Sistemas a eventos discretos | Nueva aproximación al control regulatorio |
| Parra, 2008 | Sistemas de producción continua | Basado en Holones PROSA | Sistemas descritos en variable continua y discreta. Redes de Petri o Máquinas de Estado Finito. Orientada al proceso y a los recursos | Método tradicional de Ramadge y Wonham, o de Moody y Antsaklis, dependiendo de la manera de considerar la conducta de la unidad de producción. |

**Implementación del razonamiento.** La mayoría de propuestas estudiadas establecen como fundamento para implementar a los HMS la tecnología de agentes, difiriendo en la forma en que programan sus capacidades de razonamiento y representación del conocimiento. Así, Simao establece el Holón de Control orientado a los Agentes, utilizando las reglas de tipo condición-acción (reactivas), las cuales implementan soft-holones. Leitao también utiliza agentes, con reglas de tipo condición - acción. Adam utiliza agentes cognitivos. Wyns y Bongaerts no proponen explícitamente el uso de agentes, sino que profundizan el paradigma orientado a objetos para proponer su arquitectura de referencia. Blanc utiliza agentes que utilizan algoritmos heurísticos para planificar actividades. La tesis de Giret diseña un agente abstracto, que luego se instancia para producir las funcionalidades de cada tipo de holón. Para obtener este agente recurrió a la ingeniería de software y una mezcla de refinamientos top-down y bottom-up de desarrollo.

**Evaluación.** Como método para validar el esquema de supervisión diseñado, algunas propuestas se basan en la simulación, como la de Simao, donde desarrolla una herramienta simuladora de plantas, y luego se hace holonificación. El kernel del simulador está basado en redes de Petri, con capacidad para crear eventos. Leitao recurre a una validación formal, junto con una plataforma de emulación. Adam también recurre a la simulación con redes de Petri. Wyns utiliza un simulador de eventos discretos, basado en Arena, al igual que Blanc.

**Implementación.** Giret, Blanc y Leitao utilizan un entorno multi-agente JADE, basado en java. Como herramienta de modelado está UML, utilizado por Wyns, Bongaerts, Adam y Gouyon. Implementación directa al piso de planta es la de Gouyon y Simao, el cual desarrolla una herramienta propia: Analytice. Leitao también propone una descripción XML del producto y de la organización industrial, para interactuar con su esquema Adacor. Este criterio y los anteriores se resumen en la tabla 1.2.

Tabla 1.2: Comparación de esta propuesta con otras similares en cuanto a los aspectos de Razonamiento, implementación y validación

| Propuesta | Razonamiento | Evaluación | Implementación |
|---|---|---|---|
| Bongaerts, 1998 | No propone el uso de agentes de manera explícita | - | Diseño basado en UML |
| PROSA | No propone el uso de agentes de manera explícita | Simulador de eventos discretos basado en ARENA | Diseño basado en UML |
| Adam, 2000 | Agentes cognitivos | Simulación basada en redes de Petri | Diseño basado en UML |
| Gouyon, 2004 | Sistemas de manufactura | - | Diseño basado en UML, con implementación directa al piso de planta |
| Blanc, 2006 | Agentes que utilizan algoritmos heurísticos para planificar actividades | Simulador de eventos discretos basado en ARENA | Herramienta multi-agente basada en Java |
| Giret, 2005 | agente abstracto, que luego se instancia para producir las funcionalidades de cada tipo de holón | - | Herramienta multi-agente basada en Java |
| Leitao, 2004 | Agentes, con reglas de tipo condición - acción | Validación formal, con implementación directa o emulación | Implementa a ADACOR en JADE. Propone una descripción XML del producto y de la organización industrial, para interactuar con su esquema Adacor |
| Simao, 2005 | Holón de Control orientado a los Agentes, utilizando reglas de tipo condición-acción, las cuales implementan soft-holones | Desarrolló herramienta de simulación propia, Analytice, la cual simula la operación de plantas basándose en redes de Petri, con capacidad para crear eventos | Implementación directa al piso de planta |
| Hellgren, 2002 | No lo trata | Simula la marcación de redes de Petri | Implementación directa de los supervisores en PLCs. |
| Koffman, 2006 | No lo trata | Diseña simuladores DEVS cuantificados | Propone nuevos métodos de integración numérica como simuladores de variable discreta y continua |
| Parra, 2008 | Agentes implementan gestionadores de holones, más las funciones de supervisión y detección de eventos | Simulación multi-agente basada en DEVS y DESS, con detección de eventos | Diseño del control supervisorio de un HMS basado en UML. |

El documento de la tesis está organizado como sigue: en el capítulo dos se menciona el marco de arquitectura de automatización donde se va a desarrollar la investigación. Los aspectos relacionados con la descripción de la dinámica de la conducta de un sistema holónico se muestran en el capítulo tres. En el capítulo cuatro se aborda el tema de la supervisión del sistema holónico, y los mecanismos de síntesis necesarios para obtener el supervisor. La implementación del Sistema de Control Supervisorio como un sistema multi-agente, así como la gestión de holones de producción con tecnología de agentes se plantea en el capítulo cinco, mientras que en el capítulo seis se establecen las pautas del modelado conceptual del sistema holónico supervisado, para facilitar su implantación dentro de una arquitectura de computación y telecomunicaciones. En el capítulo siete se plantea el método de validación basado en simulación multi-agente a eventos discretos, y en el capítulo ocho se enuncian las conclusiones derivadas de este trabajo doctoral.

# Capítulo 2

# Sistemas Holónicos de Producción

En esta sección se va a profundizar sobre la estructura de las arquitecturas holónicas de producción, en particular sobre la arquitectura de referencia PROSA [10], la cual se tomará como punto de partida para describir un Sistema de Producción Continua (SPC) y proponer así su supervisión, lo cual le brindaría autonomía para regular su comportamiento con respecto a su misión de producción.

La sección empieza con un recuento de los enfoques existentes en cuanto a arquitecturas de automatización. Posteriormente presenta las definiciones de holón y holarquía, luego describe brevemente la arquitectura de referencia PROSA en sus aspectos fundamentales, también muestra como es la organización de un Holón típico bajo esta arquitectura, su relación con los sistemas multi-agente y la comparación con otras arquitecturas se describirán posteriormente, así como el procedimiento para concebir una organización existente como una composición de holones.

## 2.1. Arquitecturas de Automatización

Una organización que contiene sistemas de producción se concibe por medio de una *estructura organizativa* que le permita llevar a cabo sus labores de producción, la

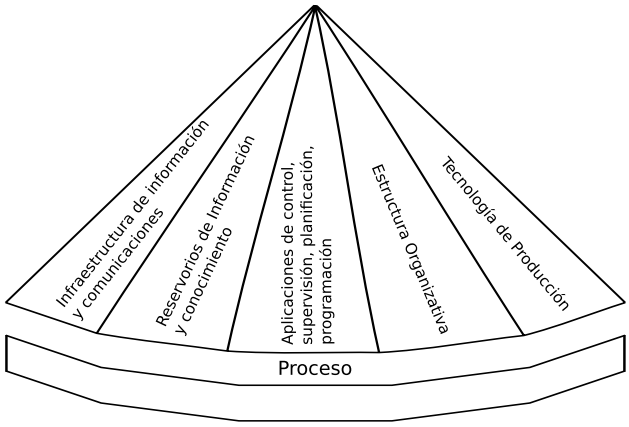


Figura 2.1: Relaciones entre la organización y las tecnologías de automatización

cual corresponde a un modelo lógico que facilite su descripción, gestión, supervisión y control; y a su vez se pueda proyectar hacia una arquitectura de informática y telecomunicaciones que permita su operación. El proceso productivo se apoya en los sistemas de toma de decisiones y en los sistemas de información del proceso a nivel de piso de planta, que corresponden a las aplicaciones de control. Estas aplicaciones también se implementan en una infraestructura de información y comunicaciones, y utilizan repositorios de información. Este proceso productivo se lleva a cabo intercambiando material e información, lo que origina la dinámica total del sistema de producción. Las relaciones entre la estructura organizativa, tecnologías de automatización, aplicaciones de control y el proceso productivo se muestran en la figura 2.1.

Existen tres enfoques para diseñar una estructura organizativa: *estructuras jerárquicas de producción, estructuras heterárquicas y estructuras holárquicas.*

### 2.1.1. Arquitecturas jerárquicas

Varias arquitecturas de referencia se han propuesto para implementar automatización industrial, basándose en cada uno de estos enfoques. Las arquitecturas de tipo jerárquico son centralizadas, y su estructura general es piramidal, como la que se observa en la figura 2.2. Derivadas de la **pirámide de automatización** propuesta

por la ISO, para sistemas de control de procesos continuos, se han definido varias arquitecturas que proyectan los sistemas de control hacia los sistemas de información. Una de ellas es CIMOSA [23], la cual es una arquitectura abierta para definir, especificar e implementar sistemas de control automatizado. Otro trabajo que vale la pena mencionar es la arquitectura de referencia PERA, propuesto por la Universidad de Purdue [27]. Esta arquitectura considera tres subsistemas: el equipamiento físico, el recurso humano y los sistemas de información y control, así como un ciclo de vida de cada proyecto productivo. La sociedad ISA ha propuesto los estándares jerárquicos SP88 para procesos por lotes, y SP95 para petroquímica y procesos continuos. A medida que evoluciona la tecnología de información y comunicaciones también lo hace la forma en que se monitorean y controlan los procesos industriales. Un ejemplo de ello es la aplicación de la tecnología publicador-suscriptor a las redes de campo para monitorear procesos. La incorporación de programas inteligentes al piso de planta (shopfloor) permite a su vez detectar fallas y eventos, permitiendo una mejor gestión y supervisión del proceso.

En general, este enfoque totalmente centralizado puede requerir una gran cantidad de información y comunicación para procesos productivos muy complejos, ya que las decisiones se toman a nivel de gestión (el mas alto) y el estado del proceso se conoce también a ese nivel, lo que puede ocasionar retardos cuando se aplican respuestas ante algún evento inesperado. Otra desventaja de estas arquitecturas consiste en que si se requiere un cambio en la configuración de la planta industrial se necesita detener el proceso para llevar a cabo tal cambio.

Recientemente se han propuesto implementaciones sobre industrias con estructuras centralizadas, llevándola a un sistema reconfigurable como el que propone el grupo de Automatización de Plantas basado en Sistemas Distribuidos (PABADIS) [92]. El proyecto PABADIS se basa en agentes móviles, y está diseñado para funcionar sobre *Sistemas de Producción de Manufacturas*, y en estructuras centralizadas. Esta

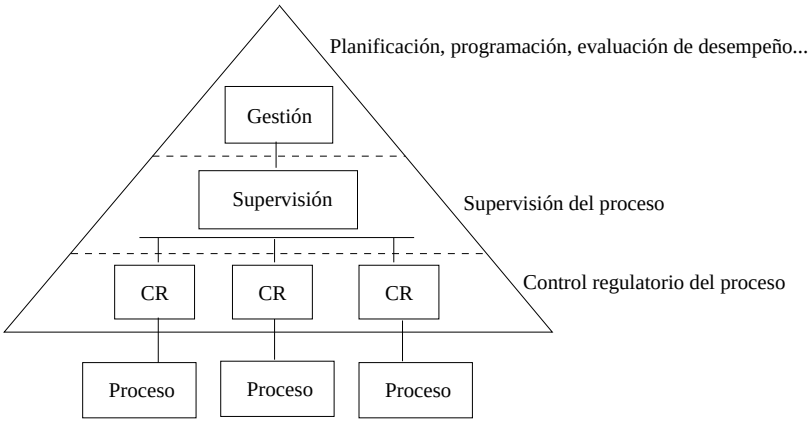


Figura 2.2: Enfoque jerárquico de integración.

propuesta propone el reemplazo del Sistema de Ejecución de Manufactura (MES) y el Control Supervisorio por grupos de agentes de software, en cuanto al manejo de la producción a nivel de proceso. Este proyecto se basa en sistemas distribuidos, y utiliza modelos y software orientado a objetos, para describir y ejecutar tareas de automatización. Divide al MES en dos partes: una centralizada conectada al ERP, y una descentralizada, que interactúa con los agentes. Con este proyecto se pretende resolver las limitaciones de los sistemas MES y SCADA en cuanto a flexibilidad, complejidad y robustez.

### 2.1.2. Arquitecturas heterárquicas

Otro enfoque de automatización que resuelve los problemas derivados de la organización jerárquica es el de la *heterarquía*, una organización plana donde cada unidad de producción basa su funcionamiento en la negociación con otras unidades de producción. La implementación de este enfoque está basado en componentes de computación autónomos denominados *agentes*. En estas arquitecturas una labor importante a realizar es la *coordinación* entre diversas unidades, para lograr una meta común. Uno de los aspectos positivos es que se gana en reactividad ante cambios súbitos en las condiciones del equipamiento o en la misión de producción, además, evita incoheren-

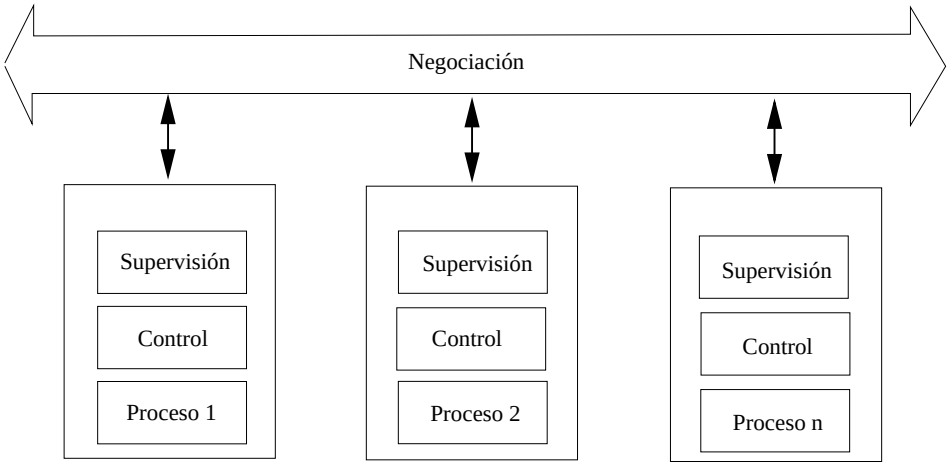


Figura 2.3: Organización de la producción como una heterarquía

cia entre formas de operación, ya que antes de operar cada unidad de producción se pone de acuerdo con las demás sobre las actividades a ejecutar. Un esquema de organización heterárquica se puede observar en la figura 2.3, donde se muestran varias unidades de producción que negocian su capacidad de producción para lograr una meta común.

### 2.1.3. Arquitecturas holárquicas

El último enfoque de automatización considera a la organización como una jerarquía especial denominada *holarquía*, donde su componente fundamental es una unidad autónoma denominada *Holón*. El concepto general de Holón se propuso por primera vez en 1967, por Arthur Koestler [62], cuya definición se mostrará más adelante. Este concepto se aplicó inicialmente a los organismos vivientes y organizaciones sociales, y explica algunas propiedades de adaptabilidad y flexibilidad frente al cambio de condiciones del ambiente. El consorcio HMS [10] ha llevado estos conceptos al campo de la manufactura industrial, tratando de obtener los beneficios que las organizaciones holónicas proporcionan, como la estabilidad frente a perturbaciones, adaptabilidad y flexibilidad, entre otras.

Un esquema de enfoque holárquico de la producción se puede observar en la figura 2.4,

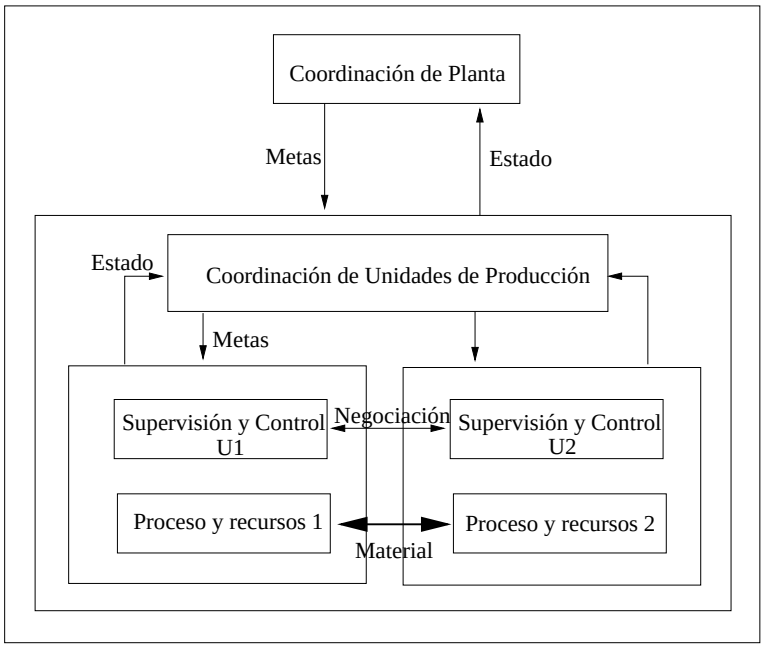


Figura 2.4: Esquema de organización holárquica

donde se muestran unas unidades de producción cuya misión es ejecutar y controlar un proceso determinado. Cada una de estas unidades aporta a la misión global del coordinador de producción su sub-producto, el cual puede ser insumo de otra unidad de producción, pertenecientes todas a una planta, con su correspondiente coordinador. La coordinación de estas unidades de producción puede llevarse a cabo de manera explícita, por medio de una unidad de coordinación, o cada unidad negocia recursos e información con las otras unidades, de acuerdo a su capacidad de producción.

Dado que nuestra propuesta doctoral se centra en las arquitecturas holónicas, es necesario definir en que consisten los holones, y como se concibe una holarquía. En la siguiente sección se van a establecer los conceptos que se van a utilizar en el resto del documento.

## 2.2. Definiciones de Holón y Holarquía

**Definición 2.1. Holón.** Arthur Koestler [62], fue el primero que acuñó este término en su libro "El Fantasma en la Máquina" hace 40 años, componiendo la palabra *Hol-on*, a partir del término de origen griego *Holos* que significa "todo", mientras que el sufijo *on* da a entender un significado de una partícula indivisible, como si fuera parte de algo. El significado general es que Holón se asimila a un objeto que es todo y parte a la vez, ya que desde el punto de vista de sus partes, un Holón es una entidad completamente autónoma, pero se considera como un objeto que compone a otros, desde un punto de vista de mayor jerarquía dentro de una organización. Se basa en dos postulados fundamentales: uno de ellos establece que un sistema complejo evolucionará a partir de un sistema simple si hay una forma intermedia estable, el sistema complejo resultante será jerárquico. El otro establece que los holones son simultáneamente un todo auto-contenido desde el punto de vista de sus partes subordinadas, y partes dependientes cuando se observan desde una instancia superior.

Una arquitectura general de un holón se propuso por primera vez en 1994 en el trabajo de Christensen [22], y se muestra en la figura 2.5. Esta arquitectura refleja los dos tipos de elementos que intervienen en un sistema de control de producción: los elementos de software junto con las diversas entidades físicas o dispositivos. La *capa de procesamiento físico* se compone de un elemento hardware que realiza una operación de fabricación, como por ejemplo ensamblado, con su respectivo elemento de control físico, que corresponde a un controlador que regula la operación del elemento físico. La parte física del holón requiere de un componente software que actúe como envoltorio del equipamiento hardware, y que se comunique con la *capa de procesamiento de información*. Esta capa contiene el componente de toma de decisiones, que representa el núcleo del razonamiento del holón y proporciona dos interfaces: la primera para la interacción con otros holones, y la segunda para la interacción con humanos.

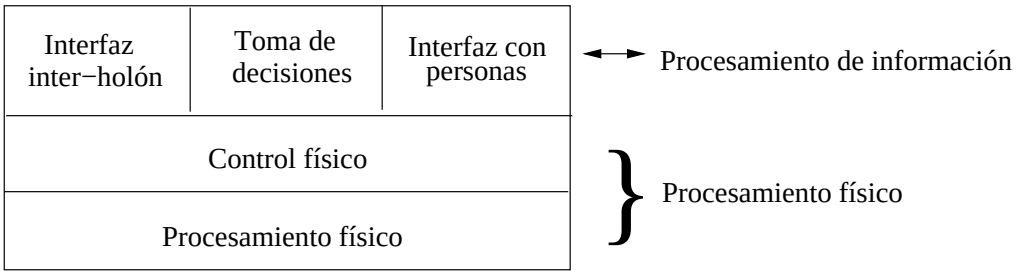


Figura 2.5: Arquitectura de un Holón de Producción

**Definición 2.2. Holarquía.** Bongaerts, Wyns *et al* [8, 110] definen la Holarquía como una estructura jerárquica compuesta por holones que cooperan para lograr un objetivo común. La Holarquía define las reglas básicas para la cooperación de los holones, y limita su autonomía en cuanto a que coloca su misión global como guía de las acciones de los holones que la conforman.

**Definición 2.3. Sistema de Manufactura Holónica (HMS).** Llevando el concepto de holones y holarquía al dominio de la industria, un Sistema de Manufactura Holónica es una holarquía compuesta por entidades autónomas y cooperativas denominadas holones. Esta holarquía integra todas las actividades de manufactura, desde procesar una orden de producción hasta su despacho, pasando por el diseño del producto, mercadeo y ejecución de la orden, tal y como la propone Wyns [110]. Cada holón corresponde a una parte identificable del sistema de manufactura, la cual a su vez se compone de otras partes, y hace parte de una unidad más compleja.

## 2.3. Estructura de un Holón

En el dominio de los sistemas industriales, el Holón se considera como un componente fundamental de un sistema de manufactura, entre sus atributos están la autonomía y la cooperación. Por autonomía se entiende que es la capacidad para manejar completamente un proceso en cuanto a establecer el estado del proceso y decidir sobre como controlarlo. Dentro de las funciones del holón está la de transformar, transportar y

almacenar objetos físicos, así como validar información sobre estos objetos y tareas en ejecución [110]. Como elementos del Holón se ha definido una parte física y otra para procesar información. Por lo tanto se tiene: *cabeza, cuello y cuerpo*, donde el cuerpo se relaciona directamente con el proceso a controlar, el cuello se relaciona con la red de campo que tiene la misión de obtener los datos para monitorear el proceso, y para transmitir los comandos a la parte física, y una "cabeza" donde se procesa información, se toman y comunican las decisiones que toma el Holón y se relaciona con otros holones y con personas. En la figura 2.6 se puede apreciar la estructura del Holón y su relación con las Tecnologías de Información (TIC). En la cabeza del Holón se encuentran las aplicaciones y estructuras de datos que permiten manejar esa información. El "cuello" es el elemento de interconexión que actúa como la interfaz entre la toma de decisiones y el procesamiento físico del Holón, los componentes de software que llevan a cabo las labores de medición del estado del proceso físico, la interfaz software del equipamiento físico, los controladores, los sistemas de coordinación y supervisión, los objetos que componen la imagen del proceso y la interfaz con las personas, junto con su respectiva red se consideran como el "cuello" del Holón. El "cuerpo" del Holón contiene los equipamientos físicos que permiten llevar a cabo la transformación del producto. Obviamente estos equipamientos deben disponder de algún mecanismo que informe de su estado al cuello del Holón, así como del estado de los recursos que está procesando. Por medio de la composición de holones se obtendrá un holón de mayor complejidad, de la misma manera como la composición de Unidades de Producción permite que se obtenga la Planta a controlar.

## 2.4. Sistemas holónicos y sistemas multi-agente

La relación entre los Sistemas Holónicos de Manufactura (HMS) y los Sistemas Multi-agente (SMA) es muy cercana, tanto que parecieran ser la misma cosa. De hecho, tanto

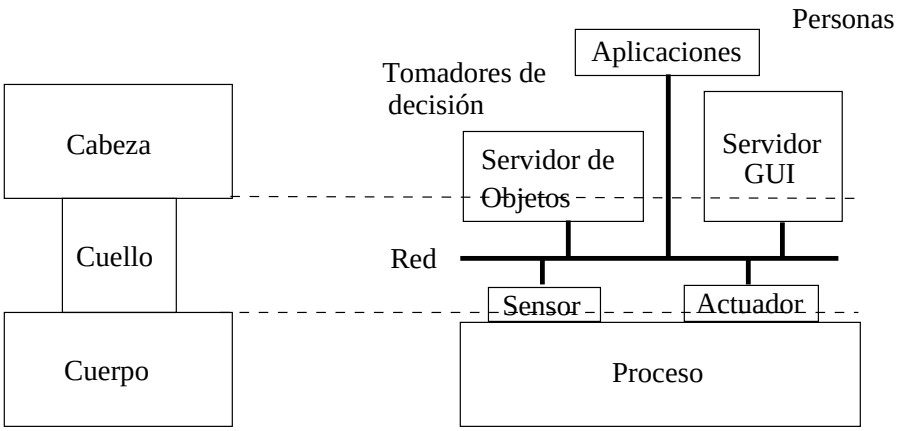


Figura 2.6: Estructura de un Holón, interacción con TIC

el Agente como el Holón comparten atributos: son abiertos, cooperativos y autónomos [63]. La mayoría de los HMS se han implementado utilizando tecnologías basadas en agentes. Lo que conviene aclarar es que los holones tienen un componente físico y uno de información, y este último se implementa utilizando tecnologías de agente. Giret [45] muestra en su tesis un estudio comparativo entre HMS y SMA: concluye que ambos tienen las mismas propiedades base de la comparación, solo que difieren en la motivación para la que son construidos. Los SMA son de propósito general, con gran cantidad de aplicaciones en diferentes áreas, mientras que los HMS están en el dominio de las organizaciones industriales flexibles. Existen algunas implementaciones para control de procesos industriales basados en agentes, reemplazando a los Sistemas de Ejecución de Manufactura (MES). La propuesta de PABADIS [92] es un ejemplo representativo de esta implementación. Por otro lado, algunas arquitecturas de referencia holónica explícitamente recurren al agente como parte fundamental en la estructura del holón (su cabeza, o implementación de estrategias de planificación de alto nivel, dentro de la propuesta de Deen y Fletcher, de la Universidad de Keele, y que se muestra en la figura 2.7 [41]. Esta estructura divide al holón en tres bloques funcionales: una base, donde se encuentran los proyectan los componentes software y hardware (este último es opcional, y se implementa por medio de un envoltorio

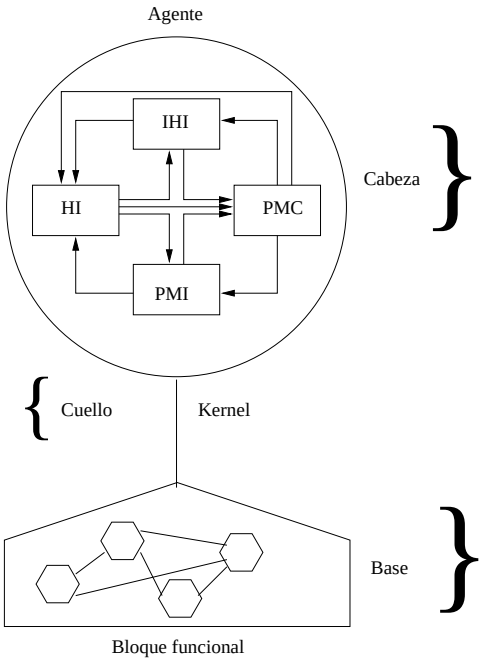


Figura 2.7: Estructura de un Holón, propuesta de Deen y Fletcher

software), un kernel, y la cabeza, la cual implementa un agente cuyos elementos son el Control Proceso/Máquina o PMC, Interfaz Proceso/Máquina o PMI, Interfaz Humana o HI, e Interfaz Inter-Holón o IHI. Estos componentes están relacionados entre sí, y permiten que los holones ofrezcan capacidades de control de manufactura.

Una arquitectura adaptativa basada en agentes es Metamorph, propuesta por Maturana, Shen y Norrie [72], la cual consiste en comunidades dinámicas de agentes que se agrupan en coaliciones temporales, implementando así el cambio organizacional estructural, el cual funciona a cuatro niveles: empresa virtual, sistemas distribuidos inteligentes, ingeniería concurrente y arquitectura inteligente de agentes. Los agentes pueden ser de dos tipos: *agente de recurso* y *agente mediador*. Un agente de tipo de Recurso se utilizan para representar dispositivos de manufactura (equipamiento), los agentes mediadores se utilizan para coordinar las acciones entre los agentes, tanto mediadores como de recurso. El mecanismo de negociación entre agentes se basa en la descomposición de tareas. Una comparación detallada entre Holones y Agentes fue desarrollada por Giret de acuerdo a sus propiedades [45]. Un resumen de esta

comparación se puede observar en la tabla 2.1.

De la misma manera en que se relaciona un Agente con un Holón individual se puede afirmar que existe una relación entre una Holarquía y un Sistema Multiagente. Según la propuesta de Kotak *et al* [63] una holarquía logra un objetivo del sistema, y esta holarquía se coordina en un dominio de cooperación (CD), el cual es implementado por el facilitador de directorio de un sistema multiagente. Un holón pertenece a una holarquía, y tiene un componente físico, un componente de información y un componente de cooperación. Un agente implementa el componente de cooperación, la cual es dispuesta por el dominio de cooperación CD, el cual a su vez también dispone de un componente de comunicación.

## 2.5. Arquitectura de Referencia PROSA para Sistemas Holónicos

El Sistema de Manufactura Holónica (HMS) es uno de los enfoques principales de un Sistema de Control de Manufactura, y la arquitectura de referencia PROSA es una de las principales implementaciones de un sistema HMS, ya que especifica en detalle como implementar sistemas de control de producción. PROSA deriva su nombre de los holones básicos que la conforman, estos son *Producto, Recurso, Orden y un holón StAff, que es opcional*, y fue diseñado por el grupo de investigación en automatización de la Universidad de Lovaina, en particular por los trabajos de Van Brussel, Wyns, Valckenaers, Bongaerts y Peeters [10, 8, 110]. Esta especificación puede servir o bien para diseñar un sistema de control de manufactura nuevo o para aplicar la arquitectura a un sistema ya existente, para analizarlo y estudiarlo. Este último uso se puede denominar *Holonificación*. Los tres tipos de holones básicos corresponden a tres aspectos diferentes de un proceso de manufactura: el manejo óptimo de los recursos los asume el holón Recurso, los aspectos tecnológicos relacionados con el producto

Tabla 2.1: Comparación de Holones y Agentes de acuerdo a sus propiedades [45]

| Propiedad | Holón | Agente |
|---|---|---|
| Autonomía | Sí | Sí |
| Reactividad | Sí | Sí |
| Proactividad | Sí | Sí |
| Habilidad Social | Sí. La interfaz humana es específica de cada Holón | Sí. La interfaz Humana se implementa por uno o varios agentes especializados |
| Cooperación | Sí. Los holones nunca rechazan de manera deliberada la cooperación con otro holón | Sí. El agente puede competir y cooperar. |
| Re-Organización | Sí. Por medio de Holarquías. | Sí. Jerarquías, heterarquías, entre otras. Las holarquías se pueden implementar utilizando federaciones de agentes, como facilitadores o mediadores. |
| Racionalidad | Sí. | Sí. |
| Aprendizaje | Sí. | Sí. |
| Benevolencia | Sí. | Sí. |
| Movilidad | Los holones raramente necesitarían de movilidad para la ejecución de sus tareas. | Sí. |
| Recursión. | Sí. | No existe ninguna arquitectura recursiva como tal, pero algunas técnicas se utilizan para definir federaciones que simularán los diferentes niveles recursivos. |
| Procesamiento físico y de información | Sí. La separación es explícita, el procesamiento físico es opcional. | No existe una separación explícita. |
| Actitudes Mentales | Sí. Los holones no necesitan razonar sobre sus propias actitudes mentales o aquellas de otros holones. | Sí. |

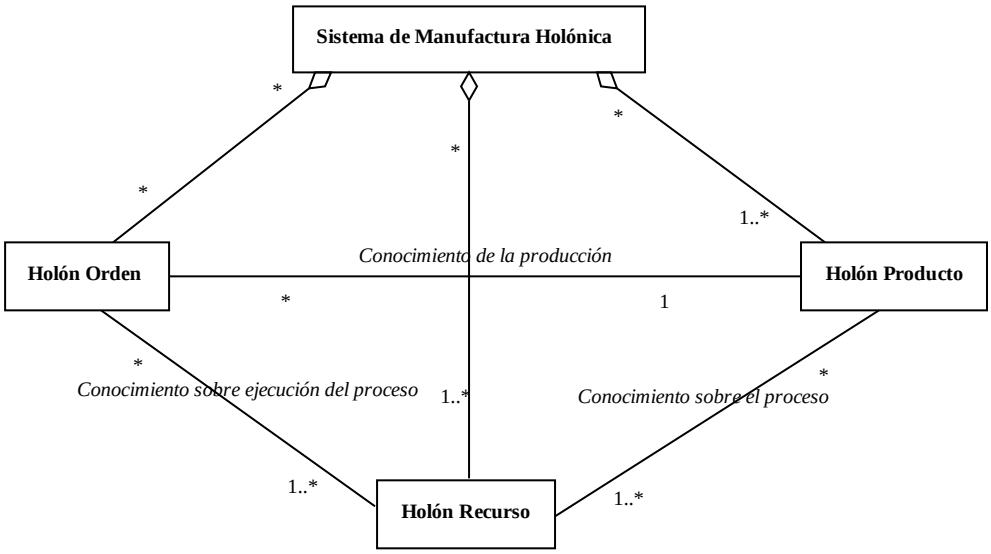


Figura 2.8: Diagrama UML de los componentes básicos de un HMS

los maneja el holón Producto, y los aspectos logísticos relacionados con el cliente los asume el holón Orden. Estos tres holones forman una holarquía, tal como se defininió en la sección anterior. El holón Staff proporciona conocimiento experto sobre algún otro aspecto, pero es opcional. Esta arquitectura de referencia se basa en el paradigma orientado a objetos para establecer la estructura y diseño de un HMS, y utiliza el Lenguaje Unificado de Modelado (UML de aquí en adelante) para representar la implementación computacional de los holones y las relaciones entre ellos. La figura 2.8 muestra el diagrama UML de un Sistema de Manufactura Holónico, con los tres holones básicos que lo conforman, su cardinalidad y la información que resulta de las asociaciones entre estos holones.

Los aportes de la arquitectura PROSA consisten en la independencia de la estructura del sistema con respecto a los algoritmos de control, al igual que la independencia de asuntos tecnológicos o logísticos, y la auto-similaridad en la arquitectura. Los aspectos más importantes de esta arquitectura de referencia se presentarán a continuación, como lo son su organización interna, las características que exhibe, así como la forma

en que se relaciona con nuestra investigación.

### 2.5.1. Estructura de la arquitectura de referencia PROSA

La estructura de la arquitectura PROSA está basada en tres holones básicos: Holón Recurso, Holón Producto y Holón Orden, como se había mencionado anteriormente. Estos holones se estructuran utilizando conceptos del diseño orientado a objetos, como lo son *agregación y especialización*.

- El *Holón Recurso* refleja el equipamiento de un sistema de manufactura, en una parte física y una parte de procesamiento de información. Ofrece capacidad de producción y funcionalidad a los otros holones.
- El *Holón Producto* contiene las especificaciones de un producto determinado en cuanto a procedimientos para asegurar su calidad, proceso para obtenerlo, requerimientos de insumos, entre otra información.
- El *Holón Orden* representa una tarea dentro del sistema de manufactura. Maneja información asociada al estado del producto que se está produciendo, logística de la ejecución del proceso, pedidos de nuevos productos, entre otros aspectos.

A partir de las relaciones entre estos holones básicos se puede obtener conocimiento sobre otros aspectos del sistema de manufactura. Asi, la relación entre los holones Recurso y Producto permite obtener conocimiento sobre el proceso de producción, por ejemplo cómo ejecutar cierto proceso sobre un recurso, capacidad de un recurso, parámetros del proceso, calidad del proceso, entre otros aspectos. Entre los holones Producto y Orden se tiene el conocimiento sobre la producción, el cual se representa como información y métodos para obtener un producto utilizando ciertos recursos. Entre los holones Orden y Recurso se puede obtener información sobre la ejecución de un proceso de producción, tal como la manera de iniciar un proceso, como reservar recursos, y monitorear el progreso de la ejecución, entre otras. En la figura 2.9

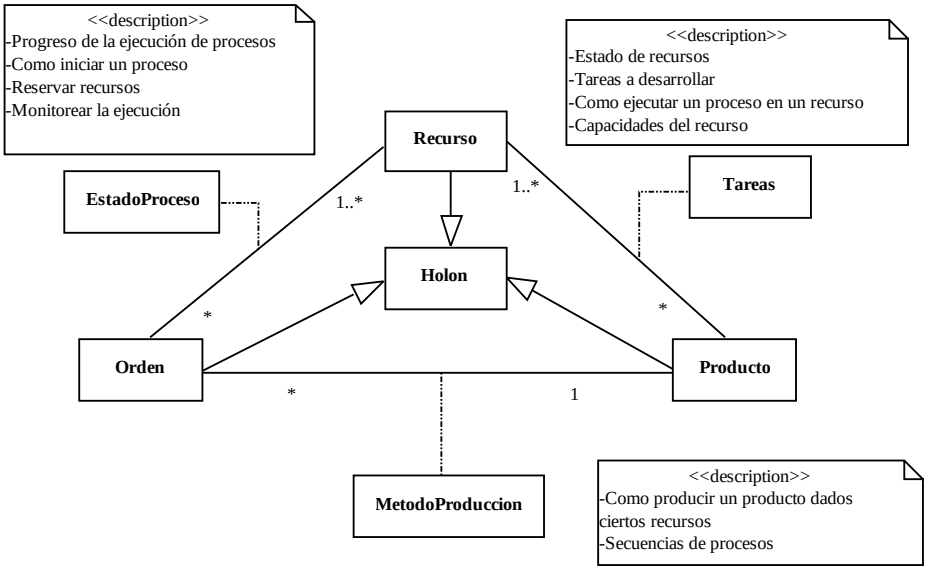


Figura 2.9: Holones básicos de la arquitectura PROSA

se pueden observar las relaciones de especialización de estos tres holones con la *clase Holón*, representados como un *Diagrama de Clases* utilizando la notación del Lenguaje Unificado de Modelado (UML). Además se pueden apreciar las clases que resultan de las relaciones entre estos holones básicos y la información que manejan, en comentarios.

### 2.5.2. Modelo detallado de los holones básicos

Además de su organización en holones básicos, la arquitectura PROSA para los Sistemas de Manufactura Holónica (HMS) bosqueja tres aspectos que detallan a los holones básicos. Estos son: la información que deben manejar, las funcionalidades que deben de tener y de qué manera van a interactuar con los demás holones [110], obteniéndose de estas interacciones el conocimiento del proceso, la producción y la ejecución del proceso. La figura 2.10 extiende a la figura 2.8, puesto que agrega a cada clase los atributos y métodos de cada holón básico, al igual que las asociaciones entre estos holones. En su tesis doctoral, Wyns establece que no es necesario ceñirse a esta

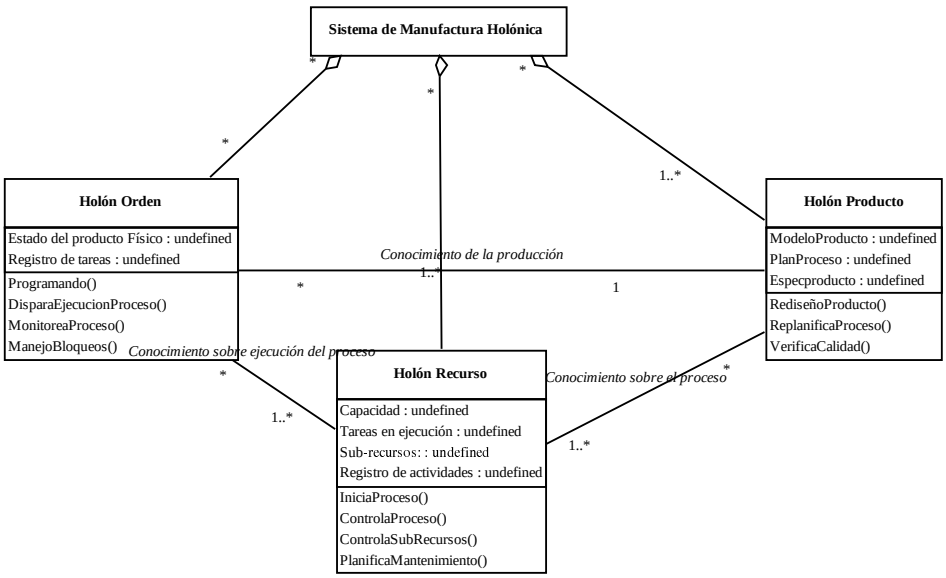


Figura 2.10: Proyección de datos y funciones a los holones básicos

información y funcionalidad que manejan los holones, sino que mas bien el modelo detallado es una descripción de lo que se puede esperar de un holón, con respecto a las especificaciones del sistema de manufactura. Como ejemplo de una implementación detallada de un HMS bajo la concepción PROSA se puede encontrar en el trabajo de Blanc [5] dirigida a un proceso de manufactura de vidrios blindados.

En la figura 2.10 se muestra la información que maneja cada holón, en forma de atributos que pertenecen a una clase UML. Estos atributos se encuentran en la sección central del rectángulo que describe a cada clase. Así, el holón Recurso maneja información sobre la capacidad del recurso, las tareas que está ejecutando, los subrecursos que tiene asociados, y un registro de actividades. El holón Producto maneja información sobre el plan del proceso, descripción del producto y requerimientos de calidad; mientras que el holón Orden maneja información sobre la traza del estado actual del producto físico, el progreso de la tarea, los datos históricos de las tareas realizadas. Es importante recalcar que estos atributos son descriptivos: pueden variar de acuerdo a las especificaciones del sistema de producción. Las funcionalidades que

tiene cada holón se muestran en este diagrama, en la parte inferior del rectángulo que describe a cada clase. Por ejemplo, para el holón Producto, las funcionalidades son: Rediseño del producto, replanificación del proceso, y verificar la calidad del producto. El holón Recurso tiene las funcionalidades de iniciar el proceso, controlar el proceso, controlar subrecursos y planificación del mantenimiento. El holón Orden tiene entre sus funcionalidades la de programar el proceso, disparar la ejecución de un proceso, monitorear el proceso y manejar bloqueos de un proceso determinado.

Para una mejor profundización en cuanto al comportamiento e interacción de los holones básicos se recomienda revisar la tesis doctoral de Jo Wyns [110], donde se describe la dinámica de las siguientes interacciones: Creación de una Orden, Manejo de la producción por una Orden, y agregar un nuevo Recurso. Esta interacción se describe utilizando diagramas de colaboración UML.

### 2.5.3. Autosimilaridad

Según Mandelbrot [71], un objeto es autosimilar o autosemejante si sus partes tienen la misma forma o estructura que el todo, aunque puedan presentarse a diferente escala y su forma varíe levemente. Este concepto lo utiliza la arquitectura PROSA, permitiendo que los holones básicos que conforman una Unidad de Producción se puedan agrupar en un *holón agregado*, manteniendo su misma estructura y mismos tipos de holones (Producto, Recurso, Orden y Staff). Esto se denomina *principio de autosimilaridad* de los sistemas holónicos [10, 110]. Para implementarlo, se adoptaron los mecanismos de estructuración orientados a objetos, como lo son la *agregación* y la *especialización*, los cuales permiten el diseño por capas de los holones, donde las mas altas son compartidas (abstractas) y pueden interactuar con otros subsistemas, de esta manera, holones del mismo tipo tienen interfaces y conducta similar. Esta característica determina en parte la re-configurabilidad del sistema de control.

La arquitectura PROSA muestra autosimilaridad "horizontal" y "vertical". La au-

tosimilaridad horizontal se refiere a la similaridad entre diversas especializaciones de holón dentro de un mismo nivel. Como ejemplo se tienen los holones que implementan los insumos, equipamiento y recurso humano dentro de una unidad de producción. Cada uno de estos holones son especializaciones del holón Recurso, y presentan la misma interfaz y funcionalidad. La autosimilaridad vertical se relaciona la semejanza de la estructura holónica en cada nivel de agregación. Un holón Recurso asociado a un dispositivo presenta una funcionalidad similar al holón Recurso de una Unidad de Producción, y al holón Recurso de una Planta Industrial compleja.

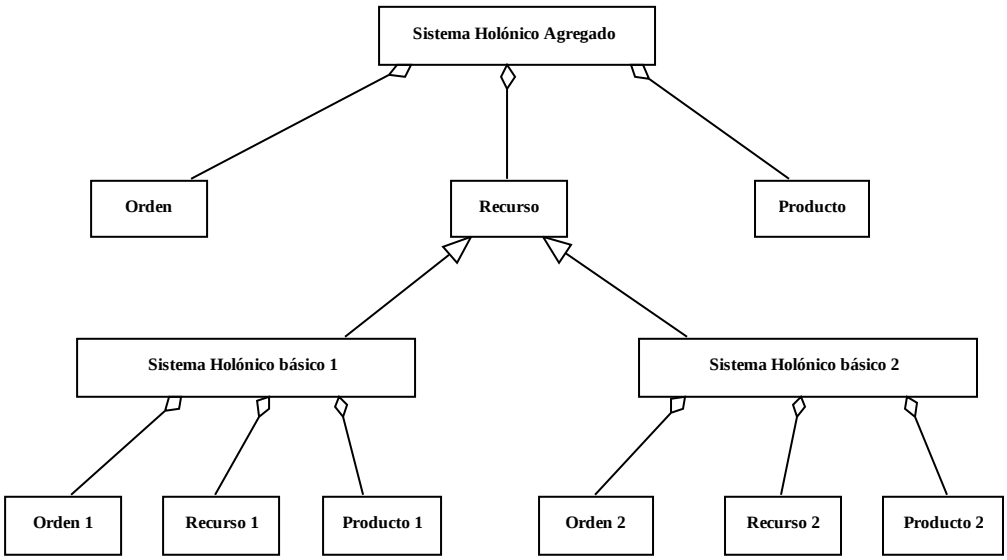


Figura 2.11: Una Holarquía de producción

En la figura 2.11 se puede observar que varios holones que controlan un proceso determinado forman parte de un holón de mayor complejidad, que mantiene la misma estructura. El holón global recibe unas metas de producción, las cuales distribuye como sub-metas para los holones que lo conforman. Cada uno de estos holones a su vez ejecutan su sub-meta, y pueden coordinarse o bien por el holón que los contiene, o negocian los objetivos de producción entre ellos.

## 2.6. Comparación de PROSA con otras arquitecturas holónicas

A pesar de que PROSA es la implementación holónica más exitosa, existen otras propuestas de arquitecturas basadas en holones, y sobre esta área del conocimiento todavía se está investigando. Un resumen de estas arquitecturas se encuentra en el trabajo sobre Sistemas Multiagente de Manufactura, documentado por Botti y Giret [44]. Los siguientes elementos indican brevemente en que consiste cada arquitectura, sus respectivos autores, y la referencia original de las publicaciones que las definen. Estas arquitecturas son las siguientes:

- La arquitectura basada en Agentes y Bloques funcionales, propuesta en la Universidad de Keele por Deen y Fletcher [41].
- Metamorph II, propuesto por Maturana y Norrie, de la Universidad de Calgary, basado en una propuesta de Maturana *et al* basada en agentes inteligentes [72].
- INTERRAP. Propuesto por el Centro Germano de Investigación para la Inteligencia Artificial [75]. Esta arquitectura presenta tres niveles concurrentes, y no solo se ha utilizado en HMS, sino en otras aplicaciones tales como el control de flujo de trabajos, logística en empresas virtuales y fuentes de información basada en agentes, entre otras.

Estas arquitecturas de referencia basadas en holones mas la arquitectura PROSA, se pueden comparar bajo los siguientes criterios:

1. Ámbito de aplicación. Hacia que tipos de proceso de fabricación o manufactura está orientada esta arquitectura. Incluso, si existe alguna arquitectura de propósito general.

2. Estructura del holón. Forma en que conciben y definen al Holón fundamental. Tipos de holones fundamentales.

3. Estructura con varios holones. Se refiere a la forma en que se organizan estos holones: una holarquía, heterarquía, mixta, variable o alguna otra forma de organización.

4. Relación con los agentes. Este criterio de comparación consiste en determinar si la arquitectura considera explícitamente al agente como parte o todo el holón.

5. Controlabilidad. Se basa en determinar si la arquitectura de referencia considera explícitamente algún mecanismo de control de su comportamiento, y como representa su conducta.

Una mención especial merece el aspecto de controlabilidad, como aspecto para comparar varias arquitecturas de sistemas holónicos. La controlabilidad se asocia a la posibilidad de que las variables de estado tomen cualquier valor deseado, en un tiempo finito, sin importar las condiciones iniciales. Esto se interpreta como la capacidad de que un sistema pueda adaptar sus diferentes configuraciones para cumplir con una meta de producción, dada cierta planificación. La controlabilidad permite que el sistema exhiba las capacidades de *manufactura ágil*, para cumplir con rápidos cambios en el proceso de manufactura o en el estado de los recursos. Gouyón [48] establece dos tipos de flexibilidad: la de las máquinas (los recursos) y la flexibilidad de las trayectorias (o trazas de las variables asociadas al proceso). La controlabilidad permite ejercer control, valga la redundancia, sobre el estado de los recursos e incidir en el proceso, por tal motivo, es un aspecto importante en la comparación entre arquitecturas de sistemas holónicos.

La tabla 2.2 muestra la manera en que se establece la comparación entre estas arquitecturas:

Tabla 2.2: Comparación de arquitecturas de referencia basadas en Holones

| Arquitectura / Criterio | PROSA | Agentes y bloques funcionales | Metamorph II | Interrap |
|---|---|---|---|---|
| Ámbito de aplicación | Empresas de Manufactura | Entorno de manufactura | Sistemas inteligentes de manufactura | De Propósito general, utilizada en aplicaciones web, aunque se puede llevar a un entorno industrial |
| Estructura del Holón | Definida en UML. Tiene componente físico y componente de información | Un holón se conforma por un agente, que implementa la planificación de alto nivel, y un bloque funcional que proporciona el control de bajo nivel en tiempo real | Solo el holón producto contiene componente físico. Los demás holones manejan información | Similar a PROSA. |
| Estructura social | Tres holones básicos, un holón Staff. Interrelaciones entre ellos | Se organizan los holones en un dominio de cooperación, donde los holones se comunican y operan, creado dinámicamente | Holones de producto, de modelo de producto y de recurso. Se organizan en estructuras cambiantes denominadas cluster virtuales dinámicos | En el nivel de planificación cooperativa se organizan las holarquías |
| Relación con agentes | No es explícita, aunque varios autores han implementado algunos holones utilizando agentes | Basada en agentes que interactúan con holones | Basado en agentes, los cuales administran dispositivos físicos y otros agentes realizan labores de mediáción | Agentes organizados en varias capas |
| Controlabilidad | Separa algoritmos de control de la estructura del holón | Reside un control proceso/máquina en cada bloque funcional, a bajo nivel | No la trata explícitamente | No la trata explícitamente |

## 2.7. Descripción de un Sistema de Producción Continua bajo la arquitectura PROSA

Para desarrollar este trabajo doctoral es necesario enmarcar las características generales de un SPC dentro de una arquitectura de automatización holónica, de manera que luego se pueda diseñar un mecanismo de supervisión adecuado y validarlo. Para ello se selecciona a PROSA como la arquitectura de referencia para describir nuestro SPC. Los aspectos que motivan esta decisión son los siguientes:

- La descripción de sus holones se hace mediante el lenguaje de modelado UML, lo que facilita su diseño e implementación dentro de una arquitectura computacional.

- La característica de autosimilaridad, con lo que se puede reproducir la estructura de holones básicos en el piso de planta, para una unidad de producción, una planta industrial, y así sucesivamente. De esta manera se obtiene la capacidad de reutilización y herencia, para los sistemas de información que utilizan el estándar PROSA.

- Por medio de la adopción de la arquitectura PROSA se facilita que un SPC obtenga la capacidad de reconfiguración, ya que establece explícitamente las relaciones entre los recursos, producto, método de producción, orden y configuración para ejecutar una misión de producción. Sobre la reconfiguración estarán dirigidos los esfuerzos de la supervisión y el control.

- Por último, y no menos importante, es que PROSA separa los algoritmos de control de sus holones, lo que facilita para implementar el mecanismo de supervisión y control sobre una unidad de producción holónica y luego llevarlo hacia holones de mayor complejidad sin cambios en su estructura general, razón por

la cual la supervisión y control del holón también presentan la característica de autosimilaridad.

Para aplicar esta arquitectura de referencia a un SPC, que cumpla con los necesidades de integración y descentralización, es necesario considerar que el SPC está conformado por varias Unidades de Producción Autónomas (UPA, de aquí en adelante) asociadas a un proceso físico de transformación, el cual puede subdividirse en etapas atómicas que ejecutaría cada de estas unidades. La **autonomía** de la UPA se basa en la capacidad que tendría cada UPA para **supervisar su propia conducta**, en función de sus metas y en la observación de su entorno. Los siguientes capítulos establecen como modelar la conducta de la UPA, cómo tiene lugar la supervisión, y como gestionar la UPA en función de metas globales del SPC y en las relaciones con otras UPA. En cuanto al proceso físico, se asume que la UPA dispone de mecanismos de medición de las variables más representativas, donde el *sensor* es un ejemplo de estos mecanismos, y un subsistema de control físico de procesos, implementado por diversos controladores y accionamientos electrónicos. Todos estos dispositivos por definición son *recursos* de la unidad de producción, con los que interactúa directamente con el proceso industrial a desarrollar.

Una vez establecido que la UPA es un conjunto de holones basados en la arquitectura de referencia PROSA, es necesario establecer un punto inicial, que para nuestro caso sería una Unidad de Producción a cargo de un proceso atómico, es decir, un proceso que no pueda descomponerse en sub-procesos. La figura 2.12 muestra una UPA básica, donde se puede apreciar que en la lógica de su funcionamiento están presentes los tres holones fundamentales de la arquitectura PROSA, interactuando con un mecanismo de Supervisión y Control, compuesto por un supervisor, un conjunto de detectores de eventos, los cuales van a estar monitoreando el proceso físico por medio de sensores, y por un conjunto de actuadores, que van a afectar el proceso físico por medio de accionamientos electrónicos implementados en controladores o en el equipamiento

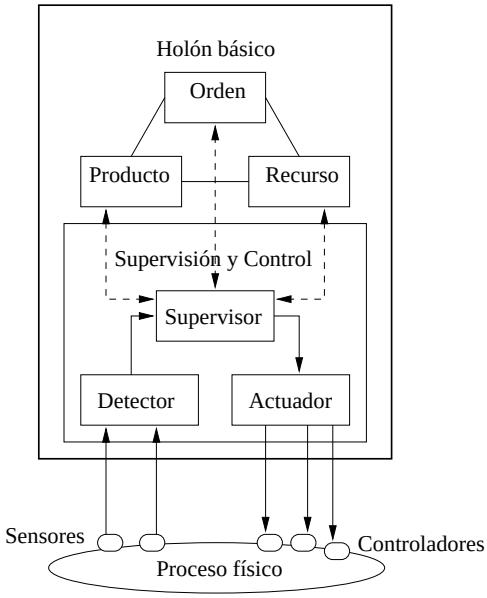


Figura 2.12: Unidad de Producción Autónoma Básica

físico de la UPA. La misión del supervisor es asegurarse que la conducta de esta unidad de producción se encuentre dentro de una trayectoria deseada, para lo cual en ciertas ocasiones genera comandos de control para cambiar la configuración de la UPA y así inducir cambios en el comportamiento del proceso fisico. Esta UPA se considera por sí sola como un holón, de manera que se pueda denominar *Holón Básico*, pero a su vez es un *Recurso* de una planta que contenga varias unidades de producción. Varios Holones Básicos conforman un *Holón Agregado*, el cual a su vez puede conformar holones de mayor complejidad.

La figura 2.13 muestra como se organizan los holones dentro de una *Planta de producción*, la cual dispone de varias unidades de producción continua. Cada proceso físico es monitoreado por un conjunto de dispositivos, cuyas mediciones son procesadas por un *detector de eventos*, el cual notifica a un *supervisor del holón* sobre la ocurrencia de tales eventos, con el fin de ejecutar alguna decisión por medio de un dispositivo *actuador*. El proceso que se está monitoreando corresponde al avance de una orden de producción, generada por un holón Orden, y aplica un método de producción que pertenece a un producto determinado, que está asociado a la instancia de un holón

Producto. La generación de las órdenes de producción se presenta en un nivel de agregación superior al de cada Unidad de Producción, en este caso denominado *Nivel de planta*, donde cada Unidad de Producción básico es un *recurso* de este holón agregado. De acuerdo a los eventos detectados en este holón del siguiente nivel, establece unas acciones de control que consisten en disparar otras órdenes de producción para los holones del nivel inferior, y el proceso se reinicia. Otra manera en que pueden coordinarse los holones a un mismo nivel es que varios de ellos reciban una meta global de producción, y luego negocien entre sí de acuerdo a las especificaciones de lo que se va a producir, sus capacidades de producción, insumos que reciben y producto intermedio que pueden entregar. Como resultado de esta negociación, cada uno de estos holones generan sus propias órdenes de producción, iniciando sus correspondientes procesos productivos.

Organizando en el tiempo las acciones e intercambio de mensajes entre los diversos actores que intervienen durante el monitoreo de un proceso industrial se obtiene una secuencia de eventos que se puede expresar por medio de un *diagrama de secuencia* UML. La secuencia de eventos es similar tanto para el holón básico como para un holón agregado, la diferencia está en que el holón básico interactúa directamente con los dispositivos de la UPA y demás equipamiento, mientras que el holón agregado interactúa por medio del intercambio de mensajes con los demás holones que contiene (cada unidad de producción es un holón recurso para una planta, por ejemplo). La figura 2.14 muestra un diagrama de secuencia en UML que explica como sería el proceso de detección de eventos, cuando se está monitoreando un recurso. La secuencia principal de eventos es la siguiente: periódicamente un controlador o sensor está midiendo valores directamente del proceso, y los está reportando a un componente software que hace la función de detector de eventos. El detector de eventos conoce el estado actual del proceso porque el supervisor así se lo indica. Si el detector de eventos determina un cambio en el estado del proceso, indica al supervisor la ocurrencia

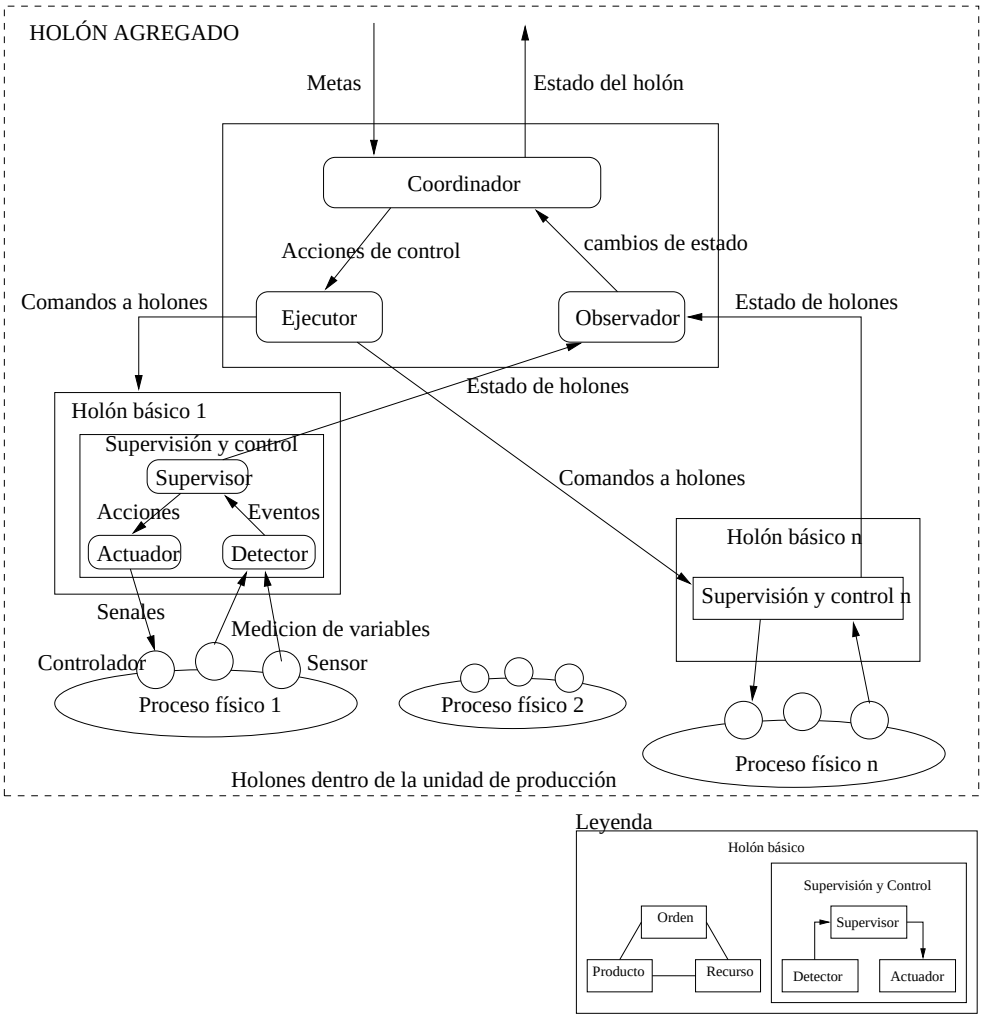


Figura 2.13: Sistema de Producción Continua bajo un enfoque Holónico

de un evento. El supervisor actualiza el estado del proceso, y de acuerdo a la meta de producción, decide si se debe ejecutar alguna acción sobre el proceso industrial, la cual ejecuta por medio de un actuador, que es un recurso de la UPA al igual que los sensores, detectores y supervisores. Una vez se ejecuta la acción por medio de un cambio a la configuración de la planta, sigue el proceso hasta que se detecte nuevamente un evento. Si se desea profundizar mas en estos detalles conceptuales, el capítulo seis de este documento está dedicado al modelado conceptual de la supervisión y control de esta UPA holónica.

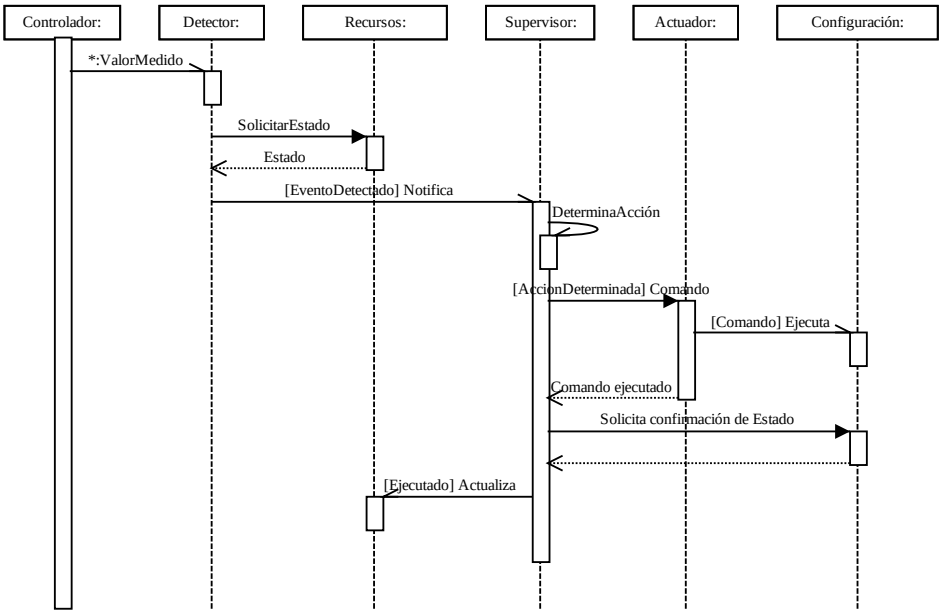


Figura 2.14: Diagrama de secuencia para detección de eventos en un recurso

## 2.8. Conclusión

Se ha seleccionado la arquitectura holónica de referencia PROSA, para describir unidades de producción autónomas, debido principalmente a sus cualidades de auto-similaridad, lo que permite reproducir la organización de los holones en diferentes niveles de agregación. Algunas unidades de producción tienen a su cargo procesos continuos, por tal motivo se ha establecido un esquema que permite describir un proceso continuo dentro de una unidad de producción holónica mediante la interacción de varios holones PROSA, cuyo comportamiento es supervisado para proporcionarle mayor autonomía. La agregación de estos holones permiten obtener una unidad de producción de mayor complejidad, cuya conducta a su vez está coordinada por un mecanismo de supervisión análogo al que tienen las unidades de producción más sencillas.

# Capítulo 3

# Descripción del comportamiento dinámico de un HMS basado en la arquitectura PROSA

En un Sistema de Producción Continua la misión principal de una Unidad de Producción Autónoma (UPA, de aquí en adelante) consiste en mantener constante un flujo de producto, bien sea para distribuir producto terminado a un cliente, o como un paso intermedio dentro de un proceso productivo de mayor complejidad. Se asume que esta UPA se concibe bajo la arquitectura de referencia PROSA, tal como se describió en la sección anterior, y trabaja en conjunción con otras UPAs que conforman un sistema de producción de mayor complejidad que debe coordinarse por medio de algún mecanismo para cumplir con la misión global de producción, sin que se presenten situaciones no deseadas, y se utilicen de manera óptima los recursos e insumos. Por lo tanto, se requiere de un mecanismo de *Coordinación y Control* que permita mantener el comportamiento global del sistema dentro de un plan de producción, con lo cual es necesario obtener y hacer seguimiento a los valores de las variables del proceso a controlar (de aquí en adelante se denominará como la "planta" al sistema

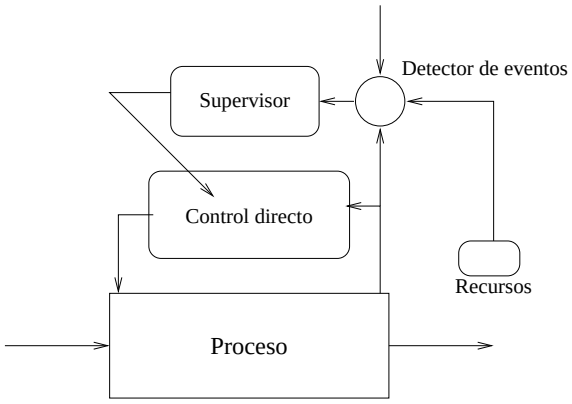


Figura 3.1: Sistema de Control Supervisorio estándar

de producción que hay que controlar dentro de una unidad de producción), y deducir el estado de ese proceso, que de aquí en adelante se llamará la "imagen" del proceso. Conociendo el estado en que se encuentra el proceso en un momento determinado, se puede describir la dinámica del cambio de estados por medio de un Sistema de Eventos Discretos, y así poder aplicar la *teoría de Control Supervisorio* para resolver el problema de como controlar un sistema de producción continua concebido bajo el paradigma holónico.

El Control Supervisorio tiene una estructura general, que se puede apreciar en la figura 3.1. Generalmente existe un proceso continuo, modelado como un sistemas de ecuaciones diferenciales o algebráicas en variable continua, un control regulatorio modelado como sistema de eventos discretos y en variable continua, y un control supervisorio cuya naturaleza es discreta. Los componentes de control regulatorio y el proceso pueden agruparse en una unidad autónoma, y un proceso industrial complejo se puede representar como la *composición* de varios de estas unidades. El Control Supervisorio actuaría como un *control de alto nivel*, influyendo sobre una familia de controladores que están regulando un proceso físico. Para SPC este control significa mantener un proceso dentro de un estado deseado, el cual está cambiando de acuerdo a las leyes físicas que gobiernan el proceso industrial (en otras palabras, la física de la planta).

Para controlar una planta donde ocurre un proceso industrial hay que llevar a cabo dos pasos: 1. Representar la evolución de las variables de ese proceso, y la síntesis de un supervisor para procurar que las variables del proceso a controlar exhiban una conducta deseada, o al menos prevenir que ocurran situaciones no deseadas. La dinámica del proceso industrial se puede representar por medio de un Sistema a Eventos Discretos (DES, de aquí en adelante), donde cada estado corresponde a un modo de operación, mientras que los eventos se consideran como el cambio instantáneo de un estado a otro. Esta dinámica está muy relacionada con la evolución continua de las variables que rigen al proceso continuo, por lo que es necesario acoplar de alguna manera el DES con el comportamiento de estas variables continuas, y a su vez proyectar los valores de estas variables hacia un conjunto de estados discretos. En todo proceso industrial las etapas del proceso y las condiciones de operación se pueden expresar cualitativamente como **estados**, y el cambio de un estado a otro se denomina una **transición**. Si se etiqueta cada transición con un símbolo, el comportamiento de un proceso industrial en el se representa por medio de una secuencia de símbolos, los cuales representan las transiciones que hayan tenido lugar. En la figura 3.2 se aprecia un esquema de proyección de una variable continua hacia un conjunto de estados discretos. Los cambios en el comportamiento de la variable continua se puede expresar como la ocurrencia de eventos, con el consiguiente cambio de estado. De acuerdo al dibujo, cuando la variable continua se comporta según la región de operación A, el autómata se encuentra en el estado $A$. Si posteriormente la variable se comporta de acuerdo a la región de operación B, implica que ha ocurrido el evento $\beta$, pasando al estado B. Si la variable empieza a comportarse como la trayectoria de la región G, entonces ha ocurrido el evento $\epsilon$. Al final de esta sección se describirán algunos aspectos relacionados con la detección de eventos.

Las siguientes definiciones establecen los conceptos de estado del sistema y eventos, necesarias para caracterizar una unidad de producción.

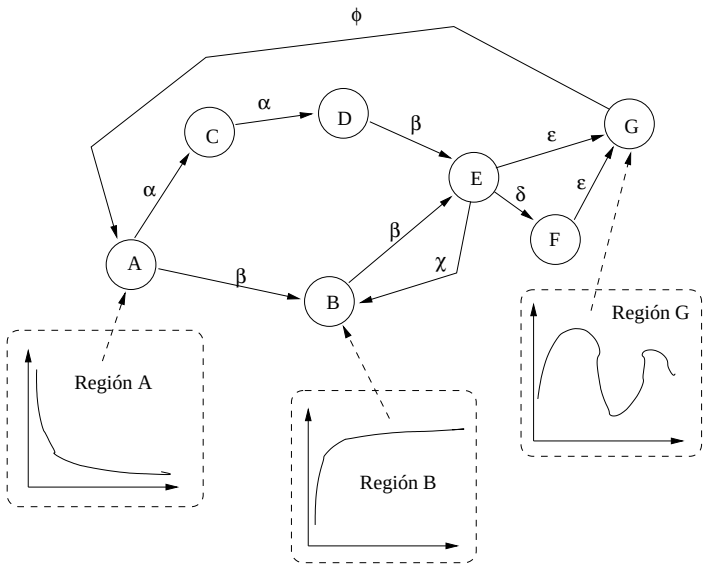


Figura 3.2: Proyección de variables continuas a estados discretos

**Definición 3.1. Estado de un sistema.** Dependiendo del *espacio de estados* existen dos tipos de Estados para un sistema dado. Cuando el espacio de estados es continuo, el estado toma valores de un conjunto de vectores n-dimensionales representado por los números reales (o a veces números complejos). Por ejemplo, $\Re^n$ para todo $n \geq 1$. Este espacio continuo se denota como $X \in \Re^n$. El estado $X$ puede ser dimensionalmente finito o a veces puede ser dimensionalmente infinito. Ejemplo de espacios de estado continuo se tiene cuando se mide la altura de un nivel en un líquido, presión, temperatura, voltaje, entre otros. Cuando el espacio de estados es discreto, los estados se toman de valores de un conjunto discreto finito o contable $\{q_1, q_2, \ldots\}$. Este estado discreto se denota por $q$. Como ejemplo de espacios de estado discreto se tienen las diversas configuraciones de una unidad de producción, las condiciones de operación en que se encuentran las variables, las etapas en que se encuentra cierto proceso están asociados a un estado discreto del sistema.

**Definición 3.2. Evento.** Por evento se define a un suceso instantáneo que puede originar una transición entre estados.

Dentro de esta sección se explicará la manera como se concibe un proceso industrial, y como se describe su comportamiento, para luego presentar un panorama de como se describiría la dinámica de un Sistema Holónico. El proceso de síntesis del supervisor y la gestión del Holón se mostrarán en las secciones subsiguientes.

## 3.1. El Proceso de Producción como un Sistema Dinámico

La Dinámica de un Proceso Industrial que se quiere controlar difícilmente se puede regular por medio de la teoría de control continuo o discreto de forma separada, ya que involucra variables continuas y discretas al mismo tiempo. Su regulación se puede lograr por medio de la integración de estas variables por medio de funciones de proyección, y la coordinación se puede llevar a cabo por medio de técnicas de control supervisorio, que actúan sobre un sistema de eventos discreto, para luego proyectar sus acciones al sistema continuo. Normalmente, cualquier proceso industrial tiene tanto variables continuas como discretas, ya que dispone de magnitudes físicas y estados lógicos tanto del proceso como de los diversos dispositivos electro-mecánicos con los que interactúa la UPA. Como estas variables constantemente están cambiando en el tiempo, se puede considerar el Proceso Industrial como un Sistema Dinámico (SD, de aquí en adelante). De manera formal, un SD se describe como una estructura $(T, U, Y, Q, \Omega, \delta, \lambda)$, definida dentro de la teoría de sistemas [40]:

Donde:

$T$ es el conjunto de tiempo

$U$ es el conjunto de entradas, que contiene los valores de entrada posibles al sistema

$Y$ es el conjunto de salidas

$Q$ es el conjunto de estados

$\Omega$ es el conjunto de entradas *admisibles*, contiene un conjunto de funciones de entrada para usar durante la operación del sistema. Este conjunto está contenido en todas las entradas posibles $U$

$\delta$ es la función de transición. $\delta : Q \times T \times T \times \Omega \rightarrow Q$

$\lambda$ es la función de salida

$\lambda : Q \rightarrow Y$

Llevando los elementos de la estructura matemática descrita anteriormente al dominio de un Proceso Industrial, se tiene que el conjunto de tiempo $T$ serían los números reales positivos, las entradas $U$ corresponderían a señales que envían los dispositivos, que afectan las magnitudes y leyes físicas de este Proceso. El conjunto $Y$ de salidas consistiría en aquellas variables que se pueden medir, mientras que el conjunto de estados $Q$ correspondería con aquellas variables del proceso industrial que se está estudiando, tanto de naturaleza continua como discreta.

## 3.2. Sistemas Dinámicos Híbridos

Un sistema que presenta mezcla de variables continuas con dinámica de eventos discretos se conoce como un Sistema Dinámico Híbrido, (o SDH, de aquí en adelante). Por lo tanto, a la hora de analizar y de modelar los SDH hay que tener en cuenta que tanto la dinámica continua como la dinámica discreta están interactuando permanentemente. Actualmente existen varios enfoques para estudiar los SDH. Un sistema controlado por un computador es un ejemplo de un sistema híbrido, ya que las variables del proceso físico son medidas por sensores y sus valores representados digitalmente, como magnitudes discretas. Un esquema de un sistema híbrido básico se observa en la figura 3.3. Un componente esencial de estos sistemas híbridos lo son los

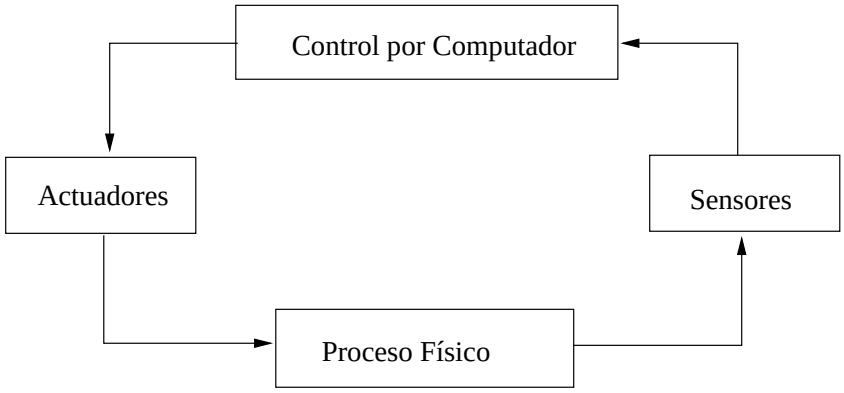


Figura 3.3: Sistema híbrido general

*sensores y actuadores*, los cuales cumplen con la misión de ser interfaz entre variables continuas con variables discretas.

Uno de los primeros trabajos fue el de Sreenivas y Krogh, que postulan que un sistema híbrido resulta de la interacción de dos sistemas: uno de ellos es continuo, y el otro es discreto. El sistema continuo evoluciona de acuerdo a una secuencia de funciones a trozos, las cuales se seleccionan a partir de reglas prefijadas, que dependen del estado de los procesos continuos [104]. Nerode y Kohn definen a un sistema híbrido como una interacción entre procesos continuos y discretos, estos últimos concebidos como un autómata secuencial [82]. Bemporad y Morari presentan un enfoque basado en control predictivo y lógica para el manejo de procesos [77], al igual que una estrategia de verificación por medio de programación matemática. Lygeros [69] establece una formulación matemática destinada a controlar procesos en gran escala, la cual es la base del trabajo de Chacón *et al* con los sistemas dinámicos acoplados [20].

Uno de los trabajos sobre SDH para tener como referencia en modelado es el de Chacón *et al*, derivado del de Lygeros [69, 21] donde establecen que un SDH contiene tanto variables discretas como continuas, y está definido sobre un conjunto temporal $T$, donde $T$ es el conjunto de valores que puede tomar el tiempo. $T = [t_i, t_f] \subset \Re$, bien de forma continua, o continua en saltos instantáneos. El conjunto $T$ evolucionando se

puede representar como:

$$T = ([t'_0, t_1][t'_1, t_2] \ldots [t'_{n-1}, t_n]) \tag{3.1}$$

Con $t_i \in T$, para todo $i$. $t_0 = t_i$ y $t_n = t_f$. Debe cumplirse que: $t_i = t'_i \leq t_{i+1}$, para $i = 0, 1, 2, ..., n-1$. Donde $t_i$ es el tiempo en que ocurre una entrada o un cambio de estado. La evolución de los estados continuos es una función de $t \in T$, mientras que los estados discretos varían como una función constante. Una vez establecido el conjunto temporal $T$ es necesario definir formalmente a un SDH, para llevar ese concepto hacia el control de procesos industriales.

**Definición 3.3. Sistema Dinámico Híbrido** Un SDH puede describirse formalmente como una 7-tupla de la siguiente manera:

SDH $= (X, U, Y, I, E, f, h)$, donde:

$X = X_d \cup X_c$ es el conjunto de variables de estado, discretas y continuas.

$U = U_d \cup U_c$ es el conjunto de entradas, tanto discretas como continuas.

$Y = Y_d \cup Y_c$ es el conjunto de salidas, discretas y continuas.

$I$ es el conjunto de condiciones iniciales.

Los elementos de estos conjuntos X,U,Y se denotan como $(q, x)$, $u$ y $y$ respectivamente. $I \subset X$ es el conjunto de valores iniciales del sistema, donde $(q(t'_0), x(t'_0)) \in I$. Además $f : X \times U \longmapsto X_c$ es la dinámica continua del sistema, q(t) es constante para el intervalo $(t_{i-i}, t_i)$, $f$ es una función vectorial, y tanto la razón de cambio de $x$ como la función de salida $h : X \times U$ se definen para $t \in [t_i, t_f]$ como:

$$\dot{x} = f(x(t), q(t), u(t)) \texttt{ para todo } t \in [t_{i-1}, t_i] \tag{3.2}$$

$$y(t) = h(q(t), x(t), u(t)) \tag{3.3}$$

Donde $E$ es la evolución discreta de los estados del sistema, $E \subset X \times U \times X$. El estado puede cambiar en el tiempo $t_i$ desde $(q(t_i), x(t_i))$ a $(q(t_i'), x(t_i'))$ siempre y cuando esos elementos pertenezcan a $E$.

Llevando los conceptos abstractos del SDH al proceso industrial que se desea controlar, se encuentran las siguientes correspondencias:

- Los valores físicos medidos directamente del proceso industrial por medio de los sensores corresponden a las variables continuas del sistema, a $X_c$. Como ejemplo están los niveles de un depósito, el voltaje en un momento dado, el pH de una mezcla, la concentración de una sustancia en un líquido, entre otras magnitudes que se puedan medir.

- Las condiciones de operación del proceso corresponden a las variables de estado discretas, a $X_d$. Como ejemplo está el arranque de la planta, operación plena, estado de falla, etc.

- Las entradas continuas al sistema $U_c$ son aquellas que varían los parámetros del controlador del proceso, permitiendo una evolución diferente del estado. Ejemplo: el grado de abertura de una válvula, caudal de un fluido, etc.

- Las entradas discretas $U_d$ son eventos puntuales que modificarán el estado posterior del proceso. Una abertura o cerradura de válvula, interrupción del paso de corriente son algunos de estos ejemplos.

- Las salidas continuas $Y_c$ son los valores que reportan los sensores sobre el proceso en el tiempo, las salidas discretas $Y_d$ son las condiciones de operación que se pueden observar externamente.

- Las condiciones iniciales $I$ del proceso industrial son los valores continuos iniciales del proceso y la condición inicial en que ese proceso se encuentra.

- La evolución del sistema $E$ fue descrita anteriormente, indica el cambio que ocurre en los estados discretos del sistema cuando ocurre un evento. La ocurrencia de un evento implica un paso de una condición de operación a otra.

- $f$ es la evolución continua del sistema. Viene dada por la naturaleza física y química del proceso que se quiere controlar.

- $h$ es la función de salida, que tiene en cuenta los valores de las magnitudes asociadas al proceso físico, y sus condiciones de operación para luego permitir la visualización de los mismos y su registro.

Lygeros [69] establece que un sistema de esta naturaleza evoluciona tanto discreta como continuamente por medio de unas funciones aplicadas a un conjunto $(t, q, x, y, u)$. Una extensión a esta definición de sistema híbrido se propone agregando una función de proyección de una región continua definida en $\Re$ hacia un elemento de un conjunto discreto, y estableciendo una función discreta $f_d$ que permite ir de un estado discreto a otro, cada vez que ocurre un evento, tal como lo define Chacón, Szigeti y Camacho [21]. Es necesario tener en cuenta que esta función $\psi$ no tiene inversa, ya que es una función que proyecta desde una región continua hacia una variable discreta. La función se define como $f_d : E \subset X \times U \times X \longmapsto X_d$ y describe la dinámica discreta del sistema, y la función de proyección $\psi$ se define como:

$$\psi(x(t_{i-1}), q(t_{i-1})) = q(t_i) \tag{3.4}$$

Si se considera a un SPC como un conjunto de unidades de producción intercambiando materia e información, el estado global del proceso se puede determinar como la suma de los estados de cada unidad de producción. Una explicación de como describir la dinámica de un sistema dinámico acoplado se encuentra más adelante, en esta sección.

## 3.3. Sistemas a eventos discretos

En la sección anterior se plantea que un proceso continuo puede proyectarse a una imagen discreta, que refleje como estados la región de operación en que se encuentra el proceso. En una planta industrial, por la parte discreta los eventos pueden ser generados por cambios en las regiones de operación o por paradas y/o arrancadas de la planta o por cambio en la estructura dinámica del sistema continuo, y puede ser modelado por una *Máquina de Estados Finitos o una Red de Petri*; y por la parte continua el sistema puede generar señales continuas a partir de fenómenos físicos como un cambio de temperatura y/o presión medida en la planta, por ejemplo, y puede describirse por medio de ecuaciones algebraicas o diferenciales. Un esquema de un SDH se puede ver en la figura 3.4. Los valores del sistema continuo que está evolucionando en el tiempo son los que determinan que eventos han ocurrido, los cuales cuando se detectan actualizan el sistema de eventos discretos, el cual a su vez determina las acciones de los actuadores, dependiendo del estado en que se encuentre. Un problema potencial puede surgir si no se detecta un evento, y el sistema discreto envía los parámetros correspondientes a un estado determinado, mientras que realmente se encuentra en otro estado, lo que llevaría a una conducta indeseada en el sistema en el mejor de los casos. Por este motivo es necesaria una detección precisa y oportuna de los eventos, y una supervisión de la planta.

Existen dos grandes enfoques matemáticos para representar la conducta de los DES: Máquinas de Estado Finito y Redes de Petri. Los trabajos de Cassandras y Lafortune [16]; Ramadge y Wonham [93], basados en teoría de lenguajes formales y autómatas han impulsado los estudios de control utilizando Máquinas de Estado Finito (MEF), mientras que Murata [80], Zhou and DiCesare [115] y otros investigadores han estudiado mecanismos para representar procesos industriales utilizando las redes de Petri. Las definiciones subsiguientes están basadas en los anteriores autores.

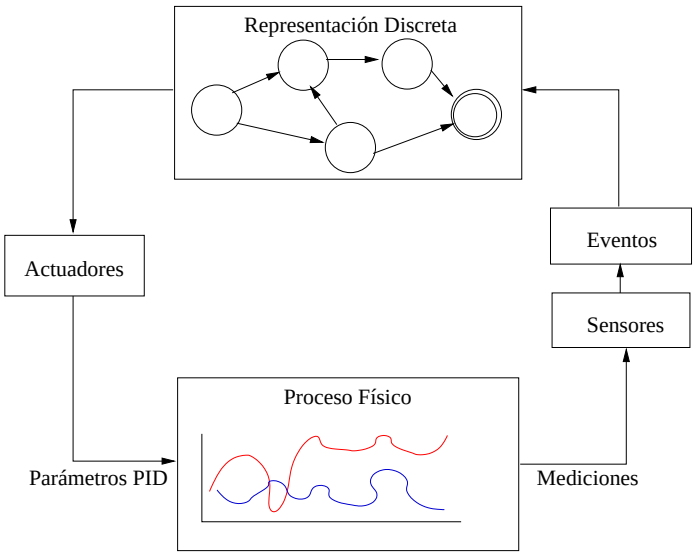


Figura 3.4: Sistema Dinámico Híbrido y elementos relacionados

### 3.3.1. Enfoque de Máquinas de Estado Finito

Supóngase que la planta que se quiere controlar es modelada por un autómata $G$, el cual tiene un lenguaje asociado $L(G)$ finito. El lenguaje se obtiene a partir de las sucesiones admisibles de eventos que se generan por la propia estructura del DES. Los autómatas se representan como un grafo dirigido, como el que se puede apreciar en la figura 3.5. Para continuar, es necesario establecer una definición más formal de lo que es un autómata de estado finito, y recurrimos al concepto de una máquina de estados que acepta un conjunto particular de palabras sobre un alfabeto $\Sigma$. Esta definición estándar se puede consultar en el texto de Cassandras y Lafortune [16].

**Definición 3.4. Autómata de Estado Finito** Una máquina $G$ de estados finitos es una quíntupla

$$G = (X, \Sigma, f, x_0, X_m) \tag{3.5}$$

Donde:

$X$ es un conjunto finito de estados denotados por $x$, $x$ es discreto.

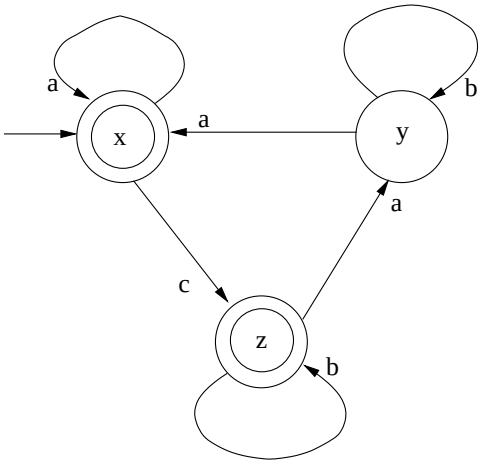


Figura 3.5: Autómata de Estado Finito

$\Sigma$ es un conjunto de eventos de entrada, denotado cada uno con una letra de un alfabeto.

$f : X \times \Sigma \mapsto 2^X$ es la función de transición o evolución del autómata

$x_0 \subset X$ es el estado inicial

$X_m \subset X$ es el conjunto de estados deseados de $G$ (estados marcados)

El alfabeto de entrada $\Sigma$ es el conjunto de todos los eventos que pueden ocurrir en esa máquina de estados, ocasionando un transición desde el estado actual a otro estado de la máquina. Para representar adecuadamente la dinámica de un proceso industrial, se requiere que el autómata $G$ sea además determinístico, es decir: $f(x, e) \in X$ es única para cada pareja $(x, e)$, y además exista un solo estado inicial $x_0$. Este concepto se tratará mas adelante, cuando se introduzca el concepto de DES controlado, o DESC. Además del grafo dirigido, una máquina de estados finitos se puede representar por una *tabla de transición de estados*, lo que facilita su implementación computacional. Cuando se considere la función $f$ definida solamente para algunas parejas $(x, e)$, $f$ se establece como una *función parcial*. Los autómatas que tienen esta función parcial se denominan *Generadores*, con los cuales se van a establecer las siguientes definiciones y propiedades.

**Definición 3.5. Lenguaje generado de $G$.** Se denota por $L(G)$, y $L(G)$ equivale a la expresión $\{s \in \Sigma^* | f(x_0, s)!\}$. El carácter parcial de la función de transición de los generadores hace que el lenguaje generado sea un subconjunto de $\Sigma^*$. Dada esta definición de lenguaje, es cerrada con respecto a los prefijos (las secuencias que permiten llegar a un estado determinado), así, $L(G) = L(\bar{G})$ Por analogía, también se puede establecer el *Lenguaje Marcado* $L_m(G) \subseteq \Sigma^*$, donde $f(x_o, s) \in X_m$. A partir de estas definiciones se establece que $L_m(G) \subseteq L(G)$, y representan lenguajes asociados a un DES.

Considerando estas definiciones de lenguajes, se pueden ordenar los lenguajes de acuerdo a su extensión, de la siguiente forma:

$$L_m(G) \subseteq L_m\bar{(G)} \subseteq L(G) = L(\bar{G}) \tag{3.6}$$

De acuerdo a su diseño, un generador puede tener estados inaccesibles, es decir, estados a los que no se puede llegar partiendo del estado inicial. Las siguientes definiciones ayudan a establecer unas propiedades deseables de los generadores, para la representación de procesos secuenciales, como los que ocurren en la industria.

**Definición 3.6. Accesibilidad** . Un estado $x \in X$ es un estado accesible si se cumple que: $\exists s \in \Sigma^* | f(x_0, s) = q$. Un generador $G$ es accesible si $x$ es accesible para todo $x \in X$. Llevando este concepto a los procesos industriales, esto se interpreta como que se puede alcanzar un estado dado desde un estado inicial, con la ocurrencia adecuada de ciertos eventos. El componente accesible de un generador $G$ se obtiene por la eliminación de estados inaccesibles y las transiciones asociadas a estos.

**Definición 3.7. Co-Accesibilidad** . Un generador $G$ se considera co-accesible si cada palabra (secuencia) $s \in L(G)$ por un prefijo de una palabra en $L_m(G)$, o sea si y solo si $L(G) \subseteq \overline{L_m(G)}$. En otras palabras, el generador es co-accesible si a partir

de cualquiera de sus estados existe al menos una secuencia que lo lleve a un estado marcado. Esto se resume como:

$$L(G) = \overline{L_m(G)} \tag{3.7}$$

**Definición 3.8. Generador Trim** . Un generador $G$ se considera como generador *trim* si es accesible y co-accesible.

Si se cumple la ecuación 3.7 para un autómata dado, se concluye que el DES asociado a ese autómata es de *no-bloqueo*. Por lo tanto un DES será de bloqueo cuando a partir de una secuencia $u \in (L - L_m)$, no se podrá completar alguna tarea en el sistema, o no se podrá recuperar de una falla, por ejemplo.

### 3.3.2. Enfoque basado en Redes de Petri

Otro enfoque para representar la dinámica discreta del proceso industrial es el de utilizar Redes de Petri (o RdP, de aquí en adelante), ya que las especificaciones del un sistema de producción se representan en la estructura y dinámica de la Red de Petri. Existen diversas maneras de definir una red de Petri. Una de ellas establece que una red de Petri es un grafo bipartito (de dos tipos de elementos) dirigido, con una regla de evolución y un "estado inicial". Los dos tipos de elemento son: las *transiciones*, que representan el comienzo o fin de una actividad, y se representan por medio de una barra. El otro tipo de elemento son los *lugares*, que representan una operación, o un estado del proceso. Las marcas, conocidos también como la marcación, es un número entero asignado a cada uno de los lugares, y representan una disponibilidad o no disponibilidad para el status de un recurso, o una cantidad para una operación. Una representación gráfica de una red de Petri se observa en la figura 3.6, donde se muestran los lugares $p_1, p_2$ y $p_3$, al igual que las transiciones $t_1$, $t_2$ y $t_3$ con sus respectivos arcos. Una marcación inicial se aprecia en el lugar $p_3$, el cual contiene

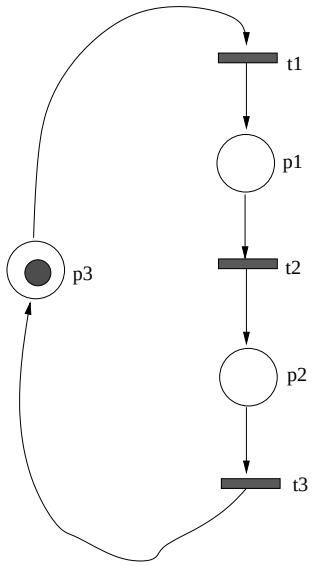


Figura 3.6: Ejemplo de una red de Petri

exactamente una marca.

Otra definición de redes de Petri es la formal, orientada hacia el análisis matemático de modelos concurrentes, y se basa en la teoría de conjuntos. A partir del modelado formal de una red de Petri pueden determinarse propiedades de esta red, las cuales se pueden proyectgar a las propiedades de un sistema de producción.

**Definición 3.9. Red de Petri** Una Red de Petri es una quíntupla $PN = (P, T, F, W, M_o)$ donde:

$P = \{p_1, p_2, \ldots, p_n\}$ es un conjunto finito de lugares

$T = \{t_1, t_2, \ldots, t_m\}$ es un conjunto finito de transiciones

$F \subset (P \times T) \cup (T \times P)$ es un conjunto de arcos que relacionan lugares con transiciones y viceversa

$W : F \rightarrow \{1, 2, 3, \ldots\}$ es una función de asignación de peso (ponderación)

$M_0 : P \rightarrow \{1, 2, 3, \ldots\}$ es la marcación inicial

$P \cap T = \oslash$

$P \cup T = \oslash$

Una Red de Petri que considera solamente su estructura como la cuádrupla $N = (P, T, F, W)$ sin ninguna marcación inicial, se denota por N, mientras que si se considera la marcación inicial la denotación es $(N, M_0)$. Existen otras definiciones formales de Redes de Petri, donde las funciones mapean explícitamente transiciones a lugares y viceversa como lo establece Peterson en [90], en lugar de definir conjuntos de arcos y funciones de asignación de peso.

La otra definición de una red de Petri es matricial, donde para los arcos que relacionan lugares y transiciones [80]. Una matriz $E$ de dimensión $n \times m$ (n es el número de lugares y m la cantidad de transiciones) representa las entradas a las transiciones. Si un elemento $e_{ij}$ es igual a 0, indica que no existe un arco que comunique al lugar $l_i$ con la transición $t_j$. Si existe el arco el valor de ese elemento será el "peso" del arco. De la misma manera, se define una matriz $S_{n \times m}$ como la matriz de salida de las transiciones hacia los lugares, y una matriz $D$, que se denomina *Matriz de incidencia*, y que resulta de la operación matricial $S - E$. De aquí en adelante, para referirnse a una red de Petri se utilizará su matriz de incidencia $D$. La marcación de la red se denota por un vector $M$, el cual contiene en su elemento $M_i$ la cantidad de marcas asignadas al lugar $i$. Una función para denotar la marcación en un lugar es $M(p)$. El motivo para utilizar esta notación matricial es que facilita su tratamiento computacional, tanto para el análisis como para posteriores operaciones, como lo es la síntesis de un DES que actuará como "supervisor" de un sistema dado.

Diversos investigadores han desarrollado metodologías para construir una RdP a partir de las especificaciones de un sistema de producción, destacándose el trabajo de DiCesare y Zhou donde a partir de las especificaciones de un proceso de manufactura diseñan una RdP [30, 115]. Su propuesta se basa en la clasificación de los lugares de la RdP en tres tipos: lugares que representan el estado de una operación, lugares que representan entidades temporales, como lo son las piezas a procesar, y lugares que representan las entidades permanentes, como lo es el equipamiento. Johnsson [58]

propone una metodología para aplicar redes de Petri en el modelado de sistemas de producción por lotes. Hsieh [53] utiliza redes de Petri en la representación de sistemas de manufactura holónica basada en los holones Recurso y Producto de la arquitectura PROSA. Incluso propone una manera de generar holarquías utilizando el protocolo multi-agente Contract Net (CNP), y las representa por medio de la composición de redes de Petri.

Para un proceso de producción continuo hay una propuesta para modelar la conducta de una unidad de producción hecha por Chacón et al [18], donde los lugares de una RdP pueden representar el estado del proceso (la operación como la propone Zhou y Dicesare), los recursos y los métodos de operación. Una marcación en los lugares que corresponden a los métodos de operación pueden indicar que el método está disponible, por ejemplo.

Las redes de Petri pueden clasificarse de acuerdo a sus propiedades, las cuales se pueden agrupar en dos tipos [80]: *propiedades estructurales*, que no dependen de la marcación inicial, y propiedades que dependen de la marcación inicial, las cuales se denominan *propiedades conductuales*. En la descripción de procesos industriales, es deseable que el comportamiento del proceso exhiba ciertas propiedades, las cuales se van a reflejar en las propiedades de la red de Petri. Según Dicesare, estas propiedades deben ser: red de Petri limitada, viva, reversible, consistente y conservativa [30]. La segunda propiedad indica que el sistema no se va a bloquear, en tanto que la tercera indica que cualquier proceso industrial se puede reinicializar. Una red conservativa implica que es limitada, mientras que toda red reversible es consistente. Una red consistente se define como aquella red donde hay una secuencia de disparos que llevan a una transición a si misma, disparando las demás transiciones, de manera que el proceso de manufactura es re-inicializable. Una definición en detalle de cada una de estas propiedades se establece a continuación.

**Definición 3.10. Alcanzabilidad** Es una base fundamental para estudiar las propiedades dinámicas de un sistema. El disparo de una transición cambiará la distribución de la marcación en una red de acuerdo a una regla de transición que se establece de la siguiente manera: $\mu_i = \mu_{i-1} + D * L$. Donde $D$ es la matriz de incidencia de la red, $L$ es el vector que indica cual transición se dispara. Una secuencia de disparos se denota como $t_1, t_2 \ldots t_n$.

**Definición 3.11. Red de Petri limitada** Una red de Petri $(N, M_0)$ se dice que es limitada si la cantidad de marcas en cada lugar no sobrepasa una cantidad finita $k$ para cualquier marcación alcanzable desde $M_0$. Si se cumple que $M(p) \leq 1$ para todo lugar $p$ y toda marcación $M$ entonces la red de Petri además es *segura*. En procesos industriales, los lugares se utilizan a veces para representar buffers para almacenar elementos o datos. Si la red es limitada o segura, es garatía de que no habrá desbordamiento de las capacidades de los buffers, sin importar que secuencia de disparos ocurran.

**Definición 3.12. Red de Petri viva** Una red de Petri $(N, M_0)$ es viva si es posible disparar *cualquier transición* en la red, sin importar que marcación se haya alcanzado desde $M_0$. Si un proceso industrial está representado por una red de Petri viva significa que la operación estará libre de bloqueos.

**Definición 3.13. Red de Petri reversible** Se dice que una red de Petri $(N, M_0)$ es reversible si para cada marcación $M$, $M_0$ es alcanzable desde $M$. De esta manera, un proceso industrial representado por una red reversible implica que se puede llevar a su estado inicial, un ejemplo de ello es la recuperación y corrección de fallas.

Zhou [115] recomienda que dadas las especificaciones de un sistema de manufactura, modelar el sistema como una red de Petri tal que su estructura y marcación inicial la hagan limitada, viva y reversible. Para una mayor referencia en cuanto a las

propiedades de las redes de Petri, se recomienda consultar las referencias de Murata [80], a manera de conocer las propiedades de las redes de Petri, y Dicesare, Zhou *et al* [30, 115] para establecer los principios de la *Exclusión Mutua Paralela* en los sistemas de manufactura.

Posteriores estudios han extendido las redes de Petri para abarcar modelos continuos, por medio de las *redes de Petri híbridas*, y procesos en paralelo como las redes de Petri coloreadas.

## 3.4. Sistemas Dinámicos Acoplados

A medida que aumenta la complejidad de un sistema, aumenta la complejidad de la representación de su conducta. Si determinado proceso industrial puede describirse como un SDH, se puede extraer su comportamiento discreto para describirlo como un DES. De manera que cuando este proceso industrial es parte de, o está acoplado con otros procesos, obteniendo así uno mas complejo, entonces se expresa formalmente la dinámica de este sistema como una composición de DES.

$$\Phi^c(x_0, u) = \Phi_k(\Phi_{k-1}(\ldots \Phi_2(\Phi_1(x_0, t_1), t_2) \ldots, t_{k-1}), t_k) \tag{3.8}$$

Si este DES lo describimos por medio de una Máquina de Estado Finito, entonces la composición de DES produce un autómata ampliado que resulta del *Producto Asíncrono* entre varios autómatas. Utilizando redes de Petri, puede obtenerse una red global por medio de la unión de las redes de petri de cada sub-proceso. Un esquema que muestra los componentes interconectados es el que aparece en la figura 3.7, como aparece planteado por Chacón, Khodr y De Sarrazin. Las funciones $\Phi$ corresponden a las funciones $U_j$, que un coordinador ejerce sobre cada subsistema. Cada uno de estos subsistemas tiene a su vez unas funciones $R_j$, las cuales son sus entradas, y las salidas de este subsistema además de percibirlo el coordinador, lo envían a los otros

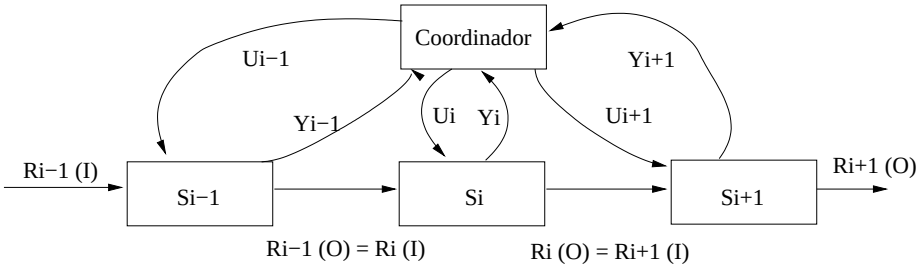


Figura 3.7: Sistema con componentes interconectados

subsistemas, a manera de entrada.

El Sistema Complejo (SC) de la figura 3.7 es una colección $\{S_i\}$ de subsistemas conectados uno con otro. El SC tiene una dinámica multiple, y cada subsistema se describe por medio de ecuaciones diferenciales o algebráicas. Una ecuación típica para un subsistema $S_i$ puede ser la siguiente:

$$\dot{x_i} = f_i^*(x_i, u_i^*, inter_i) \tag{3.9}$$

donde:

$x_i \in X_i \subset \Re^m$ es el espacio de estados del subsistema

$f_i \in C^k(X_i)$ función continua en el dominio de $X_i$

$Y_i$ es el conjunto de valores de salida: $y_i = h_i(inter_i)$

$u_i \in S_{\Re+}(U)$ es el conjunto de controles

$inter_i$ es la conexión entre $r_i^I$ (entrada para $S_i$ y salida para $S_{i-1}$) y $r_i^O$

De una manera conceptual, la arquitectura de control de un SC es una composición de varios subsistemas que contienen controladores continuos. La composición de estos sistemas controlados generan un nuevo sistema, el cual a su vez puede componerse con otros sistemas y así obtener un sistema global. La conducta de este sistema global puede describirse por medio de un DES, el cual es adecuado para tratar con el cambio de modos de operación, decisión, fallas y supervisión. Un sistema de control clásico se puede considerar como un Sistema Dinámico Acoplado, de acuerdo a la propuesta

de Chacón, DeSarrazin y Khodr [20], por lo tanto, el control de un sistema holónico se podría considerar como una composición de DES. La definición de un SDA se establece a continuación, y se utilizará luego para explicar la dinámica de un sistema holónico.

**Definición 3.14. Sistema Dinámico Acoplado** Un SDA es una tripleta $(\bar{X}^1, \bar{X}^2, F)$ donde: $\bar{X}^i$ es un sistema pseudo-dinámico [20] y $F = (f, h)$ es una proyección que preserva la dinámica, tal que $f(\theta_1) = \theta_2$.

En nuestro caso, $\theta_1$ corresponde a un segmento de números reales, que corresponden a las magnitudes físicas del proceso industrial que se quiere controlar. $\theta_2$ está asociado a un estado particular en que se encuentra el sistema de producción, y $f$ es una función que proyecta de un conjunto de valores continuos a unos valores discretos, tal y como está definido para unos sistemas híbridos.

## 3.5. La conducta de un Sistema Holónico

A los sistemas de manufactura concebidos bajo el paradigma holónico también permiten describir su comportamiento como una composición de DES, puesto que cada unidad de producción es independiente y controla su propio proceso físico, el cual puede describirse como un sistema híbrido. A partir de los tres holones básicos que conforman una Unidad de Producción autónoma según la arquitectura PROSA se puede proyectar el *estado* de la misma, el cual es un estado compuesto por las dinámicas de los recursos (holones recurso) y del proceso (relación entre orden y recurso). La secuencia de varios estados en el tiempo se denomina una *trayectoria*, y es el resultado de la aplicación de varias entradas de control $\Phi$. Es conveniente recordar que una trayectoria está delimitada por una secuencia de eventos, la cual ha sido establecida por el sistema supervisor, aunque esto no excluye que el supervisor deshabilite algunos de ellos para que ocurran en el orden deseado. De una manera formal, la conducta de

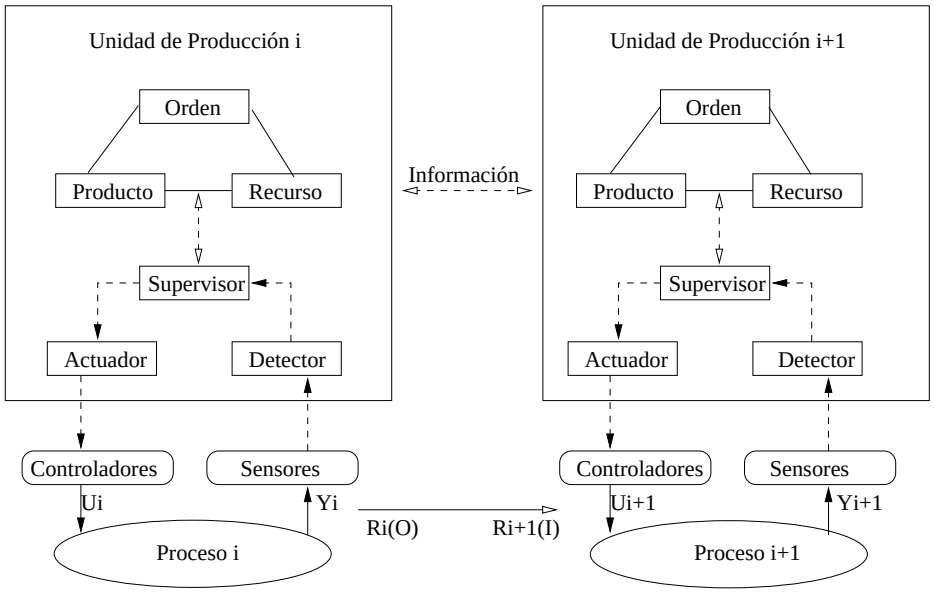


Figura 3.8: Sistema holónico es un SDA

un Sistema Holónico tiene la siguiente expresión formal:

$$\Phi_k(\Phi_{k-1}\ldots\Phi_1(\Phi_0(X_R, X_P, t_0, t_1), t_1, t_2)\ldots, t_k, t_{k+1}) \tag{3.10}$$

Donde los $X_R$ y $X_P$ corresponden a los estados de cada recurso utilizado, y cada proceso que se está ejecutando, respectivamente. Las funciones $\Phi_i$ son las entradas de control que se introducen al sistema para obtener el comportamiento deseado, durante el tiempo comprendido entre $(t_i, t_{i+1})$. Estas entradas de control están asociadas a la configuración de la unidad de producción durante ese intervalo, en particular de cada equipamiento físico que la contenga. En la figura 3.8 se muestran unas unidades de producción concebidas como sistemas holónicos, cada una de ellas controlando un proceso físico, cuyas variables se miden por medio de sensores. Cada unidad de producción se refleja en un conjunto de objetos que forma un sistema holónico, y dispone de su propio mecanismo de supervisión, proporcionándole autonomía. La asimilación de estas unidades de producción a un SDA hace de la siguiente manera:

- El proceso I le proporciona material al proceso I+1, esto se refleja en la igualdad

$R_i(O) = R_{i+1}(I)$, es decir, el producto de salida del proceso I es un insumo de entrada para el proceso I+1.

- Las salidas $Y_i$ y $Y_{i+1}$ son el resultado de la medición de variables físicas asociadas a cada proceso, y se llevan al detector de eventos de cada unidad de producción. Este a su vez proporciona al supervisor información sobre los eventos ocurridos, el cual decide variar la configuración del sistema holónico, expresada como una relación entre los recursos y el método de producción. De esta configuración se hablará en la sección de modelado conceptual, mas adelante.

- Los cambios en la configuración se manifiestan como comandos del actuador a los controladores físicos, que son los únicos que varían el estado del equipamiento, incidiendo así sobre el proceso y esto se asocia a las entradas de control $U_i$ y $U_{i+1}$ para los procesos I e I+1 respectivamente.

- Aparte del intercambio de material, ambas unidades de producción intercambian información tendiente a negociar capacidades de producción, sincronizar actividades o asignar tareas, de una manera colaborativa. La gestión de esta colaboración se mostrará más adelante, en las secciones de gestión del holón y modelado conceptual.

### 3.5.1. Conducta del Sistema Holónico expresada como un DES

Para aplicar los principios de control supervisorio sobre un HMS es necesario que la conducta de este sistema que usualmente se expresa por medio de un sistema híbrido se proyecte a un sistema de eventos discretos. Como se había establecido anteriormente, existen dos enfoques de DES para describir la dinámica discreta del HMS: con autómatas y con redes de Petri.

Para describir una unidad de producción holónica bajo el enfoque de autómatas es necesario llevar a cabo las siguientes actividades:

1. Representar por medio de un autómata los estados y transiciones asociados a cada equipamiento (Recurso). Se obtendrían los autómatas $R_1$, $R_2$, ... ,$R_N$. $N$ es la cantidad de equipamiento que pertenece a la unidad de producción. Definir el estado inicial para cada autómata.

2. Representar el proceso por medio de etapas, y el paso de una etapa a otra asociarlo a una transición. Así se obtendría el autómata $P$. Definir el estado inicial del proceso.

3. Hallar el autómata $U$ que representa a toda la unidad de producción, por medio de los productos asíncronos de los autómatas: $U = R_1||R_2|| \ldots ||P$.

4. Establecer el estado inicial del autómata $U$.

Para el caso de expresar la dinámica de un HMS utilizando redes de Petri, el procedimiento sigue la propuesta de Chacón, Besembel y Hennet [18] donde los lugares de la RdP representan el estado del proceso, los recursos y los métodos de operación utilizados. La figura 3.9 muestra un esquema simplificado de la dinámica de una unidad de producción, donde se observa por un lado la dinámica acoplada del proceso, con la de los recursos, y un método de producción que habilita esta dinámica compuesta. Una marcación en los lugares que corresponden a los métodos de operación pueden indicar que el método está disponible, mientras que una marcación en los lugares correspondientes a los recursos indicará que determinados recursos están disponibles para su operación. Un ejemplo de representación de la dinámica de un sistema holónico utilizando redes de Petri se muestra en el anexo C de este documento.

No obstante, es necesario indicar que las transiciones en un HMS deben detectarse a partir de los valores continuos que se miden a partir del proceso físico a controlar, y esto es válido para ambos tipos de representación. Este procedimiento se denomina *Detección de eventos* y será explicado a continuación.

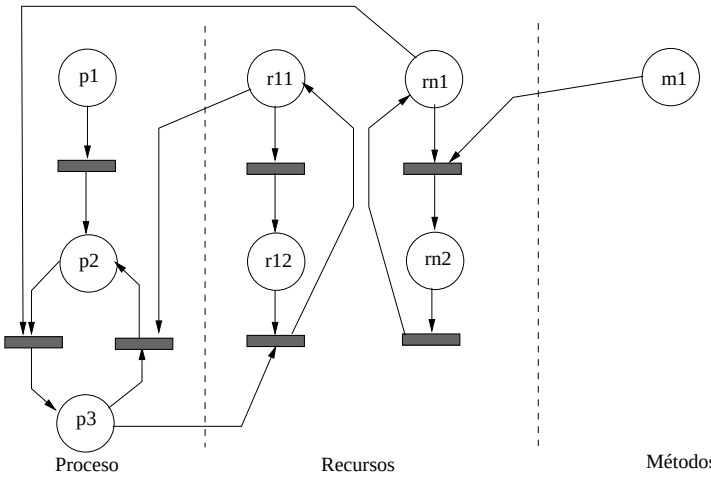


Figura 3.9: Esquema de red de Petri para describir la conducta de un HMS

## 3.6. Detección de eventos

Una condición para que funcione el Control Supervisorio está relacionada con la determinación correcta del estado en que se encuentra un determinado proceso, así como la identificación correcta del evento que implica un cambio de estado. Si se deja pasar un evento o se identifica incorrectamente, el Control Supervisorio generará cambios en la consigna de los controladores sin que haya motivo, o simplemente ignorará el cambio de una región de operación a otra. De manera que es importante en el diseño de un sistema supervisor que se tenga en cuenta un mecanismo adecuado de detección de eventos. La detección de eventos se implementa principalmente sobre las variables continuas, cuando se miden por medio de algún dispositivo. Aunque esto no excluye la detección de eventos en variables discretas de la Unidad de Producción, como lo es el encendido de un dispositivo, la llegada de una nueva orden de producción, a manera de ejemplos.

Existen diversos mecanismos para llevar a cabo la detección de un evento, desde la observación directa del valor de una variable discreta, hasta la aplicación de técnicas de inteligencia artificial como lógica difusa o redes neuronales. La tabla 3.1 muestra como se ubican estos métodos, los cuales serán explicados a continuación.

Tabla 3.1: Ubicación de los métodos de detección de eventos

| Naturaleza de Variable | Observable | No observable |
|---|---|---|
| Discreta | Detección directa del valor | Observar otras variables |
| Continua | Reglas lógicas de tipo if-then else, Funciones de abstracción, Lógica difusa | Observar otras variables continuas |

La conducta de los dispositivos dentro de una Unidad de Producción se expresa por medio de estados discretos, y los eventos consisten en el cambio de estos estados. Para detectar eventos con estas variables discretas observables basta con observar su valor, o revisar la configuración de la planta cuando el supervisor o el actuador emiten un comando que ejecuta un accionamiento físico. Cuando estas variables no pueden observarse directamente, hay que revisar su efecto sobre otras variables para deducir que ha ocurrido un evento.

Cuando se trata de variables continuas observables directamente es necesario aplicar alguna regla que permita deducir la ocurrencia de un evento, análogo a la proyección de un valor continuo hacia un estado discreto, como en los sistemas híbridos [69, 18]. El caso mas sencillo se da cuando se ubica el valor de la variable continua dentro de un intervalo, donde ese intervalo corresponde a un estado discreto, y el cambio de un intervalo a otro se asocia a la ocurrencia de un evento. La figura B.2 muestra un ejemplo sencillo para ubicar una variable continua (presión) en tres posibles valores discretos: baja, media o alta. Este caso no es el más recomendable, debido a que en muchas situaciones la medición de una magnitud por un dispositivo sensor está sujeta a variaciones, bien sea por la naturaleza fluctuante del proceso a medir, o por las características intrínsecas del sensor. Estas fluctuaciones pueden inducir a una detección errónea de eventos, o también a dejar pasar el evento sin detectarlo. Un ejemplo de medición con fluctuaciones es la de medir el nivel de líquido en un

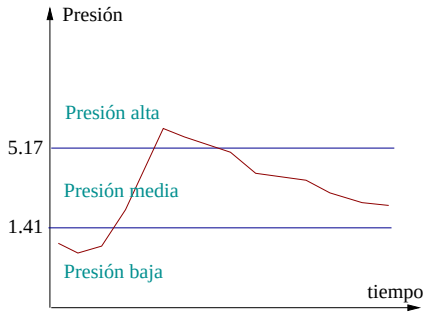


Figura 3.10: Ejemplo de mapeo directo de variable continua a discreta

tanque abierto, el cual presenta oleaje debido al flujo de entrada. Cuando la variable no se puede observar, entonces debe deducirse la ocurrencia del evento a partir de observaciones de otras variables que estén relacionadas.

Una estrategia adecuada de proyección consiste en examinar tendencias sobre un conjunto de datos, en lugar del valor del dato individual. Un método de agrupación denominado *ventana deslizante* o $\otimes$, fue diseñada por Sarrate y Aguilar [99, 100], en su trabajo sobre vigilancia y control inteligente de procesos. Este método de muestreo funciona de la siguiente manera: una ventana agrupa un conjunto de muestras contiguas que provienen de la medición de una variable determinada, la cual se caracteriza por un período $T$ de muestreo. Los parámetros de la entrada son dos: una *anchura* $a$ que indica el intervalo temporal que abarca la ventana (y por ende cuantos datos agrupa), y un *desplazamiento* $d$ que indica el período de muestreo de la ventana, expresado en múltipos de $T$. En la figura 3.11 se aprecia un ejemplo de ventana deslizante con todos sus elementos. Para esa ventana, el valor de $a$ indica la cantidad de datos por agrupar, y $d$ indica cada cuantos datos inicia una ventana (la frecuencia).

Para el análisis de una ventana deslizante se produce un dato que caracteriza esta ventana, denominado el *atributo* $S$. Un ejemplo de este atributo puede ser el promedio de los valores de la variable medida, o también su pendiente (la primera derivada), transformada de Fourier u otras funciones aplicadas sobre este conjunto de valores. Las reglas de detección de eventos se aplicarían entonces sobre los atributos, y luego

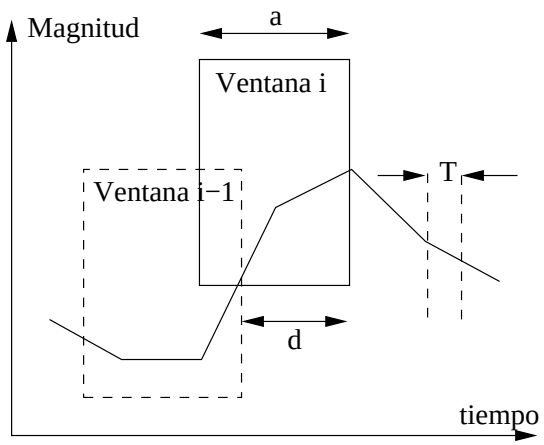


Figura 3.11: Ejemplo de ventana deslizante

se proyectarían hacia un estado discreto. El cambio de estado discreto estará asociado entonces a la ocurrencia de un evento. Un ejemplo de ventana deslizante se muestra en el anexo D de este documento.

En ciertas ocasiones no es fácil llevar a cabo la proyección directa de las variables continuas hacia un estado discreto, en especial cuando es de naturaleza cualitativa y no se tiene claridad sobre los rangos de valores que corresponden a cada variable de naturaleza linguística (cualitativa). Para ello se recurre a los métodos derivados de la lógica difusa, de acuerdo a la propuesta original de Zadeh [111], en particular el modelo constructivo de Mamdani [70], que establece la proyección de variables continuas a variables linguísticas (cualitativas). En la figura 3.12 se puede observar como opera el modelo constructivo de Mamdani, aplicado a una variable continua (presión), y tres funciones de membresía, cuya naturaleza es probabilística. Es decir, cuando la variable está en cierto valor, existe una función de probabilidad asociada a cada variable discreta. En el ejemplo, $f_1(x)$ representa la membresía para considerar a una presión baja, y es de 1 (certeza total) cuando la presión es menor a 1.25, y luego es una función lineal mientras la presión esté entre 1.25 y 1.55, hasta llegar a cero para valores de presión mayores a 1.55. El mismo concepto se aplica para las otras funciones. La cualificación de la variable se da por la evaluación de cada una de

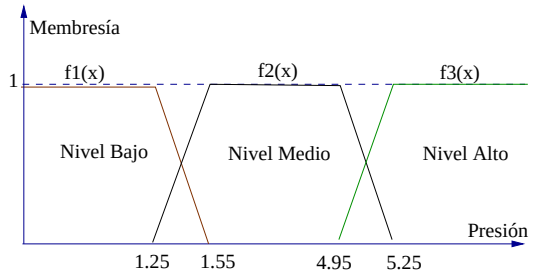


Figura 3.12: Ejemplo de proyección de variable continua a linquística, con lógica difusa

las funciones de acuerdo al valor continuo. La función con probabilidad mas alta es la que corresponde al valor cualitativo asignado a la variable continua. Estas funciones de membresía pueden ser lineales, como la de la figura, o también pueden tener otra definición, como las funciones sigmoideas, por ejemplo.

Un mecanismo de detección de eventos es fácil de implementar, y acoplar a un Sistema de Control Supervisorio. Es necesario disponer de un software que reciba las lecturas de las variables asociadas al sistema, y junto con el estado actual en que se encuentra el proceso industrial deducir si ha ocurrido un cambio de estado, al que se le pueda asociar la ocurrencia de un evento.

## 3.7. Conclusión

La dinámica de una unidad de producción se puede representar como un conjunto de sistemas dinámicos acoplados, los cuales pueden exhibir comportamiento tanto en variable continua como en variable discreta. Para poder representar y simular este comportamiento, es indispensable tener un mecanismo de proyección de variable continua a discreta, y un mecanismo de detección de eventos que permita establecer si ocurrió algún cambio entre los estados discretos en que se puede estar un proceso determinado, independiente de la manera en que se conciba el sistema a eventos discreto: bien utilizando autómatas o bien utilizando redes de Petri.

# Capítulo 4

# Mecanismos de Supervisión para un Sistema Holónico de Manufactura

En las secciones anteriores se habían planteado la descripción de una unidad de producción bajo el enfoque de automatización basado en holones PROSA, y la descripción de su dinámica utilizando sistemas dinámicos en variable continua acoplados a sistemas a eventos discretos. En esta sección se van a tratar los aspectos matemáticos y conceptuales sobre cómo controlar el comportamiento de un HMS basado en la arquitectura de referencia PROSA, utilizando mecanismos de Control Supervisorio basados en sistemas a eventos discretos DES, así como también aspectos relacionados con su implementación. Estos mecanismos van a actuar sobre el comportamiento dinámico del HMS, previniendo que ocurran situaciones no deseadas o que las variables del proceso se mantengan en las condiciones de operación nominales. Para explicar los enfoques utilizados en la síntesis de sistemas supervisores, se presentan las siguientes definiciones:

**Definición 4.1. Planta.** Es el sistema a controlar, incluye el proceso productivo físico, los insumos a utilizar, el método de producción y los controles a nivel de equipamiento (PID). Ogata [83] la define como una pieza de equipo, un conjunto de

partes de máquina cuyo propósito es realizar una función particular. Todo objeto físico sometido a control.

**Definición 4.2. Supervisor.** Es un controlador que permite que la *planta* evolucione libremente, actuando únicamente para impedir que se llegue a un estado no deseado.

**Definición 4.3. Sistema de Control Supervisorio.** Es un sistema de eventos discretos (DES) que es regulado por el Supervisor. Puede representarse como un autómata finito determinista, una red de Petri, o como un conjunto de reglas basadas en lógica.

**Definición 4.4. Síntesis del Supervisor.** Es el proceso mediante el cual se obtiene un DES que va a controlar la conducta de un sistema, también expresada por medio de DES. Bajo ciertos enfoques, se trata de extender un DES con el comportamiento controlado.

Con las anteriores definiciones, y dado que una planta puede describirse matemáticamente como un DES y que por lo tanto esta planta es un *generador de lenguaje*, se puede afirmar que el Control Supervisorio es una construcción basada en teoría de lenguajes, donde un lenguaje de un proceso industrial se define como una sucesión de eventos $\{\delta_1, \delta_2, \ldots, \delta_n\} \in L$, de posible ocurrencia dentro del proceso, y el supervisor es una función $h : L \rightarrow \Phi$, que asocia a cada cadena de eventos generados $w \in \Phi$, un subconjunto de entradas de control $\phi = h(w) \in \Phi$. Estas entradas de control inhiben eventos que lleven el sistema controlado hacia un estado no deseado. Su implementación física es relativamente sencilla, y puede resolverse a través de un sistema de control conmutado descrito en los sistemas de ingeniería de control moderna.

La Supervisión se puede llevar a cabo de diversas maneras, aunque existen dos enfoques fundamentales: uno utiliza Máquinas de Estado Finito generadoras de lenguajes,

propuesto por Wonham y Ramadge [93], y otro se basa en Redes de Petri, propuesto por Moody y Antsaklis [76]. Aunque existen propuestas, bajo enfoques diferentes, como la de Caines y Wang [12], establecida en el reporte COCOLOG, basada en árboles de predicados de Lógica de Primer Orden, para describir la lógica de observadores y controladores condicionales. Los autómatas o máquinas de estado finito se basan en las restricciones al lenguaje de la planta, y su método consiste en duplicar el modelo de la planta para el controlador, eliminando los estados no deseados por medio de la deshabilitación de transiciones controlables. Las Redes de Petri funcionan limitando el comportamiento a lazo abierto de la planta, ejecutando predicados lineales en los vectores de estado, los cuales se basan en restricciones lineales que representan marcaciones prohibidos. En esta sección se van a describir con más detalle como funcionan ambos enfoques.

## 4.1. Control Supervisorio basado en Máquinas de Estado Finito

El comportamiento de una planta industrial descrito mediante un DES, como lo es un autómata, puede controlarse por medio de un *supervisor*. Para obtener este supervisor existen varios enfoques basados en autómatas. Una teoría clásica de control supervisorio la proponen Ramadge y Wonham, donde establecen un *supervisor mínimamente restrictivo*, el cual observa e inhibe eventos. Esta teoría se extendió posteriormente por Golaszewski y Ramadge, introduciendo el concepto de eventos forzables desde un supervisor [47]. Para tratar con procesos industriales mas complejos, Cury propone el concepto de *control modular*, el cual lleva al concepto de composición de supervisores, adecuado para sistemas holónicos que tienen una perspectiva modular también.

### 4.1.1. Enfoque clásico de Ramadge y Wonham

Un enfoque para diseñar o sintetizar Sistemas de Control Supervisorio es el propuesto por Ramadge y Wonham [93], donde establecen que un supervisor se considera como un autómata de estado finito determinista $S$, al igual que la planta que se desea controlar, que es el autómata $G$. El supervisor se relaciona con la planta a través de la siguiente función:

$$S : L(G) \rightarrow \Gamma \tag{4.1}$$

Donde $L(G)$ denota el lenguaje generado por la planta, el cual se asocia a una secuencia de eventos, y $\Gamma \subseteq 2^{\Sigma}$ es el conjunto de las entradas de control. El supervisor S se define de tal manera que duplica al autómata G, sin los estados no deseados. Sus eventos están definidos de tal manera que permiten o inhiben los eventos de G equivalentes. Si se formaliza la planta G se obtiene la siguiente estructura:

$$G = (Q, \Sigma, \delta, \Gamma, q_0, Q_m) \tag{4.2}$$

Donde $Q$ es el Conjunto de estados de la planta, $\Sigma$ es el Conjunto de eventos, $\delta : Q \times \Sigma \rightarrow Q$ es la función de transición de estados, $\Gamma$ es el conjunto de transiciones permitidas, $q_0$ es el estado inicial, $Q_m$ es el conjunto de estados deseables. El conjunto de eventos de un DES puede particionarse de la siguiente manera:

$$\Sigma = \Sigma_c \cup \Sigma_{nc} \tag{4.3}$$

Donde:

$\Sigma_c$ es el conjunto de eventos controlables, los cuales pueden inhibirse.

$\Sigma_{nc}$ es el conjunto de eventos no controlables, sobre los cuales no se pueden inhibir de ninguna manera.

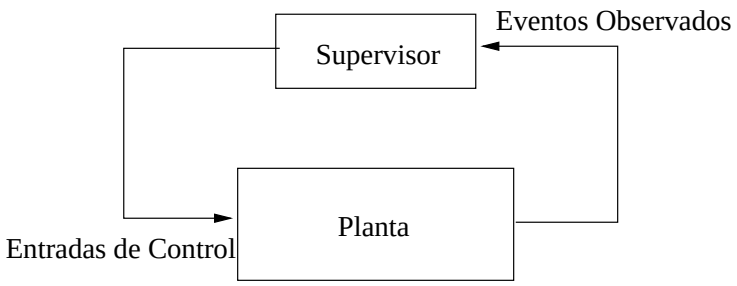


Figura 4.1: Esquema de control de un DES a lazo cerrado

Por lo tanto, dado un generador $G$, el *conjunto de entradas de control* asociadas a $G$ está dado por $\Gamma = \gamma | \Sigma_{nc} \subseteq \gamma \subseteq \Sigma$. La condición $\Sigma_{nc} \subseteq \gamma$ indica que los eventos no controlables necesariamente deben estar habilitados.

El control de un DES no solo consiste en aparear entradas de control, en función del comportamiento pasado del sistema. Por analogía con la teoría de control, el comportamiento controlado se denomina *comportamiento a lazo cerrado*. Para ello, es necesario distinguir el sistema a controlar o planta, del Supervisor, cuyas definiciones se mostraron al principio de esta sección. Un esquema de DES con comportamiento a lazo cerrado se muestra en la figura 4.1. El supervisor recibe los eventos generados en la planta, y emite unas entradas de control para generar una secuencia deseada de eventos.

Con los conceptos antes mencionados, al igual que el esquema de supervisión basado en entradas de control para un DES, se pasa a la definición formal de un Supervisor, y el comportamiento del sistema bajo la acción de este Supervisor.

**Definición 4.5. Supervisor** Un Supervisor denotado por $f$ es una proyección $f : L \rightarrow \Gamma$ que especifica para cada cadena posible de eventos generados por la planta denotada por $w \in L$, una entrada de control $\gamma = f(w) \in \Gamma$.

**Definición 4.6. Comportamiento de un sistema supervisado.** Un sistema a eventos discretos $G$ controlado por $f$ tiene un comportamiento definido por un lengua-

je $L(f/G) \subseteq L(G)$, donde se cumple que a) $\epsilon \in L(f/G)$ y b) $w\sigma \in L(f/G)$ *sii* $w \in L(f/G), w\sigma \in L(G),\ \sigma \in f(w)$.

**Definición 4.7. Comportamiento marcado de** $(f/G)$ Se define de la siguiente forma: $L_m(f/G) = L(f/G) \cap L_m(G)$. Este comportamiento consiste en una parte del lenguaje marcado original que permanece luego de la acción del Supervisor, es decir, las tareas que se pueden completar con supervisión.

El proceso para obtener el Supervisor, denominado síntesis, está encaminado a modificar el comportamiento de un DES, descrito por un lenguaje $L(G)$, preservando una propiedad de no-bloqueo. El problema de modelado del supervisor tiene dos tópicos: a) en términos del lenguaje marcado, encontrar un Supervisor de no-bloqueo, siempre que sea posible, de manera que el comportamiento a lazo cerrado satisfaga $K = L_m(h/G)$, donde el comportamiento deseado se especifica por medio del lenguaje $K \subseteq L_m(G)$ que representa las tareas deseadas; o b) en término de un lenguaje prefijo-cerrado, encontrando un Supervisor $h$ tal que el comportamiento a lazo cerrado satisfaga que $L(h/G) = K$, donde $K \subseteq L(G)$ que representa un comportamiento físicamente posible y deseado bajo la supervisión. Este comportamiento a lazo cerrado se obtiene de la composición de dos autómatas $S$ y $G$, de la siguiente manera:

$$S \times G = (Y \times X, \Sigma, \Theta, (y_0, x_0), Y_m \times X_m) \tag{4.4}$$

Donde $(S \times G)$ indica que las transiciones permitidas tanto en la planta controlada $G$ como en el Supervisor $S$ deben ocurrir simultáneamente, y además si una transición no está habilitada por el Supervisor, no ocurrirá en el DES que representa la conducta de la planta a controlar la cual se describe con $G$. El Supervisor se sintetiza entonces para encontrar a) un lenguaje $K \subseteq L_m(G)$ tal que el comportamiento a lazo cerrado satisfaga a $L_m(h/G) = K$. En caso de que no se pueda cumplir con este objetivo, entonces b) obtener $K \subseteq L(G)$ que represente el comportamiento físicamente posible

deseado bajo supervisión, y que satisfaga $L(h/G) = K$. Este lenguaje $K$ del sistema controlado debe cumplir con el criterio de *controlabilidad*, que se define a continuación:

**Definición 4.8. Controlabilidad** Un lenguaje $K \subseteq \Sigma^*$ es controlable con relación a $L(G)$ si se cumple que:

$$\bar{K}\Sigma_{nc} \cap L(G) \subseteq \bar{K} \tag{4.5}$$

Partiendo del hecho de que una planta con lenguaje $L(G)$ cumple con la condición de existencia y controlabilidad de un supervisor, el cual genera un lenguaje $K$, es necesario revisar si $K$ especifica a un comportamiento deseado. Si este comportamiento deseado tiene eventos no controlables, se le puede hallar un lenguaje controlable máximo $K\uparrow$, el cual cumple con la condición $K\uparrow \subseteq K$. Este lenguaje satisface la condición de existencia de un supervisor asociado, el cual se denominará Supervisor Mínimo Restrictivo (SMR de aquí en adelante). El algoritmo para sintetizar $K\uparrow$ se presenta a continuación, donde $K$ es un lenguaje regular no controlable.

**Algoritmo de síntesis de un Lenguaje Controlable Máximo** Sea un DES $G$ cuyo comportamiento se describe por $L(G)$ y $L_m(G)$. Sea $S$ un generador Trim tal que $L_m(S) = K \in L(G)$, $K\uparrow$ se obtiene así:

1. Se obtiene $G'$ como un generador Trim, tal que $L(G') = L_m(G') = L(G)$, para ello se asume que todos los estados de $G$ son marcados.
2. Sea $C_0 = G' \times S$.
3. Se identifican los estados no deseados de $C_i$.
4. Se obtiene $C_i'$ haciendo $C_{i+1} = Trim(C_i')$.
5. Si $C_{i+1} = C_i$ entonces $K\uparrow = L_m(C_i)$, de lo contrario se hace $i = i + 1$ y regresa al paso 3.

### 4.1.2. Enfoque clásico extendido con el concepto de eventos forzables

Otra clase de eventos, denominados "eventos forzables" se presenta a continuación, como una extensión al enfoque clásico de Ramadge y Wonham. El punto de partida de este enfoque consiste en particionar el conjunto de eventos $\Sigma$ en tres conjuntos disyuntos $\Sigma_n c$, $\Sigma_c$ y $\Sigma_f$. Donde el primer y segundo conjuntos corresponden a los eventos no controlables y controlables, respectivamente, mientras que los eventos del conjunto $\Sigma_f$ se interpretan como aquellos eventos que ocurren solamente si son forzados por una entrada externa. Se supone que este evento ocurre espontáneamente, anticipándose a la ocurrencia de otros eventos. Una implementación práctica la propone Torrico [105], por medio de un controlador de respuesta más rápida que el intervalo mínimo de generación de eventos en una planta por otros controladores. A manera de ejemplo, un computador de 2.4GHz es mas rápido que un PLC de 66 MHz, suponiendo que el PLC estuviera gobernando la conducta de la planta, y por lo tanto el computador puede generar eventos forzables.

**Entradas de control** El conjunto de entradas de control se modela entonces como:

$$\Gamma = \left\{ \gamma | \gamma \in 2^{\Sigma} \wedge \left[ \gamma = \{\sigma\} \;\; Para \; \sigma \in \Sigma_f \vee \gamma \in 2^{\Sigma_n c \cup \Sigma_c} \; Con \; \Sigma_n c \subseteq \gamma \right] \right\} \qquad (4.6)$$

Que se interpreta de la siguiente manera: existen dos tipos de entrada de control, uno que contiene los eventos forzables, y otro que contienen tanto los eventos controlables como no controlables, según el enfoque clásico. La introducción del concepto de evento forzable modifica la definición de controlabilidad, estableciéndola de la siguiente manera:

**Definición 4.8b. Extensión a la Controlabilidad.** Primero se establece el *conjunto activo* de $K$ después de "s" $(\Sigma_K(s))$, para cada $K \subseteq \Sigma^*$ y cada $s \in \bar{K}$ tal que $\sigma \in \Sigma | s\sigma \in \bar{K}$. Este conjunto activo se interpreta como el conjunto posible de eventos que puedan ocurrir después del estado $s$ de G. El lenguaje controlable se establece entonces como $K \subseteq L(G)$, con una estructura de control $\Gamma$ controlable si para cualquier $s \in \bar{K}$ existe una entrada de control $\gamma$ tal que:

1. $\sigma_k(s) \in \gamma$
2. $s\gamma \cap L(G) \subseteq \bar{K}$

El primer elemento asegura que esta entrada de control se pueda escoger de manera que permita todas las alternativas posibles en $K$. El segundo elemento es una condición de controlabilidad según el enfoque de Ramadge y Wonham, el cual asegura que exista una entrada de control para mantener la evolución de una secuencia de eventos en $K$. Un algoritmo que sintetiza el máximo lenguaje controlable $supC(K)$ contenido en $K$ cuando se introducen eventos forzables es propuesto por Torrico, como una extensión al algoritmo original de Kumar, y se presenta a continuación.

**Algoritmo de Torrico y Cury** Sea $K$ un lenguaje regular, $G$ un DES con lenguajes $L(G)$, $L_m(G)$, estructuras de control $c(\Gamma)$, $S$ es un generador trim tal que $L_m(S) = K \subset L(G)$ y el conjunto de eventos $\Sigma$.

1. Obtener $G'$, un generador trim tal que $L(G') = L_m(G) = L(G)$. Se marcan todos los estados de G.
2. Construir $C_i = G', i = 0$.
3. Identificar los estados "malos" de $C_i$, así como las transiciones "malas". Los estados malos se identifican de la siguiente manera: son aquellos estados $(x, y) \in C_i$ tal que las transiciones no controlables en el estado $x$ de la planta $G'$ no

pertenecen al conjunto de transiciones de $C_i$ en el estado $(x, y)$. De estos estados malos separar los "buenos", son aquellos estados $(x, y)$ que tienen como salida eventos forzables. Si no hay, ir al paso 4.

4. Obtener $C_i'$ eliminando los estados malos de $C_i$ y sus transiciones asociadas, eliminar las transiciones malas, hacer $C_{i+1} = trim(C_i')$

5. Si $C_{i+1} = C_i'$ parar, puesto que $supC(K) = L_m(C_i)$. Si no, incrementar $i$ y regresar al paso 3.

Los eventos forzables que no salen de estados malos se comportan como eventos controlables, donde la decisión se tomará por un agente externo. La interpretación física consiste en aplicar un evento forzable a la planta, anticipándose a la ocurrencia de un evento no controlable que lleve al sistema a un comportamiento no deseado.

### 4.1.3. Ejemplo de aplicación

Un ejemplo de la síntesis de un control supervisorio se muestra a continuación, adaptado del trabajo de Cembellín [17], sobre la simulación de sistemas de control híbrido. El ejemplo académico consiste en dos depósitos idénticos conectados por una tubería que se puede bloquear por medio de una válvula. Uno de los depósitos puede alimentarse por medio de una tubería de entrada, y ambos depósitos pueden vaciarse independientemente. Los actuadores son de tipo encendido/apagado. A diferentes alturas en ambos tanques se han colocado sensores de tipo encendido/apagado, que se activan cuando se alcanzan por el líquido. El espacio de estados para el sistema se muestra en la figura 4.2, donde los 25 estados se etiquetaron de acuerdo a los valores de las variables $x_1$ y $x_2$, las cuales corresponden al nivel del tanque 1 y 2 respectivamente. La expresión del cambio de nivel para ambos tanques viene dada por las siguientes ecuaciones diferenciales:

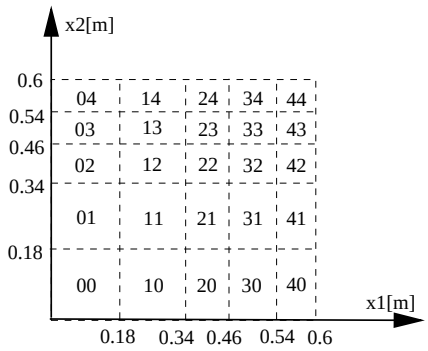


Figura 4.2: Partición del espacio de estados

$$\dot{x}_1(t) = u_1 q_p - u_2 c\sqrt{x_1(t)} - u_3 c sign(x_1(t) - x_2(t))\sqrt{|x_1(t) - x_2(t)|}$$

$$\dot{x}_2(t) = u_3 c sign(x_1(t) - x_2(t))\sqrt{|x_1(t) - x_2(t)|} - u_4 c\sqrt{x_2(t)}$$

Donde $q_p$ es el caudal máximo de entrada al tanque 1 y $c = \frac{s\sqrt{2g}}{A}$. $A$ es la superficie de cada depósito y $s$ es el área de cada sección de válvula. Con el fin de establecer control sobre el sistema, existen cuatro variables de entrada de tipo lógico (encendido/apagado) $u_1$, $u_2$, $u_3$ y $u_4$. Las cuales representan la bomba de entrada al depósito 1, la válvula para vaciar al depósito 1, la válvula que conecta a ambos depósitos, y la válvula para vaciar al depósito 2, respectivamente. Cuando estas entradas están encendidas su valor es 1, mientras que si están apagadas su valor es 0.

Cada región del espacio de estados está asociada a un *estado* del sistema, y una *transición* se asocia a un cambio entre estados. Como una primera aproximación a un DES, se puede suponer que solo están permitidas transiciones entre estados adyacentes. Aunque como se verá más adelante, hay transiciones entre estados adyacentes que son prohibidas. Por ejemplo, la transición $00 \rightarrow 01$ no puede tener lugar, ya que implicaría que se llenara el tanque 2 si el nivel del tanque 1 es igual o inferior al del

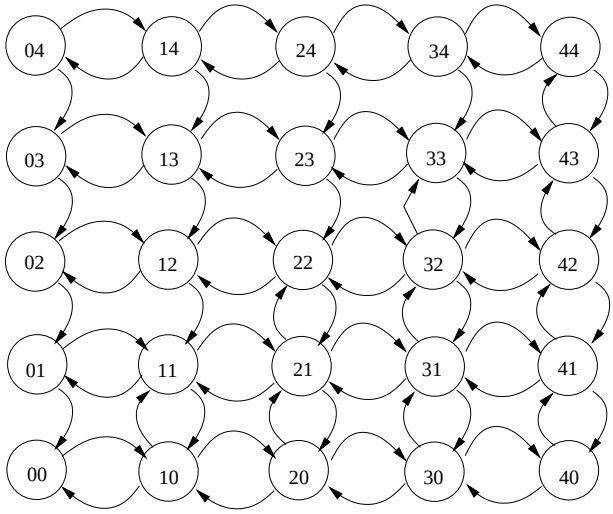


Figura 4.3: Modelo DES del sistema

tanque 2. Un análisis formal para determinar la imposibilidad de algunas transiciones lo establece Cembellín [17] en su trabajo. El modelo a eventos discretos representado en un autómata se muestra en la figura 4.3. Los estados del autómata corresponden a cada región de la partición de estados, y los arcos representan a las transiciones que son factibles de ocurrir.

El problema de control consiste en el siguiente: dado un estado inicial $X_1$ y otro final $X_2$ hay que encontrar la secuencia de estados intermedios que los conectan y las acciones de control que se deben aplicar para lograr la secuencia de transiciones que lleven a ese estado final. Si se desea ir del estado 11 al estado 13 existen varias trayectorias posibles, aunque un criterio para seleccionar la trayectoria consiste en evitar regiones que puedan llevar a comportamientos no deseados. Una posible trayectoria sería la que se muestra en la figura 4.4, la cual es un subconjunto del modelo DES y constituye un controlador DES, tal y como lo conciben Ramadge y Wonham. Las transiciones que exhibe este controlador DES se consideran como controlables, ya que existen unas entradas de control que permiten que ocurra cada una de las transiciones.

Las señales de control $U$ que permiten secuenciar la ruta mostrada en la figura es la siguiente, aclarando que la notación vectorial es de la forma $(u_1, u_2, u_3, u_4)$.

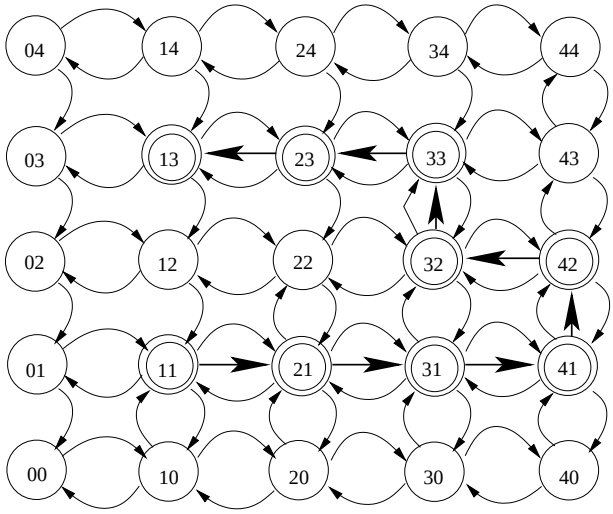


Figura 4.4: Controlador DES del sistema

$$11 \overset{1000}{\rightarrow} 21 \overset{1000}{\rightarrow} 31 \overset{1000}{\rightarrow} 41 \overset{1010}{\rightarrow} 42 \overset{0100}{\rightarrow} 32 \overset{1110}{\rightarrow} 33 \overset{0100}{\rightarrow} 23 \overset{0100}{\rightarrow} 13$$

## 4.2. Control Supervisorio basado en Redes de Petri

El diseño de mecanismos de supervisión utilizando redes de petri se basa en un enfoque distinto: el procedimiento más conocido es el de Moody y Antsaklis [76], que introduce restricciones lineales para las marcaciones de la red, introduciendo lugares adicionados denominados "de holgura" cuya función es la de prevenir que la red de Petri exhiba marcaciones no deseadas. Existen otras definiciones de redes de Petri que permiten controlar marcaciones, como la inclusión de arcos inhibidores, redes de Petri continuas, entre otras. El procedimiento clásico establece que la restricción general es de la forma $l^T \mu_p \leq b$, la cual se puede llevar a $l^T \mu_p + \mu_c = b$, donde $\mu_p$ es la marcación de la red de Petri que representa el proceso a controlar, y $\mu_c$ introduce la marcación que ejecuta el control sobre el proceso. El resultado neto de introducir esta restricción es la adición de uno o mas lugares a la Red de Petri que modela a la planta, para mantenerla controlada. La estructura del supervisor se diseña utilizando el concepto

de *invariantes de lugar* de la Red de Petri, equivalente a los *supervisores basados en monitores*. Estos conceptos se enuncian a continuación, para mayor claridad:

**Definición 4.9. Invariantes de lugar** Corresponde a un conjunto de lugares cuya cantidad de marcas (o tokens) permanece constante para todas las marcaciones posibles. Si esos lugares se agrupan en un vector $x$, se debe cumplir que $x^T\mu = x^T\mu_0$ donde $\mu_0$ es la marcación inicial y $\mu$ es una marcación subsiguiente.

**Definición 4.10. Monitor** Es una Red de Petri que restringe mínimamente la conducta de la planta, es decir, prohibe disparos de transición que lleve a marcaciones prohibidas.

### 4.2.1. Procedimiento de síntesis de Moody y Antsaklis

El procedimiento para llevar a cabo la implementación de las restricciones lineales opera de la siguiente manera: la planta o sistema a controlar se modela por una Red de Petri cuya matriz de incidencia es $Dp_{n\times m}$, el controlador a sintetizar tiene una matriz de incidencia $Dc$, la cual se agrega a $Dp_{n\times m}$ para obtener la matriz de incidencia de la planta controlada $D$. $D_c$ se halla a partir de la igualdad $LD_p + D_c = 0$, donde $L$ es el vector fila que especifica la restricción del diseño, o sea los lugares que deben cumplir la restricción lineal. De esta manera, el controlador se calcula de la forma $D_c = -LD_p$. La marcación inicial de los estados de holgura $\mu_{0c}$ se obtienen despejando de la ecuación $L\mu_{0p} + \mu_{0c} = b$. Una vez que se obtiene la marcación inicial y la fila adicional de la matriz de incidencia, se agregan los lugares correspondientes, teniendo en cuenta que los términos positivos en la matriz de incidencia indican entradas al lugar del controlador, y los términos negativos son las salidas del controlador a las transiciones correspondientes.

No basta con aplicar los criterios de invarianza de lugar para sintetizar un supervisor

basado en redes de Petri. Es necesario aplicar el *criterio de admisibilidad* para las restricciones, cuyo tratamiento matemático se puede consultar en la obra de Moody y Antsaklis [76]. Para ello hay que tener en cuenta el tipo de transiciones con las que interactúa el supervisor, ya que algunas de ellas no las puede observar, se dice entonces que son *transiciones no observables*, y también hay transiciones que el supervisor no puede inhibir, las cuales se denominan *transiciones no controlables*. Para decidir si el supervisor es admisible hay que verificar las siguientes condiciones: $LD_{uo} = 0$ y $LD_{uc} \leq 0$, donde $D_{uo}$ y $D_{uc}$ son las matrices de incidencia de la planta cuando se eliminan las columnas que corresponden a las transiciones no observables y no controlables, respectivamente.

En caso de que las restricciones no cumplan con el criterio de admisibilidad, se puede buscar otro conjunto de restricciones que cumplan con este criterio, y obtener así un controlador que satisfaga a ambas. Moody y Antsaklis establecen una proposición para determinar si una restricción para una marcación de una red de Petri es admisible solamente si i) Las condiciones iniciales de la planta satisfacen la restricción, y ii) existe un controlador máximalmente permisivo (construido bajo el supuesto de transiciones controlables) que satisfaga la restricción y no inhiba cualquier transición no controlable que de otra manera esté habilitada, es decir, evitar que de un controlador salgan arcos hacia transiciones no controlables. Por lo tanto, una restricción no admisible puede llevarse a una admisible, por medio de manipulación algebráica, generalmente agregando un término adicional a la restricción. Un ejemplo de aplicación del proceso de síntesis de redes de Petri se puede encontrar en el *anexo A*.

### 4.2.2. Otros procedimientos

Los métodos para sintetizar sistemas supervisores es un área sobre la que aún se están realizando investigaciones, de manera que no existe un método definitivo para generar sistemas supervisores. De hecho, en sistemas con dinámica continua es difícil

obtener un supervisor utilizando el procedimiento estándar, debido a la dificultad en proyectar de variables continuas a discretas, o las transiciones que se van a inhibir son no controlables. Por tal motivo, una estrategia adecuada puede ser el uso de *arcos inhibidores*, que también previene marcaciones no deseadas, siempre y cuando se mantenga una marcación en un lugar específico. Este mecanismo funciona de la siguiente manera: cuando un lugar está conectado a una transición por medio de un arco inhibidor, esta no se puede disparar mientras el lugar disponga de marcas. Por otro lado, si el lugar no posee marcación alguna en un momento dado, entonces estará habilitando a la transición que corresponda.

La red de Petri de la figura 4.5 muestra un sistema supervisado, donde un lugar de la planta impide que ocurra una transición mientras que tenga una marca en él. Así, el lugar InAB ejerce inhibición sobre la transición mid, al igual que el lugar SupMes sobre la transición ensamblaABC. Este esquema de supervisión fue propuesto por Parra y Chacón [89] para un esquema de producción compuesto por dos robots compartiendo un recurso común: una mesa giratoria. La misión es que el supervisor impida que ambos robots utilicen la mesa simultáneamente y luego la hagan girar, o que un robot empiece a ensamblar una pieza si la mesa no está en la posición adecuada.

Es necesario aclarar que sintetizar supervisores con arcos inhibitorios requiere un conocimiento específico sobre la red de Petri a controlar, con el fin de agregar los lugares y transiciones adecuados que impidan la marcación incorrecta. Consultas sobre el potencial descriptivo de los arcos inhibidores se pueden hacer en el trabajo de Silva [102].

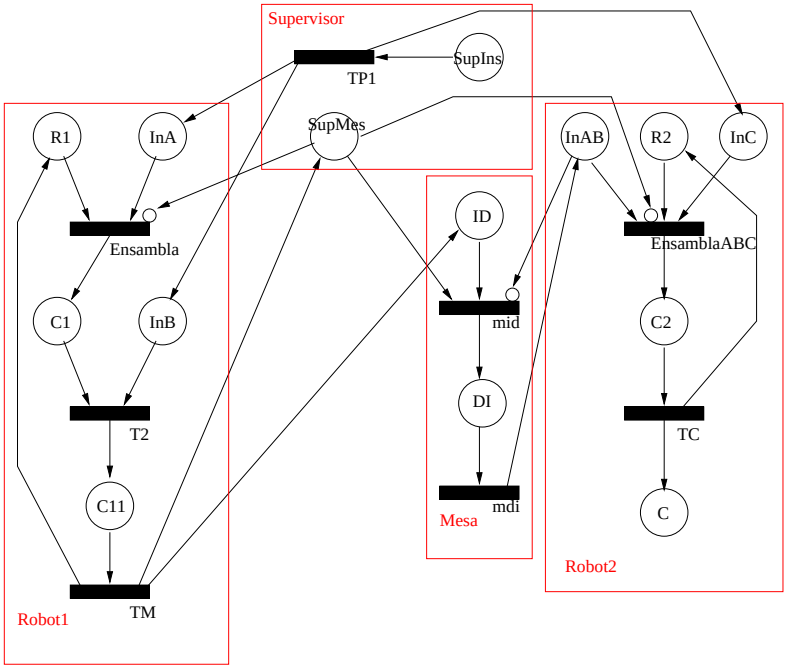


Figura 4.5: Red de Petri que presenta arcos inhibidores

## 4.3. Sistemas de Control Supervisorio como un control orientado a procesos y recursos

Para supervisar una unidad de producción es necesario llevar a cabo dos pasos: el primero de ellos consiste en describir la dinámica de la conducta de esa unidad de producción por medio de un DES, como se explicó en la sección anterior, donde cada estado corresponde a una condición de operación del sistema. El segundo tiene como punto de partida las especificaciones de estados o marcaciones deseadas de esa conducta, y de ahí sintetizar un supervisor para mantener esa conducta dentro de una trayectoria de estados determinada, o prevenir que el sistema exhiba una conducta no deseada. En esta sección se han mencionado los enfoques para describir la conducta supervisada de un sistema, y ahora se está llevando al caso particular de supervisar un HMS bajo la arquitectura de referencia PROSA.

A partir de las teorías y métodos sobre cómo sintetizar el Control Supervisorio se puede establecer un esquema de cómo obtener el Supervisor de una unidad de pro-

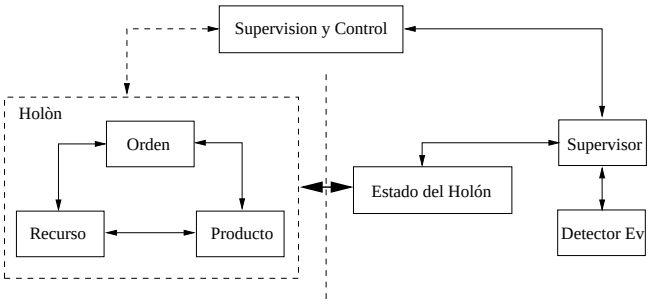


Figura 4.6: Esquema de Supervisión de un Sistema Holónico

ducción PROSA: a partir de los tres holones básicos que conforman la unidad de producción se debe supervisar el cumplimiento de la *Orden de producción*, y para ello es necesario determinar el *estado actual* de la misma, el cual es un estado compuesto por las dinámicas de los recursos (cada uno de los holones Recurso) y del proceso (la relación entre la Orden a ejecutar y los Recursos). La *trayectoria* de estados se establece de antemano, de acuerdo a la meta de producción y a la planificación que se haga para ejecutar una orden determinada, de acuerdo a un método de producción. Esta trayectoria establece cuales eventos deben ocurrir, y en qué secuencia. La misión del supervisor es asegurarse que los eventos dentro de una unidad de producción ocurran en el orden deseado.

En la figura 4.6 se puede apreciar este esquema de supervisión del sistema holónico, donde el estado de la unidad de producción que proyectan estos holones lo monitorea un detector de eventos, a partir de la medición de las variables del proceso físico, y la determinación de la configuración actual de la planta. Este detector de eventos reacciona ante la ocurrencia de un evento determinado notificando al sistema supervisor, el cual a su vez interactúa con la *clase Supervisión y Control*, que contiene un DES donde se planifican las trayectorias de la conducta exhibida por el sistema y genera de esta forma la configuración que debe tener la unidad de producción para incidir sobre el proceso físico, y así mantener los valores de las variables dentro de las condiciones nominales de operación.

### 4.3.1. Síntesis del Supervisor bajo el enfoque de Máquinas de Estado Finito

Este proceso tiene como punto de partida la descripción de la unidad de producción como un autómata que resulta del producto síncrono de los autómatas asociados a las *etapas del proceso y los recursos involucrados*. Para sintetizar un Supervisor se deben hacer además las siguientes actividades:

1. Particionar regiones de operación, delimitándolas con planos, cada región la asocia a un estado.

2. Definir las transiciones posibles que puedan existir entre estos estados, a partir de las características físicas del modelo híbrido que tenga definido una hipersuperficie que contenga a ambas superficies delimitada por un plano.

3. Establecer los estados deseados y no deseados del autómata compuesto.

4. Hallar los lenguajes $L(G)$ y $L_m(G)$ como especificación inicial de conducta.

5. Aplicar el algoritmo de Torrico y Cury para hallar el máximo lenguaje controlable $supC(K)$ que cumpla las especificaciones de conducta.

6. Determinar el estado inicial en que se ubicaría la unidad de producción, para hacer que el sistema evolucione.

### 4.3.2. Síntesis del Supervisor bajo el enfoque de redes de Petri

Este procedimiento también tiene como punto de partida la descripción de la unidad de producción por medio de redes de Petri. Cada lugar de la red de Petri en la sección de lugares del proceso representa una región de operación diferente, y los transiciones representan los cambios de regiones que están permitidas. En las secciones de recurso y método los lugares denotan el estado discreto en que se encuentran, mientras que

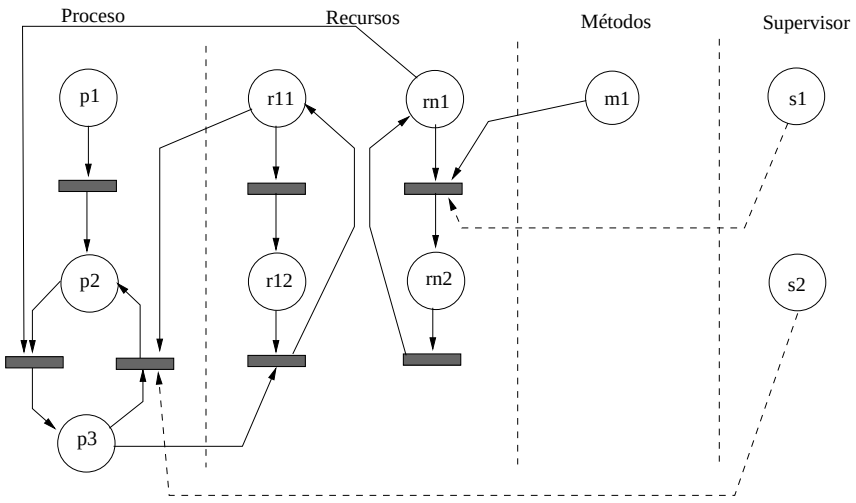


Figura 4.7: Esquema general de supervisión de una unidad de producción

las transiciones se asocian a cambios en la configuración. Una vez armada la red de Petri que representa la dinámica a lazo abierto de la Unidad de Producción se definen las restricciones lineales que van a indicar las marcaciones incorrectas dentro de la red de Petri. Una vez definidas, se aplican el procedimiento de síntesis por medio del producto matricial $LD = a$, y se aplica el criterio de admisibilidad. En caso de no cumplir, se deben hacer combinaciones lineales sobre las restricciones para replantearlas, y así aprobar el criterio de admisibilidad una vez se vuelva a realizar el producto matricial.

Una vez obtenida la red de Petri con los lugares que corresponden al supervisor, será similar a la que aparece en la figura 4.7, donde además de los lugares que corresponden a las etapas del proceso, los estados de los recursos y los métodos de producción están los lugares que corresponden a las acciones del supervisor, con sus correspondentes transiciones para implementar las restricciones antes mencionadas. Es conveniente aclarar que este esquema no es una plantilla general, sino que es particular a cada unidad de producción. Una vez obtenida esta red se establece su marcación inicial, y se deja evolucionar de acuerdo a los eventos que vayan ocurriendo.

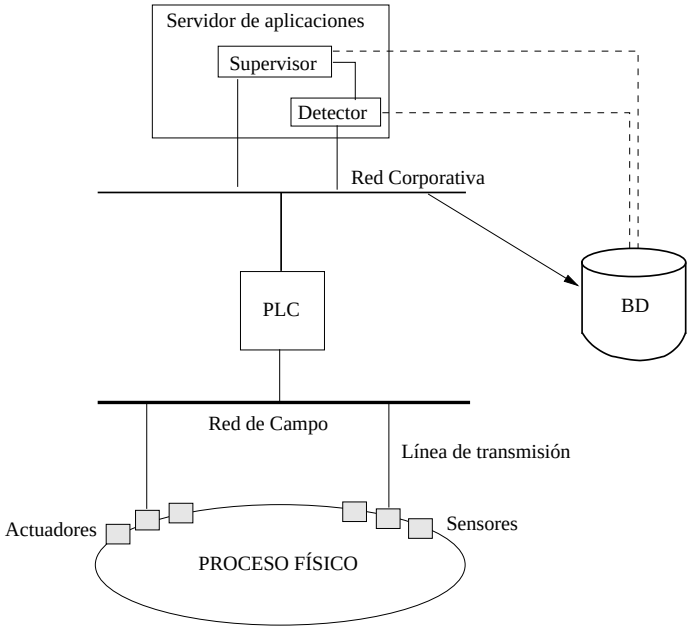


Figura 4.8: Esquema de implementación del mecanismo de supervisión

## 4.4. Control Supervisorio en una arquitectura informática y de comunicaciones

La implementación de sistemas de control supervisorio es un aspecto diferente al de la síntesis de mecanismos de supervisión, pero no implica que sea fácil su implementación una vez se ha sintetizado el supervisor. Aspectos a tener en cuenta son las limitaciones del hardware y el lenguaje de programación con que se va a implementar. La interacción entre la planta y el supervisor físico no siempre va a corresponder con la interacción entre el modelo de la planta y de su supervisor. Una implementación real debe ser mucho mas detallada que el modelo teórico, y esta implementación debe basarse en una arquitectura de referencia del sistema de producción.

En la figura 4.8 se muestra un esquema de implementación para un sistema supervisor, utilizando tecnologías de computación y comunicaciones actuales. El proceso físico de transformación está conectado a un conjunto de sensores y de controladores, los sensores elaboran mediciones de las variables asociadas al proceso de transforma-

ción, y los actuadores son controladores que cambian la configuración de los recursos de la planta con el fin de introducir transformaciones en el proceso. Tanto sensores como actuadores están conectados a una red de campo, la cual también está conectada a un controlador lógico programable (PLC) o algún otro dispositivo, como un sistema de control distribuido (DCS). El PLC o cualquier otro dispositivo también está conectado a una red de datos corporativa, y dependiendo de sus características, puede albergar dentro de su programación la represetentación matricial de un DES, o este DES puede estar implementado por objetos que están alojados en un servidor de aplicaciones, al mismo nivel que las aplicaciones de planificación y gestión corporativa. En el caso de PROSA, los objetos que reflejan lo que ocurre dentro de la organización estarían dentro de estos servidores de aplicaciones y se comunicarían con los objetos que llevan a cabo la supervisión por medio de algún protocolo establecido. Es importante registrar las lecturas de los sensores al PLC en una base de datos, ya que esto proporciona la posibilidad de trazabilidad de los datos tomados del proceso fisico, bien sea para detectar eventos aplicando ventanas, o para medir el desempeño de la unidad de producción.

Otro ejemplo de arquitectura de referencia para implementar sistemas supervisores es el modelo CHAMP, propuesto por Hellgren en su tesis doctoral [52]. Básicamente consiste en un control de alto nivel y un control de bajo nivel, interconectados a una base de datos. El control de alto nivel contiene al Supervisor y el Planificador (Scheduler), mientras que el control de bajo nivel contiene el hardware que controla los dispositivos de la planta. El mecanismo de supervisión del control de alto nivel está basado en redes de Petri. La base de datos guarda las trazas de la configuración de la planta, así como la traza de los valores medidos. De hecho, la trazabilidad es un concepto que los modelos de síntesis de supervisores no tienen en cuenta, pero que es necesario para la implementación en un entorno de producción real. La implementación de esta arquitectura proyecta los DES en un Controlador Lógico Programable, utilizando el

lenguaje estándar IEC 61131-3, en particular los diagramas de escalera. Vale la pena mencionar otro ejemplo de implementación de sistemas supervisores es el de Torrico [105], basado en máquinas de estado finito, una aplicación que implementa esta máquina de estado finito denominada CONDES, y tiene una arquitectura física para sistemas reales que consta de los módulos Supervisor, planta física, interface y módulo intermediario.

Algunos de estos elementos de implementación se usarán en la sección de modelado conceptual en UML, para establecer un posible panorama de implantación sobre la arquitectura holónica basada en el modelo PROSA, y serán considerados estos aspectos para la validación de la conducta del sistema holónico mediante simulación.

## 4.5. Conclusión

El uso de mecanismos de supervisión basados en sistemas a eventos discretos proporciona autonomía a los sistemas holónicos, puesto que les permite controlar su conducta en función de las metas de producción. Es necesario encontrar entonces la manera de llevar estas metas de producción hacia una representación que utilice sistemas a eventos discretos, bien como autómatas o bien como redes de Petri. Los *modelos multiresolucionales* que se expondrán en el siguiente capítulo es una alternativa viable. Una vez representado el modelo de conducta deseada queda por aplicar alguno de los métodos de síntesis de sistemas supervisores descritos anteriormente, para obtener el mecanismo de supervisión adecuado.

Para simular este comportamiento, se debe basar el modelo de la planta y en la síntesis del supervisor, basado en sistemas a eventos discretos, y las reglas que permiten decidir si ocurrió un cambio de estado. En la implementación de sistemas supervisores en un entorno real, aspectos como la interacción entre la física de la planta y el hardware del supervisor no se puede explicar tan fácilmente como las interacciones entre sus

respectivos modelos, de manera que un mecanismo de registro de las variables que se están controlando debe tenerse en cuenta para la simulación, puesto que proporciona trazabilidad al comportamiento del modelo. Una definición de trazabilidad es proporcionada por la Asociación Española de Codificación Comercial (AECOC), que la define como aquellos "Procedimientos que permiten controlar el histórico, la situación física y la trayectoria de un producto o lote de productos a lo largo de la cadena de suministro en un momento dado, a través de unas herramientas determinadas".

# Capítulo 5

# Gestión de los Holones y del mecanismo de supervisión

En la sección anterior se han establecido los mecanismos para sintetizar un sistema supervisor, cuya función es la de controlar la conducta de una Unidad de Producción concebida bajo el paradigma holónico. Este sistema supervisor permite la programación de actividades destinadas a mantener unas trayectorias en las variables asociadas a un proceso industrial, mientras se cumplen restricciones de diseño sobre los recursos utilizados. La Unidad de Producción como un Sistema Holónico bajo la arquitectura PROSA no solo debe cumplir con la misión de producción, sino que también debe estar en condiciones de *comunicarse y negociar* con otras Unidades de Producción u Holones, ubicados dentro de la misma planta, o incluso con clientes externos (que pueden ser otras plantas de producción). Dentro del diseño de esta Unidad de Producción, se requiere de alguna entidad que gestione cada uno de sus Holones, reflejando la parte física y de razonamiento utilizando alguna clase de software. La entidad más adecuada para llevar a cabo esta acción consiste en los denominados *Agentes*, entre cuyas características se tiene la cooperación, adaptabilidad y distribución, lo que los hace adecuados para desempeñarse como un puente entre el piso de

planta y las aplicaciones corporativas, así como permitir la cooperación entre diferentes unidades de producción.

Este capítulo inicia con unas definiciones sobre agentes y la teoría de influencias y reacciones. Luego se profundiza sobre lógica de agentes y la especificación de la conducta de un agente utilizando lógica de primer orden, para pasar posteriormente a aspectos relacionados con tecnología de agentes como el diseño, metodologías y plataforma de implementación. Finalmente se presenta un panorama de implantación de los sistemas holónicos supervisados bajo esta tecnología y la generación de reglas de conducta de los supervisores y gestores de holones.

**Definición 5.1. Agente** La definición más general de agente es la de Russel y Norvig, que establecen que un agentes es: "Toda entidad que observa su ambiente, razona sobre lo observado y actúa en función de unas metas" [97]. Nick Jennings [55], define al agente como un sistema computacional autónomo y flexible, que es capaz de actuar en un entorno determinado. Flexible significa, que el agente es:

- Reactivo, reacciona al entorno en el cual se encuentra.
- Pro-activo, es capaz de cumplir su propia agenda (planes u objetivos).
- Social, es capaz de comunicarse con otros agentes a través de algún lenguaje.

Otros investigadores atribuyen a los agentes otras propiedades tales como autonomía, habilidad social, racionalidad, reactividad, pro-actividad, adaptabilidad, movilidad, veracidad y Benevolencia. Varios de ellos han establecido definiciones del término *Sistema Multiagente*, aunque la más interesante es la de Ferber y Müller (F-M) [37], quienes dentro de su teoría de *influencias y reacciones*, introducen la definición del entorno de los agentes como un sistema dinámico, similar a la definición 3.1 en la sección tres de este documento, lo que permite establecer que un sistema industrial se puede modelar como un sistema multi-agente.

## 5.1. Teoría de Influencias y Reacciones

La propuesta inicial de Ferber y Müller describe a un Sistema Dinámico por medio de un *estado global extendido*, el cual se compone de las variables del ambiente que conforman al *estado del sistema*, y de las *influencias* de los agentes. Esta definición se establece a continuación:

**Definición 5.2. Influencias.** Influencias son los intentos que hacen los agentes para modificar el curso de eventos que ocurrirán, y que podrían tener otro curso si no fuera por estas influencias. Estas influencias son generadas por los propios agentes.

Formalmente, un sistema multi-agente consta de cuatro conjuntos:

- $\Sigma$ es el conjunto de todos los estados posibles del ambiente que rodea a los agentes.
- $\Gamma$ es el conjunto de las influencias posibles de los agentes
- $\Lambda$ son las leyes que rigen al sistema.
- $Op$ es un conjunto de operadores de los agentes.

La dinámica global del sistema podría expresarse como una séxtupla con la siguiente estructura:

$$D_g =< \Sigma, \Gamma, Op, \Lambda, Accion, Reaccion > \tag{5.1}$$

Los intentos del agente por cambiar su entorno, la reacción de este entorno ante estas influencias y la evolución del sistema se definen por medio de las siguientes expresiones:

$$\gamma' = accion(op, \sigma', \gamma) \tag{5.2}$$

$$\sigma' = reaccion(\lambda, \sigma, \gamma) \tag{5.3}$$

$$evolucion(\sigma, \gamma) = evolucion(paso(\sigma, \gamma)) \tag{5.4}$$

$$\tag{5.5}$$

Donde $\sigma \in \Sigma$, $\lambda \in \Lambda$, $\gamma \in \Gamma$, y $op \in OP$. La función $paso(\sigma, \gamma)$ se aplica a las influencias y a la dinámica del ambiente, para producir cambios en el mismo. Esta función depende del tipo de agentes que están interactuando en el sistema, los cuales exhiben unas conductas diferentes.

Según Bussmann *et al* [11] la tecnología de agentes es una de las más adecuadas para administrar un conjunto de holones supervisados, debido a la naturaleza distribuida de la planta a automatizar y las capacidades de los agentes para cooperar en pos de una meta global. Otra razón importante para considerar el uso de tecnología de agentes en el control supervisorio de procesos es que algunas arquitecturas hacen uso explícito que hacen de la variable tiempo, lo que puede potenciar la a representación de la dinámica del proceso usando DES, como Autómatas y Redes de Petri, por ejemplo. Este uso explícito de tiempo permite a) programar actividades a realizar de los agentes cuando quieran intervenir en el curso de los eventos del sistema, b) validar su diseño utilizando un simulador a eventos discretos e incorporar agentes que observen la dinámica discreta del proceso y actúen sobre su entorno. En la siguiente sección se tratará sobre cómo resolver el problema de especificar el comportamiento de un agente, mientras que los aspectos relacionados con metodología de diseño e implementación se tratarán en las secciones subsiguientes.

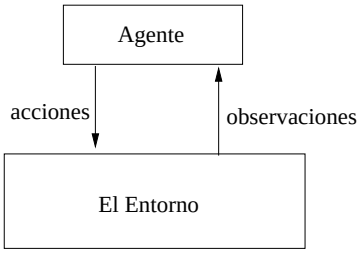


Figura 5.1: Interacción de un agente con su entorno

## 5.2. Especificación del comportamiento de un Agente

La manera más natural para especificar la conducta de un agente determinado consiste en utilizar Lógica de Primer Orden (LPO, de aquí en adelante). El enfoque basado en lógica lo propuso Robert Kowalski [64, 65], donde establece que un agente opera en forma de *ciclos*, tal que iterativamente ejecuta las actividades de observar su entorno, razonar sobre las observaciones, decidir la ejecución de un conjunto de actividades, y actuar sobre su entorno, volviendo a repetir el ciclo. El componente principal del razonamiento de este agente consiste en la LPO. La propuesta de Kowalski no solo establece que un agente tiene capacidad de razonar, sino también la capacidad de reaccionar ante su entorno, por lo tanto combina *racionalidad* con *reactividad*. Un esquema donde se muestra una primera aproximación entre la relación que establece un agente con su entorno se aprecia en la figura 5.1. Entre los motivos para especificar la conducta del agente utilizando LPO se tienen 1) Sus características lo hacen adecuado para implementar una unidad de producción con reactividad, y por medio de la racionalidad se pueden implementar reglas que le permitan actuar sobre el proceso productivo, y la comunicación con otros agentes. 2) Las especificaciones de los modos de operación de un proceso industrial se basan en comunicación textual, la cual puede llevarse directamente a *reglas lógicas de condición-acción, y procedimientos de reducción de metas*, las cuales determinan el razonamiento del agente. En lo que queda de esta sección se profundizará sobre este tipo de reglas, y sobre la arquitectura de razonamiento de nuestro agente.

La mayor parte de nuestra comunicación textual se basa en las oraciones, las cuales pueden verse desde el punto de vista computacional y lógico. Tanto el lenguaje natural como el lenguaje de la lógica representan oraciones, y manejan condiciones, aunque el lenguaje natural maneja también información del contexto. Las oraciones pueden tener varias formas: como un procedimiento de reducción de metas, una implicación, una regla condición-acción o una restricción. Un procedimiento de reducción metas modela al razonamiento proactivo, y tiene la forma lógica de: *metas si se cumplen sub-metas*, la cual se le aplica programación lógica para llevarla a la forma: para concluir meta resolver sub-metas. Se representa de la forma general *para...haga*, y utiliza razonamiento hacia atrás. Las implicaciones, las restricciones y las reglas de condición-acción utilizan razonamiento hacia adelante, el cual se puede escribir imperativamente o declarativamente. El razonamiento hacia adelante tiene la forma *si condiciones entonces conclusión*. Es importante hacer la distinción de los tipos de oración, ya que una oración declarativa puede ser cierta o falsa, mientras una oración imperativa tiene la forma de regla de condición-acción, la cual se puede obedecer o desobedecer.

**Definición 5.3. Pensamiento proactivo** La lógica modela el pensamiento proactivo, el cual ejecuta un procedimiento de reducción de metas a sub-metas y/o acciones. El pensamiento proactivo tiene *metas y creencias*, expresadas en forma de reglas del tipo: *conclusión si condiciones*. Los hechos también son un tipo especial de creencias, los cuales registran observaciones y tienen la forma de: *conclusión si nada*. El procedimiento de reducción de metas es representado por un grafo de reducción de metas, y por el razonamiento hacia atrás con implicación, el cual relaciona la conclusión con la meta, y las condiciones con sub-metas.

**Definición 5.4. Pensamiento reactivo** El pensamiento reactivo realiza observaciones sobre el entorno y dispara acciones o metas. Las reglas de condición-acción

modelan al pensamiento reactivo, generan conductas y varias de ellas conforman un Sistema de producción. Estas reglas se escriben de forma imperativa. El pensamiento reactivo usa el razonamiento hacia adelante, caracterizado por la implicación lógica.

El agente interactúa con su ambiente, procurando hacer ciertas sus metas y teniendo a la Lógica como el componente principal de su razonamiento. La lógica por si sola no distingue metas de creencias, de manera que la distinción la debe hacer el agente dentro de sus reglas internas. Las metas pueden ser de varios tipos: restricciones para prevenir estados indeseables, acciones a realizar, metas de mantenimiento para preservar relaciones entre el agente y el mundo, y metas por lograr para alcanzar a un estado deseado. Las reglas de tipo condición-acción se entienden como metas de mantenimiento, y las observaciones disparan este tipo de metas. Las creencias incluyen definiciones y taxonomía, utilizadas por los agentes para clasificar, organizar y experimentar. Además incluyen oraciones atómicas, que registran observaciones y leyes de la naturaleza. El proceso general que aplica el agente para sus observaciones y acciones es el siguiente: paso 1: razona hacia adelante, apareando observaciones con condiciones de sus reglas de mantenimiento. El paso 2 consiste en que razona hacia adelante o hacia atrás, verificando las otras condiciones que no se observaron. El paso 3 consiste en derivar la meta, para hacer cierta sus conclusiones, resultando en una meta de alcance para obtener el estado final. Este proceso se puede aplicar iterativamente para derivar mas sub-metas en el futuro. La figura 5.2 nos muestra el esquema de este proceso de razonamiento.

## 5.3. Tecnología de agentes

Partiendo de las definiciones de agente anteriormente mencionadas, donde se obtiene una especificación de su comportamiento, se pasa a su implementación dentro de una arquitectura computacional, utilizando Tecnologías de Información y Comu-

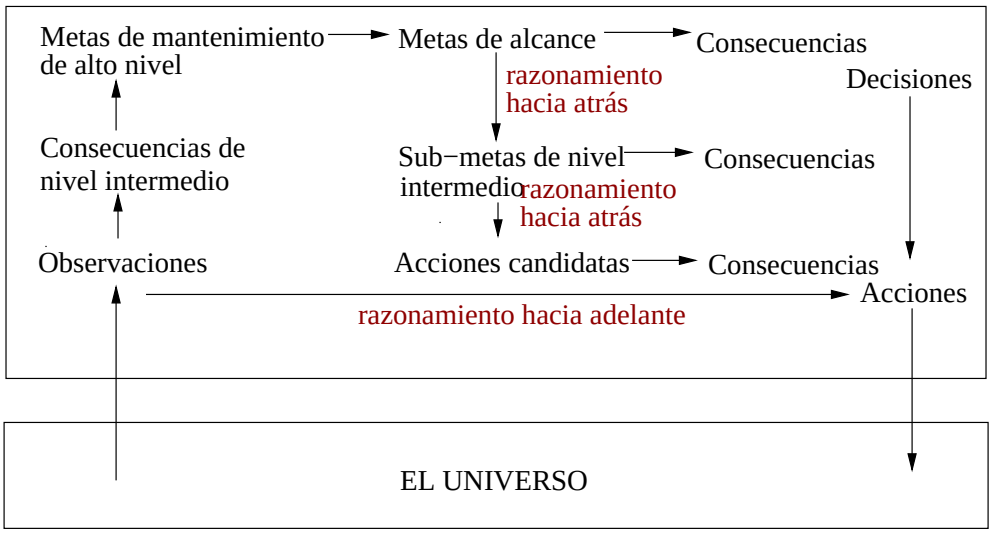


Figura 5.2: Proceso de razonamiento de un agente. Tomado de Kowalski: "como ser artificialmente inteligente" (2005)

nicación (TICs). Varios aspectos se van a cubrir, como lo son las plataformas de implementación, métodos para hallar roles de agentes, protocolos de comunicación, ontologías disponibles para sistemas de producción, y mecanismos de negociación.

### 5.3.1. Metodologías para diseñar sistemas multi-agente

Existen muchas metodologías de diseño de Sistemas Multi-Agente (SMA, de aquí en adelante). Giret [45] propone clasificarlas en dos grandes grupos: aquellos métodos de propósito general para diseñar SMA, y los métodos orientados específicamente hacia los Sistemas de Manufactura. Entre los métodos de propósito general, se subdividen a su vez en la siguiente clasificación:

1. Métodos orientados a las organizaciones. Se basan en un modelo de organización social para asignar roles a los agentes, y establecen algunas reglas de interacción entre ellos. Entre estos métodos están GAIA, propuesto por Wooldridge, Jennings y Kinny [109], en el cual establecen que la construcción de un SMA es un proceso organizacional, basado en una colección de roles (responsabilidad de cada agente, recursos a utilizar, tareas e interacciones). Genera tres modelos:

agentes, instancias y servicios asociados a cada rol. GAIA V.2, propuesta por Zambonelli [112], extiende al método GAIA original, estableciendo que el entorno del SMA constituye una abstracción de análisis y diseño, incorporando reglas y arquitecturas organizacionales. Otra metodología consiste en Agent Societies, propuesta por Dellarocas, Klein, y extendida por Dignum [29], donde describe: (i) estructura y las características globales de un dominio a partir de un modelo de organización; (ii) define la población de agentes mediante contratos sociales, que regulan la ejecución de roles por parte de agentes individuales; y (iii) la interacción entre los agentes mediante contratos de interacción. Esta propuesta se llevó a cabo dentro de un marco de comercio electrónico, que luego se estandarizó a otros contextos.

2. Otro grupo consiste en aquellas metodologías que extienden a los métodos orientados a objetos, haciendo las adaptaciones necesarias para trabajar con agentes. Entre estas metodologías está una extensión de BDI, propuesta por Kinny y Georgeff [61] la cual extiende los métodos orientados a objetos, con dos niveles de abstracción: un nivel externo que corresponde a un diagrama de clases, y un nivel interno, que contiene los modelos de creencias, objetivos y planes. Identifica roles clave en la aplicación y sus interrelaciones. Otra metodología para desarrollar software basado en agentes es Tropos, propuesta por Mylopoulos *et al* [81], la cual utiliza extensiones del Lenguaje Unificado de Modelado (UML). La aplicación del método inicia con la captura de requisitos, modelando los objetivos, actividades, recursos y actores del sistema, y ampliando el diagrama de actores con la ayuda de patrones de análisis de software, que son notaciones gráficas que denotan relaciones entre interfaces de objetos. Luego en la etapa de diseño, se le asignan a los agentes las capacidades necesarias para satisfacer los objetivos misionales del sistema.

3. En otra clasificación se encuentran aquellas Metodologías orientadas a meta-modelos (una descripción abstracta de los elementos que tiene un modelo), las cuales incorporan sus propias herramientas de desarrollo. Se destaca MESSAGE (Methodology for Engineering Systems of Software Agents) [35], la cual combina técnicas de ingeniería del software con análisis y diseño de sistemas multi-agente. Básicamente es un método orientado a agentes con notación UML, y el proceso de modelado es el de RUP, propuesto por Rational. INGENIAS, propuesta por Gómez y Pavón [46] es una metodología que profundiza a MESSAGE, en cuanto a especificación y en el proceso de desarrollo, y agrega nuevas herramientas de soporte. Define un conjunto de actividades cuya ejecución termina en un conjunto de modelos, instanciado a partir de un meta-modelo de agentes.

4. Métodos orientados a la interacción. Miles *et al* [73] proponen un método de diseño denominado *Análisis de Interacción entre Agentes*, la cual establece un análisis de interacción entre agentes, refinando sucesivamente los objetivos de un sistema. Cassiopeia es un método propuesto por Collinot *et al* [25], centrado en determinar el comportamiento individual de cada agente a partir de una especificación de tarea colectiva de la organización. Distingue tres niveles de comportamiento: (i) elemental, (ii) relacional, y (iii) organizacional. Los comportamientos organizacionales son aquellos que permiten a los agentes gestionar la formación o disolución de los grupos, organizando así el SMA para obtener comportamiento grupal.

Otro gran grupo de metodologías de desarrollo de sistemas multi-agente está conformado por aquellas aplicaciones específicas a orientadas al modelado de procesos de manufactura u organizaciones industriales, que se han modificado para acoger a los agentes como componente activo en el proceso de producción. Estas metodologías se resumen a continuación:

- Una metodología para desarrollar sistemas de agentes basados en métodos de modelado de sistemas de manufactura como IDEF combinado con métodos orientados a objetos como OMT es propuesta por Kendall *et al* [60], donde el punto de partida es un modelo IDEF0 del sistema a modelar, y un modelo orientado a objetos. Posteriormente se identifican los agentes y las interacciones entre estos modelos. Se asocian a un actor en el modelo orientado a objetos, y en IDEF se asocian sus interacciones con el intercambio de información de control entre unidades funcionales.

- Otra metodología es la de Ritter *et al* [94], la cual propone un método para identificar agentes en un sistema de manufactura, basado en las dependencias físicas y lógicas entre los objetos de manufactura, y un proceso de agregación aplicado a las entidades que intervienen en el proceso de manufactura.

- Una metodología para el modelado de sistemas de control de producción basado en agentes, es la que proponen Colombo *et al* [26], utilizando Redes de Petri. Los procesos de manufactura descritos por medios de redes de Petri sirven para identificar las acciones de control que van a ejecutar los agentes, aunque estos no se representen explícitamente en la red de Petri. Por esta razón es difícil asignarles acciones de control cuando existen varios agentes.

- Una metodología mas completa y elaborada es DACS (Design of Agent Control Systems), propuesta por Bussman *et al* [11]. El punto de partida es el problema de control de producción, que incluye una especificación de los elementos a controlar y un modelo de su comportamiento físico. Luego se aplican las siguientes fases: i) análisis de toma de decisiones, donde se observan que decisiones son necesarias para que funcione el proceso de producción, las cualses se toman sobre los objetos del sistema; y se identifican las dependencias de decisiones que puedan existir sobre las actividades de producción. ii) Luego se procede a la

identificación de agentes, donde se agrupan las decisiones para asignárselas a un agente determinado, aplicando algunas reglas que son guía de este proceso. iii) Finalmente, se aplica la tercera fase que es la de selección de protocolos de interacción. La aplicación de estas tres fases proporciona un diseño basado en agentes para resolver un problema de control de un proceso de producción. Una metodología basada en agentes y orientada específicamente para sistemas de producción holónicos es ANEMONA, propuesta por A. Giret [45], donde define un agente abstracto, el cual proporciona al sistema de producción holónico la característica de recursividad.

### 5.3.2. Comunicación entre agentes

En un sistema donde existen varios agentes estos no solo van a interactuar con su entorno, sino también entre ellos. Por medio de la comunicación pueden dar a conocer sus metas, creencias y observaciones, con el propósito de lograr entre todos un objetivo común o como un medio para intentar cambiar el estado interno de los otros agentes. Un agente puede interactuar con otro de la siguiente manera: 1) ejecutando acciones sobre el entorno, y luego otros agentes observan el cambio en él. 2) Por medio de un *pizarrón* 3) Infiriendo las acciones de los demás agentes y 4) Pasándole mensajes directos. Antes de continuar, es necesario establecer el significado de los conceptos asociados a los siguientes términos.

Cuando existe un proceso industrial compuesto por varias recursos interactuando en una unidad de producción en pos de un objetivo común, se deben descartar las categorías 1) y 3) puesto que es necesario algun tipo de comunicación entre los agentes. El pizarrón se puede utilizar como mecanismo para sincronizar las actividades de los agentes, y el paso de mensajes cuando un agente que gestione a un proceso requiera recursos o servicios gestionados por otros agentes en la misma unidad de producción.

**Definición 5.5. Pizarrón** La siguiente es una definición formulada por Carven y Lesser [15], que establece que un pizarrón es una zona de trabajo común a varios agentes, donde cada agente coloca información para que la observen los demás. También establecen que un sistema de pizarrón está conformado por la base de datos, mecanismo de control y fuentes de conocimiento independientes de cada agente. Se puede implementar por medio de un conjunto de variables a las que tienen acceso los agentes, o una base de datos.

En cuanto a la comunicación pasándose mensajes directos, existe una jerarquía de lenguajes propuesta por Finin *et al* [38], que abarca desde los lenguajes que comparten procedimientos y estructuras de datos comunes, hasta aquellos que comparten experiencias y estrategias. Pasando por lenguajes que comparten conocimiento (abarca hechos, reglas, procedimientos y restricciones, tal y como se entienden en la especificación lógica y arquitectura de agentes) y que comparten intenciones (metas, planes, entre otros). Las implementaciones desarrolladas abarcan los dos primeros niveles de jerarquía, destacándose KQML y FIPA-ACL como los lenguajes mas conocidos y orientados al compartir conocimiento.

KQML [38] es un lenguaje de comunicación y protocolo orientado a los mensajes para el intercambio de información. Los mensajes definidos en KQML comunican una actitud sobre el contenido que llevan (una pregunta, una afirmación, una solicitud, por ejemplo). Los desarrollos que se basan en este lenguaje implementan un servicio de transporte de mensajes.

FIPA es un consorcio internacional que trabaja para acelerar el desarrollo de tenologías que involucran agentes físicos. Propone una arquitectura abstracta orientada al intercambio de mensajes, donde los mensajes representan *actos del habla*, y codifican a lenguajes de comunicación de agentes. FIPA ha expedido varios estándares en cuanto a la arquitectura de agentes, manejo, transporte de mensajes y comunicación. Este último aspecto se puede consultar en el reporte técnico que FIPA emite para comuni-

caciones [42], y es fundamental para implementar sistemas multi-agente que requieran del intercambio de mensajes.

Un mensaje típico de la arquitectura abstracta FIPA tiene una estructura expresada en forma de *tuplas (clave-valor)* donde se determina el lenguaje a utilizar, el contenido expresado en ese lenguaje de contenidos, referenciando a una ontología, nombres de emisor y receptor, y un campo de *carga útil* para implementar su transporte a través de una red de datos. La plataforma JADE implementa la arquitectura abstracta de FIPA y proporciona un conjunto de clases y paquetes desarrollados en Java para crear agentes, programación de la conducta de estos agentes, y servicio de mensajería FIPA ACL para intercambiar mensajes.

### 5.3.3. Mecanismos de negociación

Como entidades racionales, los agentes interactuan con otros agentes no solo para intercambiar información, sino también para asignarse tareas y recursos. Para llevar a cabo esta labor de asignación es necesario tomar decisiones sobre la naturaleza de la actividad y las metas de cada agente. Este proceso se denomina *negociación*. Antes de explicar los mecanismos para negociación entre agentes, es necesario establecer las siguientes definiciones:

**Definición 5.6. Negociación** Es un mecanismo para asignar tareas a agentes, o decidir cuales actividades se van a resolver. En algunas ocasiones los agentes planifican cursos de actividades que originan conflictos, sobre todo cuando van a utilizar un recurso determinado, entonces se utiliza negociación para resolver la asignación de recursos de una manera óptima.

**Definición 5.7. Protocolo de negociación** Es un conjunto de reglas que permiten que los agentes lleguen a acuerdos. Este concepto está relacionado con el de *dominios*.

**Definición 5.8. Dominio** En el contexto de agentes y negociación, un *dominio* es el ambiente en el que operan los agentes.

**Definición 5.9. Estrategia** Es la forma en que se comporta un agente cuando está negociando.

Rosenschein y Zlotkin [96] han estado investigando sobre mecanismos de negociación y han identificado tres tipos de dominios : dominios orientados a las tareas, dominios orientados al estado y dominios orientados al bienestar (TOD, SOD y WOD respectivamente), y propuesto un *protocolo unificado de negociación* basado en la teoría clásica de juegos. Otras investigaciones relacionadas con la negociación entre agentes proponen un modelo orientado a servicios (SOM), propuesto por Faratin [36], que establece dos protocolos de interacción, y tres mecanismos de toma de decisiones. En el campo de la automatización industrial, y orientado a los sistemas de manufactura flexible, cuyas celdas de fabricación asignan procesos y comparten recursos, Shaw y Whinston [101] han propuesto el framework de *Contract Net*, utilizando tecnologías de la inteligencia artificial distribuida, y luego extendido por otros investigadores en el área de la automatización y la inteligencia artificial distribuida. Este framework está implementado en diversas plataformas de desarrollo de sistemas multi-agente, de manera que el usuario que implemente los agentes no tenga que preocuparse por la programación de las tareas relacionadas con la comunicación entre agentes, ni tampoco por el mecanismo de negociación, permitiéndo así concentrarse solamente en el contenido de los mensajes que van a intercambiar los agentes. Por esta razón, se seleccionó una plataforma que cumpla con las características mencionadas anteriormente, las cuales se describen a continuación.

### 5.3.4. Plataformas tecnológicas de desarrollo

Para complementar las diversas metodologías de diseño de sistemas multi-agente que existen actualmente, se han producido diversas plataformas de desarrollo, las cuales facilitan las fases de implementación y prueba de sistemas multi-agente. Una de esas plataformas es JADE (*Java Agent Development Framework*) [14], el cual es un *middleware* que facilita el desarrollo de sistemas multiagente. Está basado en el lenguaje de programación Java, dentro de su librería de clases disponibles para los programadores se encuentran las diversas definiciones de tipos de agente, para usarlas directamente o especializando sus propios agentes, y además implementa el protocolo de comunicación entre agentes FIPA-ACL. Otras plataformas de desarrollo de sistemas multi-agente son ZEUS, INGENIAS y AnyLogic, esta última es un simulador multiparadigma. Para implementar las especificaciones de los agentes en la experimentación se selecciona JADE, debido a sus características de portabilidad, extendibilidad y sus herramientas de monitoreo.

Loa siguientes servicios e infraestructura de una aplicación multi-agente se implementan en JADE [13]:

- Ciclo de vida del agente y movilidad (instanciación y disposición final de agentes)
- Servicios de páginas amarillas y blancas
- Transporte y organización de mensajes de agente a otro
- Seguridad de agentes
- Programación de tareas para varios agentes
- Herramientas gráficas para ejecutar el monitoreo, registro de actividades y depuración
- Cumple con los estándares FIPA

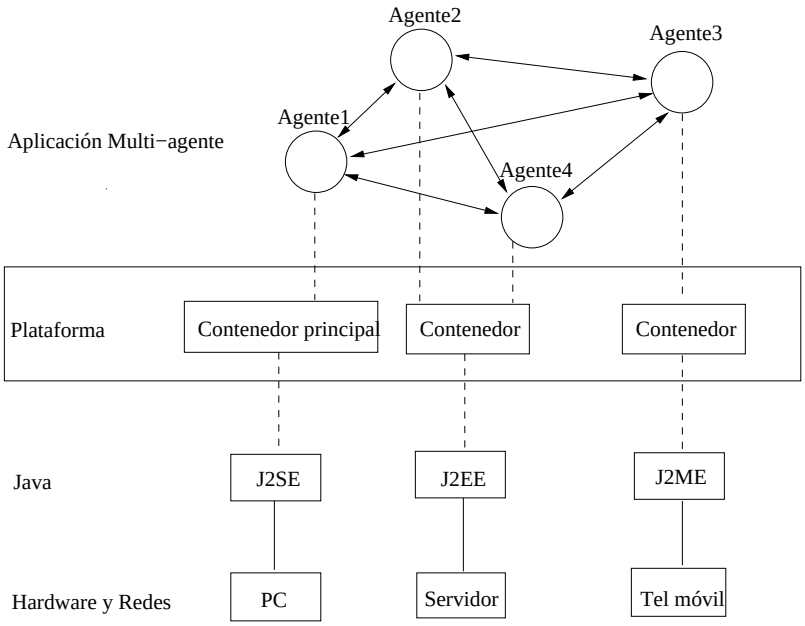


Figura 5.3: Arquitectura de la plataforma JADE

- Es un proyecto de código abierto

Un esquema que muestra la arquitectura de la plataforma JADE se muestra en la figura 5.3, donde se aprecian cuatro niveles: Aplicación multi-agente, la propia plataforma JADE, el lenguaje de programación Java, y los componentes físicos de hardware y redes. La capa de la aplicación multi-agente muestra la especificación y las interacciones que tienen lugar entre los diversos agentes de un sistema. En la capa de plataforma estos agentes se implementan en un *contenedor*, el cual se define como una instancia de JADE ejecutándose. El conjunto de todos los contenedores activos se denomina *plataforma*. El *contenedor principal* siempre tiene que estar activo dentro de una plataforma, y todos los demás contenedores se registran en el tan pronto empiezan a ejecutarse. El primer contenedor que se ejecuta en una plataforma es el contenedor principal. JADE implementa los servicios de comunicación y registro de contenedores de una manera automática. Los contenedores pueden ejecutarse bajo diversos entornos de Java, como lo es la edición estándar (J2SE), la empresarial (J2EE) o la edición micro (J2ME), la cual se implementa en dispositivos tales como teléfonos

móviles de última generación.

Para propósitos de esta investigación, se asumirá que los diseños de agente y su implementación en JADE se ejecutarán en la edición estándar de java o J2SE.

## 5.4. Generación de reglas a partir de la especificación de un Supervisor

Un agente no tiene un conocimiento por defecto inherente a la actividad que está desarrollando, requiere la intervención de un diseñador para agregar las creencias, las metas de alto nivel, y unas reglas que relacionen hechos observados con metas y con acciones. En este caso se tiene a un agente encargándose de administrar a un Holón de producción que contiene la representación de la dinámica de un proceso industrial y unos dispositivos electrónicos que toman información de un proceso físico. La dinámica del proceso corresponde a la proyección de valores continuos a un estado discreto, que corresponde a una forma de operación. La especificación lógica del comportamiento del agente puede clasificarse en los siguientes tipos de declaraciones:

- Reglas condición-acción para disparar metas a partir de eventos observados directamente
- Reglas para detectar eventos que no son observables directamente
- Procedimientos para derivar sub-metas a partir de metas de alto nivel
- Procedimientos para hallar acciones
- Observaciones que confirmen la ejecución de acciones
- Reglas para observar comunicaciones dirigidas al propio agente
- Procedimientos para comunicar hechos a otros agentes

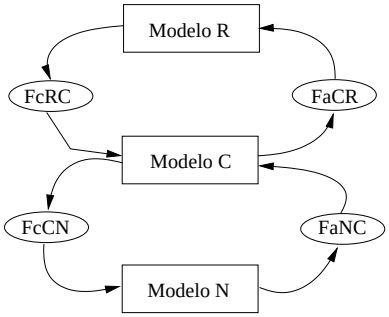


Figura 5.4: Esquema multiresolucional de Sanz, propuesta original

Una propuesta para generar reglas y que es útil en los entornos de control y automatización es la del *modelo multiresolucional* de Sanz [98], en el contexto del control inteligente, para generar decisiones a partir de variables del comportamiento dinámico de un proceso. El esquema original de la propuesta de Sanz se aprecia en la figura 5.4. Estos modelos multiresolucionales en ambientes de supervisión y control permiten simultáneamente:

- Describir comportamientos dinámicos de los procesos ante perturbaciones o acciones de control.
- Proporcionar información pertinente y actualizada sobre el estado de los procesos.
- Facilitar la comunicación entre procesos y usuarios.

La extensión al modelo multiresolucional de Sanz se muestra en la figura 5.5, donde el punto de partida es un modelo numérico aplicado a los principios físicos y mediciones de variables de interés (experiencia), sobre las condiciones de operación de los procesos (modelo N); un modelo cualitativo local (modelo $C_L$) para describir la lógica de las operaciones; un modelo cualitativo global (modelo $C_G$) que describe el estado general del proceso, y un modelo de razonamiento (modelo R) para representar el conocimiento en forma de reglas lógicas. La toma de decisiones en un sistema de control supervisorio se fundamenta en el uso de modelos apropiados capaces de describir

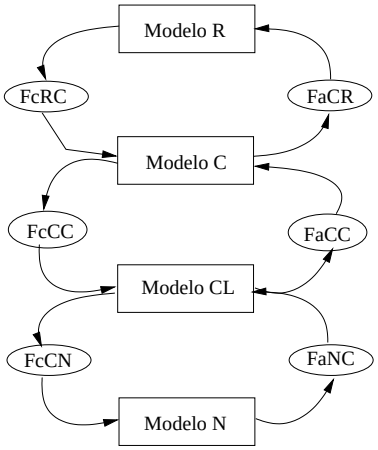


Figura 5.5: Esquema de razonamiento multiresolucional extendido

el comportamiento dinámico del proceso a controlar. Por medio de tales modelos es posible especificar acciones de control frente a la ocurrencia de perturbaciones aleatorias, o cambios en las regiones de operación del proceso, que garantizan su estabilidad y confiabilidad operacional. Esos modelos están acoplados por medio de funciones de abstracción y funciones de concreción, para especificar valores de variables de un nivel a partir de valores de variables en otros niveles.

El esquema donde se muestra la combinación de la extensión del modelo multiresolucional con el control supervisorio se puede apreciar en la figura 5.6, el cual es uno de los aportes de nuestro trabajo doctoral y se describe con mayor detalle en el anexo D. El comportamiento del Proceso Físico que está situado en el piso de planta se describe por un modelo numérico que aplica principios físicos, como conservación de masa y energía, así como las mediciones de variables de interés, las cuales proporcionan las condición de operación en que está el proceso. Este proceso industrial está expuesto a perturbaciones, de manera que los datos que miden la evolución del proceso físico son muestreados y llevados al Detector de Eventos, el cual a partir de los valores medidos y el estado actual del proceso aplica una función de abstracción que permita indicar que ocurrió un evento, y lo transmite al Supervisor, el cual tiene los modelos cualitativos y racional de todo el proceso. El evento que recibe el Supervisor se aplica

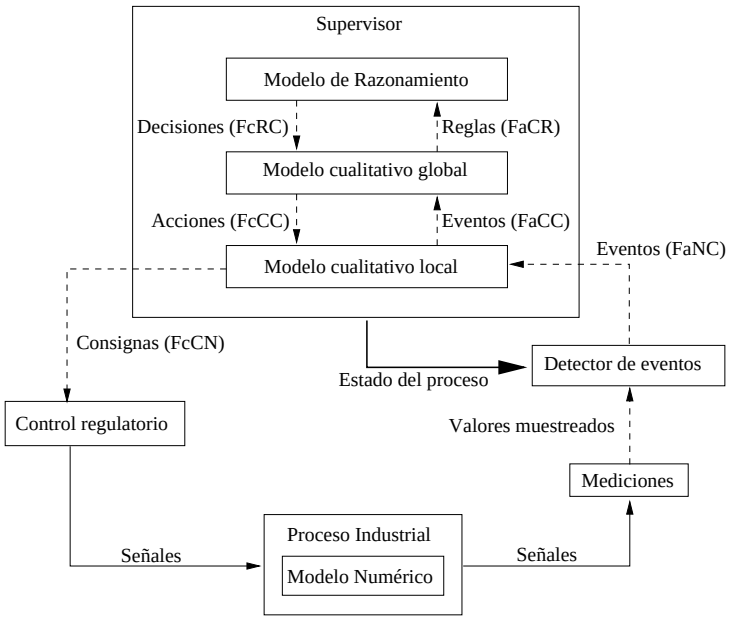


Figura 5.6: Combinación de SCS con modelo multiresolucional extendido para generar reglas

a un Modelo Cualitativo Local, que contiene las condiciones de operación ideales, o normales. Algunos de estos eventos detectados sugerirán eventos de orden global, con la función $Fa_{CC}$, lo que sugiere la ocurrencia de eventos que indican ocurrencia de fallas. El modelo racional contiene una proyección desde un sistema a eventos discretos (DES) global y local, hacia unas reglas que el Supervisor traduce en acciones enviadas en forma de consignas hacia el control regulatorio. El Supervisor contiene una planificación de secuencias de eventos, los cuales va generando de acuerdo a las reglas que contiene el modelo R.

A partir del modelo racional se aplican unas funciones de concreción de razonamiento, en forma de actualización de los estados tanto del modelo cualitativo global (cuando se recupera una falla), como del modelo cualitativo local (operación en condiciones normales). A partir del modelo cualitativo local se generan las reglas $Fc_{CN}$ del control regulatorio hacia el proceso físico, que se traduce en consignas al controlador para inducir cambios en las propiedades físicas del proceso, que a su vez afectan al modelo numérico, y aquí el ciclo se reinicia con procesos de medición y detección de eventos.

Es necesario aclarar que para procesos industriales de cierta complejidad se requiere una mayor cantidad de detectores de eventos, al menos uno para cada tipo de variable a medir, así como también crece el espacio de los modelos de razonamiento y cualitativo, aunque este último se puede asignar a un solo agente que desempeña el rol de supervisor.

## 5.5. La Unidad de Producción como un sistema multiagente

Ahora se llevarán las ideas de la Teoría de Sistemas Multiagente para describir el comportamiento de una organización industrial, la cual se compone de Unidades de Producción Holónicas, cuyo componente racional se implementa por medio de Agentes, los cuales disponen de la posibilidad de comunicarse entre sí. La correspondencia propuesta tiene la siguiente forma:

- El *estado del entorno* es una estructura que contiene los estados de las variables de cada unidad de producción, los cuales a su vez son estructuras que contienen variables continuas como símbolos para representar los estados.

- Los agentes perciben su entorno de dos maneras: los eventos observados directamente, o sea los observables, y los valores que reportan los sensores, para deducir la ocurrencia de eventos, no observables directamente. Además, dada la naturaleza perturbable de las mediciones hechas por los sensores, es probable que algunos eventos no se detecten, o se detecten luego de cierto tiempo transcurrido después de su ocurrencia real. También pueden observarse eventos que corresponden a actos de comunicación directa o indirecta de los demás agentes que administran unidades de producción.

- Las influencias que los agentes producen se pueden clasificar de dos maneras:

actos de comunicación con otros agentes, como envío de mensajes, por ejemplo, o acciones ejecutadas sobre la consigna de los controladores, que generan cambio en los parámetros del proceso industrial que se quiere controlar.

- Las leyes del sistema ($\Lambda$) vienen dadas por la física de la planta, y por la lógica de estados de cada proceso de producción. La función *reaccion*() del sistema consiste en el cambio de alguna región de operación cada vez que se lleve a cabo un ajuste en los parámetros del controlador correspondiente, los cuales a su vez cambian las condiciones físicas del proceso industrial, afectando a las ecuaciones del sistema.

Dada la heterogeneidad de los sistemas de manufactura debido a los diversos procesos que maneja sobre una amplia variedad de organizaciones, es difícil establecer una organización basada en agentes que refleje las complejas inter-relaciones que se puedan llevar a cabo. De esta manera se dificulta la concepción de una Unidad de Producción utilizando asignación de roles a unos agentes basado en la descomposición funcional, por ejemplo. Por el contrario, la Arquitectura de Referencia PROSA proporciona una manera elegante de resolver este problema, como lo establecen Valckenaers *et al* [106], donde proponen un marco de referencia para dotar a la arquitectura de referencia PROSA las cualidades de un sistema emergente, por medio de la asignación de un agente a cada componente racional de un holón, obteniendo así agentes Orden, Producto y Recurso, los cuales se encargarían de un aspecto del control de manufactura: logística, método de producción y manejo de recursos, respectivamente. En otro trabajo posterior, Valckenaers *et al* combinan estos agentes PROSA con agentes de optimización que trabajan con la filosofía *colonia de hormigas* para optimizar la coordinación de un entorno de manufactura multi-agente [107]. Estos agentes se estructuran aprovechando los conceptos de orientación a objetos como especialización y agregación, obviando de esta manera la complejidad de una organización industrial. Además, se produce de una manera directa la separación de los roles de cada agente.

Para el caso de la supervisión de la conducta de la unidad de producción, se agrega la interacción de agentes adicionales: al menos un Supervisor, que siempre tiene un *modelo del Estado del Producto*, y cuya misión es la de cumplir con una Orden (tareas) manteniendo el Estado del Producto, comunicándose con los agentes PROSA para modificar la configuración de la planta; y por otro lado un conjunto de Detectores de Eventos, los cuales interactúan con mediciones del proceso físico para indicar cambios en condiciones de operación.

En la asignación de los roles para los agentes que gestionarán a un sistema holónico se va a seguir una metodología combinada: por un lado, la propuesta de Valckenaers *et al* de asignar a cada holón PROSA un agente, en lugar de la asignación de funciones a un agente, y por otro lado, se agrupa por su funcionalidad los roles de los agentes supervisor, detector de eventos y actuador. En la figura 5.7 se muestran las interacciones entre los agentes que conforman un HMS PROSA, con los agentes que implementan el control supervisorio del sistema holónico. Los modelos multi-resolucionales que se diseñen para el control de los procesos dentro de una unidad de producción holónica son el punto de partida para generar el repositorio de reglas para los agentes que van a gestionar esta unidad de producción.

Con el fin de proporcionar flexibilidad, las reglas que determinan el comportamiento de los agentes detectores de eventos se organizarán en un repositorio, de manera que cuando un agente se instancie haga la carga dinámica de sus reglas, y de alguna manera se puedan descartar o cambiar, según cambien las condiciones de operación. Para el agente que tiene el rol de supervisión se podrían cargar reglas de acuerdo a cambios en la meta de producción. Las interacciones entre los agentes son directas, por medio del intercambio de mensajes. Cada agente Recurso y Detector de Eventos interactúan directamente con los dispositivos asociados: los Recursos con el equipamiento, y el Detector con los sensores que miden las variables del proceso físico.

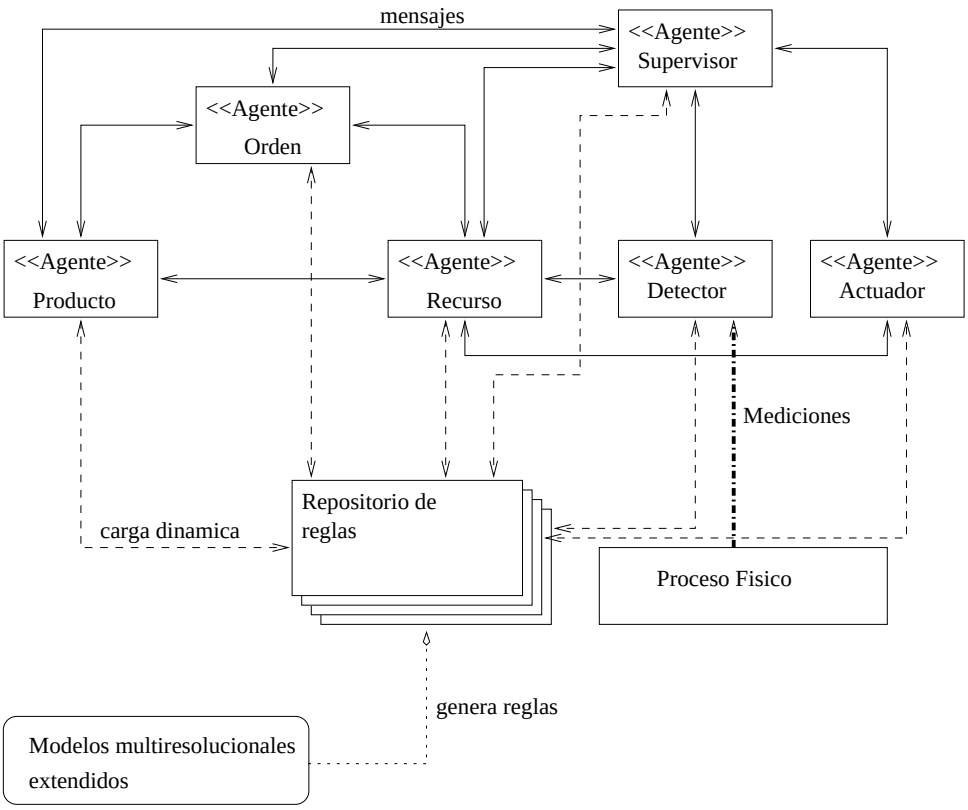


Figura 5.7: Implementación de un HMS con tecnología de agentes

## 5.6. Conclusión

Uno de los aportes de este capítulo está en considerar la unidad de producción como un sistema multi-agente, donde cada uno de estos agentes ejecuta las funciones asignadas a los holones que componen dicha unidad de producción. Además de la gestión de holones, los agentes pueden asumir también las funciones de supervisor y detector de eventos, entre otras, para proporcionar a cada holón la capacidad de supervisar su funcionamiento y comunicarse con otros holones para negociar capacidades de producción, entre otros aspectos. El otro aporte consiste en que se pudo combinar la especificación de la conducta de agentes basada en lógica con los modelos multiresolucionales extendidos. La conducta de estos agentes se puede especificar lógicamente a partir de las metas de cada agente, y también se pueden proyectar a sistemas a eventos discretos gracias a los modelos multiresolucionales, los cuales se han extendido en este trabajo para brindar mayor robustez hacia aquellos sistemas cuya conducta se debe supervisar.

# Capítulo 6

# Modelado Conceptual de la Supervisión del Sistema Holónico basado en PROSA

En las secciones anteriores se ha presentado la descripción de la dinámica de una Unidad de Producción Autónoma (UPA, de aquí en adelante) cuya concepción holónica está inspirada en la arquitectura PROSA, los mecanismos de supervisión asociados a un Sistema Holónico de Manufactura, y la especificación del componente de información y cooperación utilizando tecnologías de agentes; ahora se va a especificar de una manera preliminar la manera en que estos objetos se implantarían dentro de un entorno industrial donde los procesos son continuos, utilizando una arquitectura tecnológica de computación y telecomunicaciones. Para ello se utilizarán técnicas de modelado conceptual para describir la estructura y conducta de todo el sistema utilizando el Lenguaje Unificado de Modelado (UML, de aquí en adelante) definido por Booch *et al*, [9], y luego de ello su interacción dentro de una plataforma de automatización estándar. Una forma conceptual de capturar los requerimientos del sistema es por medio de los *diagramas de casos de uso*, los cuales se obtendrán aplicando la

metodología propuesta por Ortín *et al* [84] para obtener el modelo de requisitos a partir del modelo de negocios. Estos diagramas UML describen de una forma general al sistema, y las relaciones de este con su entorno. Además, proporcionan las bases para su implementación en cualquier arquitectura computacional. Antes de continuar, es necesario establecer una definición de *Modelado Conceptual*.

**Definición 6.1. Modelado Conceptual** Desde el punto de vista del Modelado y Simulación, Fishwick [40] establece que aquellos modelos que contienen componentes que no se hayan identificado con claridad en términos de teoría de sistemas, como *estado, evento y función* se denominan *Modelos Conceptuales*. Así, el modelado conceptual consiste en describir un sistema enfatizando en sus objetos y relaciones. Estos modelos conceptuales se pueden reformular luego como modelos orientados a objetos. Las técnicas de modelado conceptual orientadas a objetos permiten establecer un conjunto de clases para describir la estructura y funcionamiento de un sistema dado, basándose algunas de ellas en los *casos de uso*, y otras en las historias de eventos de usuario.

Para aplicar la propuesta de Ortín *et al* el primer paso para obtener el modelo del negocio consiste en capturar sus procesos de negocio, que en este caso es el de una UPA, los cuales definen los límites del modelado que se haga posteriormente. Independiente de la organización de producción a la que se le haga modelo del negocio, se va a dirigir hacia una Unidad de Producción continua, y para ello se identificarán los *objetivos estratégicos* de la Unidad de Producción, siguiendo el concepto de objetivo estratégico de Cockburn [24]. Estos objetivos se descomponen en subobjetivos, de manera que se establezca una jerarquía. Para cada objetivo que no se pueda descomponer, se define un proceso de negocio cuyo propósito será lograr ese objetivo. Cada proceso de negocio se representará mediante un*caso de uso del negocio*.

Aplicando estos conceptos a la Unidad de Producción holónica que maneja procesos

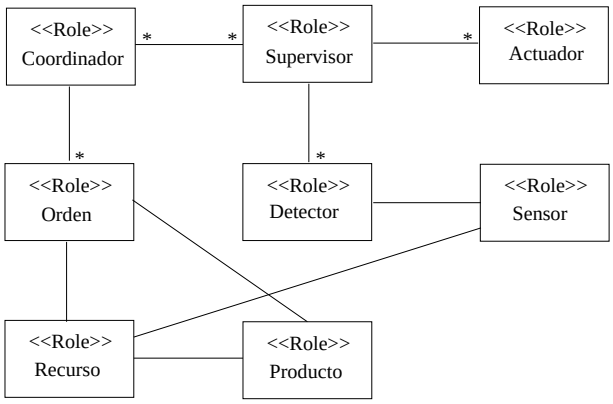


Figura 6.1: Diagrama de Roles para el proceso de negocio de la UPA

continuos, se establece que el objetivo estratégico es *satisfacer una orden de producción de una manera permanente*. Este objetivo se descompone en i) planear la orden de producción, ii) programar las actividades (asignar sub-órdenes a otros holones) y iii) Ejecutar la orden de producción. Este último objetivo puede descomponerse en a) Representar el estado inicial del proceso, b) Detectar eventos, c) Actualizar estado de la Unidad de Producción y d) Ejecutar medidas de control. Los actores involucrados en el cumplimiento de estos objetivos son de dos tipos: actores internos y actores externos. Como actor externo se puede considerar a otro holón o una persona, los cuales proporcionan las especificaciones de lo que se va a producir. Como actores internos se tienen los gestionadores de cada holón, y los encargados de ejecutar la supervisión del proceso: supervisor, detectores de eventos, sensores y actuadores. Los diversos actores y sus relaciones se pueden observar en el *diagrama de roles* que se muestra en la figura 6.1, en la se aprecia que el actor externo se ha denominado *Coordinador*, y el cual puede asimilarse a otro holón externo a la unidad de producción, un holón que contenga a la unidad de producción o inclusive una persona. Este holón genera órdenes de producción gestionadas por un holón Orden, y asigna al supervisor las pautas para controlar la ejecución de la orden. Como el proceso es continuo, una nueva orden reemplazará la anterior, con los consiguientes cambios en la planta para ejecutar dicha orden.

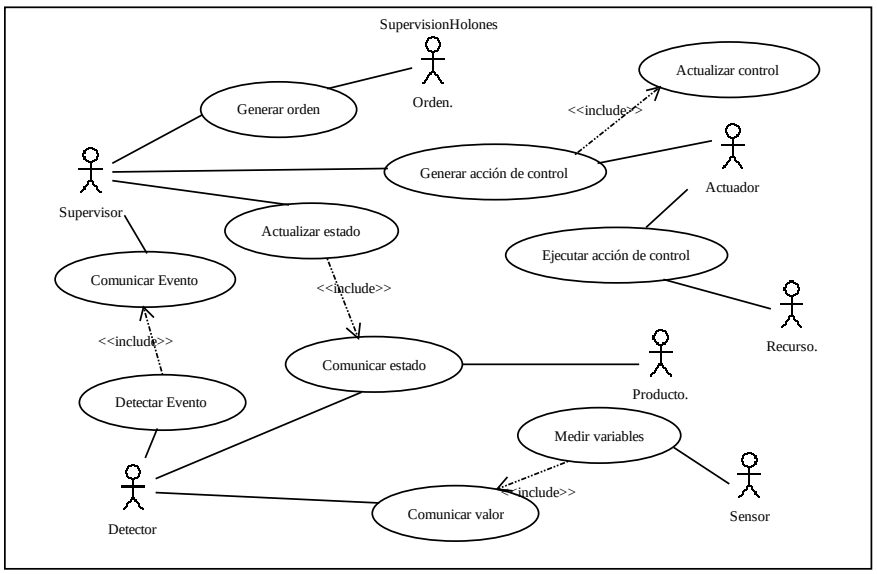


Figura 6.2: Diagrama de caso de uso general

Una vez obtenido el modelo del negocio por medio de la descomposición de objetivos, identificado actores y asignado sus roles, se puede construir el *diagrama de caso de uso* de la Unidad de Producción. Un diagrama de caso de uso general se muestra en la figura 6.2, donde se identifican los actores y casos de uso que intervienen en la supervisión de un proceso industrial. Por *actor* se entiende que es todo lo que interactúa con el sistema, bien sea una máquina o una persona. Por *caso de uso* se entiende una secuencia de actividades ejecutadas por el sistema, lo cual produce valores medibles para un actor particular. El objetivo de utilizar este diagrama es el de presentar la funcionalidad del sistema supervisado y como base del modelado conceptual estructural y dinámico de la unidad de producción.

Como se puede observar en la figura 6.2, se han identificado siete actores dentro de la Unidad de Producción. Están los actores que gestionan a los holones PROSA, y además cuatro actores que son: sensor, detector de eventos, supervisor y actuador. Estos actores representan roles que puede desempeñar el sistema de automatización industrial en un momento determinado, y en sí se pueden considerar como sistemas

autónomos, implementados por agentes, cuyos aspectos se describieron en el capítulo cinco de este documento. La manera en que estos actores utilizan las funcionalidades del sistema proporciona los casos de uso identificados en la figura, los cuales se relacionan en la siguiente lista:

- *Medir Variable*, que incluye el caso de uso *comunicar valor* del proceso físico. El actor Sensor (un dispositivo) periódicamente está midiendo las magnitudes de las variables a controlar, y registra los valores medidos. comunica al actor Detector de Eventos los valores registrados.

- *Detectar eventos*, ocurre cuando el actor detector de eventos le indica al supervisor que ha ocurrido un evento, lo que desencadena un flujo de mensajes que incluye el caso de uso *comunicar eventos* cuando el Detector da cuenta al Supervisor de la ocurrencia de algún evento.

- *Actualizar estado*, lo ejecuta el actor Supervisor, una vez ocurre algún evento. La imagen del proceso se actualiza y el supervisor comunica el nuevo estado al Detector de eventos, para dar cuenta del cambio de condiciones de operación y cambiar así el modelo multiresolucional de detección de eventos. Incluye el caso de uso *comunicar estado*.

- *Generar acción de control* lo inicia el Supervisor cuando el sistema se encuentra en un estado determinado, y desea llevarlo a otro estado, o iniciar una secuencia de eventos que impidan que el sistema llegue a un estado no deseado, según su modelo cualitativo del sistema a controlar. El actor Supervisor le envía comandos al actor actuador para que los ejecute. El caso de uso *Actualizar Control* está incluido dentro de esta operación.

- *Ejecutar acción de control* es un caso de uso que utiliza el actor actuador, cuando ya tiene un comando a ejecutar que le ha indicado previamente el actor

supervisor. El actuador ejecuta esta orden modificando las consignas de operación de los recursos involucrados en el proceso a controlar, notificando así al actor que gestiona el recurso en cuestión.

- *Generar Orden* es una funcionalidad que está relacionada con la acción de control que genera el Supervisor, y se le envía al actor que gestiona las órdenes de producción para sugerirle cambios en la cantidad del producto solicitado, por ejemplo.

## 6.1. El Lenguaje Unificado de Modelado

UML es el acrónimo de Unified Modeling Language, inicialmente un estándar para dibujar diagramas orientados a objetos, que luego evolucionó para ser utilizado en el modelado de procesos organizacionales y empresas, dando origen a un desarrollo posterior de orientación específica a la industria: SYSML [50]. Cuando se modeló la arquitectura de referencia PROSA se utilizó UML debido a que los conceptos de los sistemas de manufactura holónica son muy similares a los del diseño orientado a objetos. Esta es una de las razones por las que se utiliza UML como lenguaje de modelado conceptual del control supervisorio de un sistema holónico. Otras razones para utilizar UML se relacionan con la diversidad de diagramas no solo para representar la estructura de un sistema dado, sino también la dinámica del mismo, cosa muy necesaria para modelar el funcionamiento del control supervisorio, por ejemplo. Otra razón consiste en la abundancia de herramientas computacionales que permiten no solo editar estos diagramas, sino generar plantillas de código fuente en varios lenguajes de programación, con lo que se reduce el tiempo de codificación de software. Además, para aspectos de validación, se puede llevar cualquier modelo conceptual a varios formalismos de simulación y así ensayar la validez del diseño propuesto. Por las razones antes mencionadas, se utilizó UML como herramienta de modelado conceptual del

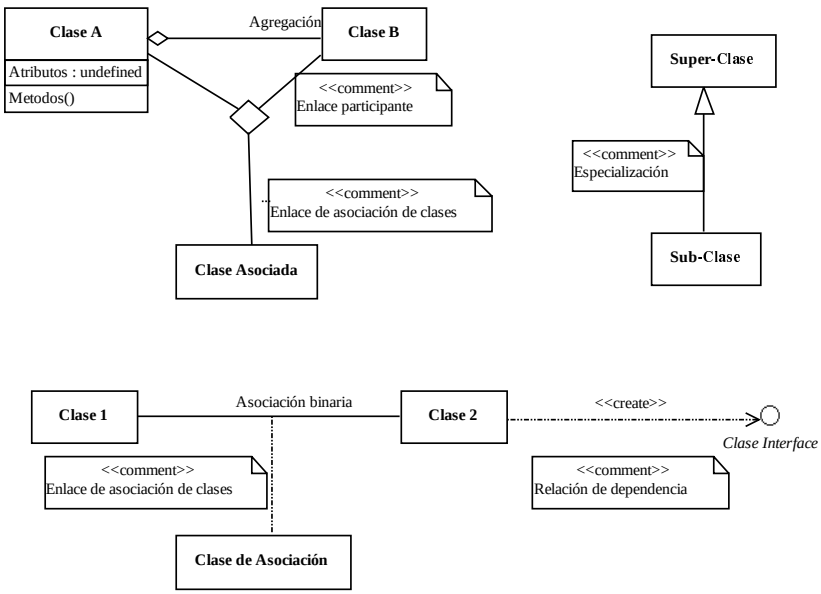


Figura 6.3: Esquemas para interpretar un Diagrama de Clases UML

control supervisorio en sistemas holónicos.

Para modelar la estructura computacional de una unidad de producción se recurrió a los *diagramas de clases* [49, 9], que representan los objetos abstractos que manejan la información relacionada con el funcionamiento de esta unidad. Para entender mejor este tipo de diagramas se incluye una figura donde se muestran los diversos aspectos que ayudan a interpretar estos diagramas. Estos se pueden apreciar en la figura 6.3, donde se muestra el dibujo general de las clases, que representa al modelo de la estructura estática de un sistema, y los tipos de relaciones posibles que pueden existir entre ellas. Los rectángulos representan a las clases, que son plantillas de objetos, y los diversos tipos de líneas representan las relaciones de agregación (componente/parte), especialización (herencia) y asociación. Esto ayudará a comprender mejor los diagramas subsiguientes, donde se mostrarán las clases que intervienen dentro de una unidad de producción, y las relaciones que existen entre ellas.

## 6.2. Estado de la Unidad de Producción basada en la filosofía PROSA

Una descripción orientada a objetos basada en el lenguaje unificado de modelado (UML) para la UPA consiste en un conjunto de clases, donde la información que se maneja está relacionada con la disponibilidad de los recursos, objetivos de producción, métodos de producción y la dinámica del proceso continuo. Como suposición inicial se tiene que la UPA contiene elementos basados en la arquitectura holónica de referencia PROSA [10], y el diagrama de clases que relaciona esta arquitectura con la propuesta propia de supervisión de la UPA se puede apreciar en la figura 6.4. En esa figura se muestra que el punto de partida son los *holones Recurso, Producto y Orden*, los cuales son especializaciones de la clase *Holón*. Una UPA se compone de estos tres holones básicos, pero para implementar su supervisión se les asocia a cada uno de estos una interfaz, que se generaliza en la *metaclase Agent*, tomando su implementación del entorno multi-agente JADE descrito en el capítulo anterior. También son especializaciones de esta meta-clase las interfaces que asumen las funciones de Supervisor, Actuador, Sensor y Detector de Eventos, diferenciándose solamente en el conjunto de reglas que van a regular su conducta, las cuales se establecen por los modelos multi-resolucionales asignados a cada rol. Para llevar a cabo la Supervisión y Control de la UPA es necesario considerar además la información que generan las relaciones entre estos holones.

De esta manera, a partir de la *asociación* entre holón Recurso y Producto se obtiene la *clase Tareas*, que mantiene información sobre las tareas que hay que desarrollar en un recurso para obtener un producto, como se ejecuta un proceso en un recurso, las capacidades del recurso, y el estado de los recursos, los cuales se describen en la clase *EstadoRecursos*. La *clase MetodoProduccion* surge de la asociación entre holones Producto y Orden, y maneja información sobre cómo obtener un producto, dados ciertos

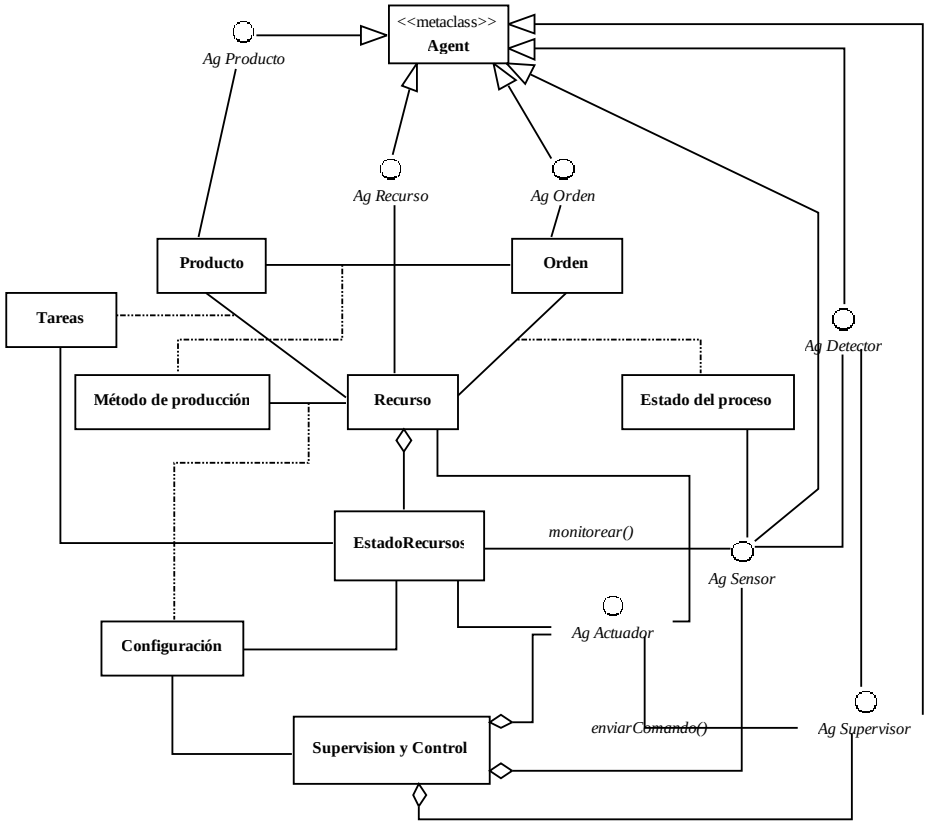


Figura 6.4: Diagrama de clases para la unidad de producción dentro de la Filosofía PROSA

recursos, y las secuencias de procesos que se deben ejecutar. La *clase EstadoProceso* resulta de la relación entre Recurso y Orden, y contiene la información asociada al progreso de la ejecución de los diversos procesos, como iniciar un proceso, como reservar recursos, monitorear el proceso, como interrumpir un proceso, como reanudarlo, entre otras. La clase *Supervisión y Control* tiene a cargo las funciones relacionadas con la supervisión de la conducta de la UPA, y el control de la configuración de la planta. La clase *Configuración* resulta de la relación entre el método de producción y los recursos asignados para desarrollar este método. En los sistemas de producción continua, la configuración de la planta se mantiene durante un intervalo de tiempo hasta que se genere otra orden de producción que amerite el cambio de configuración. El mecanismo de supervisión de la UPA está relacionada con estas clases, y a continuación se va a describir su estructura y comportamiento, así como un escenario de su aplicación a sistemas de producción continuos.

### 6.2.1. Relación entre Estado y Configuración de la Unidad de Producción

La clase *Unidad de Producción* se compone de otras clases, que manejan información sobre el estado de la unidad de producción, los métodos de producción, los recursos, el supervisor, este último asociado con una misión de producción. Además la Unidad de Producción puede contenerse así misma, de esta manera se obtiene una unidad de producción compleja a partir de la composición de otras unidades de producción. Así mismo, la dinámica del comportamiento de esta unidad de producción se representa como un Sistema Dinámico Acoplado (SDA), puesto que se compone de varias dinámicas discretas y continuas. Se observa la relación que tienen en la figura 6.5, adaptada de la propuesta de Chacón *et al* [18], donde se establece que la información asociada a la evolución de la unidad de producción se encuentra en la *clase Estado Unidad de Producción.*

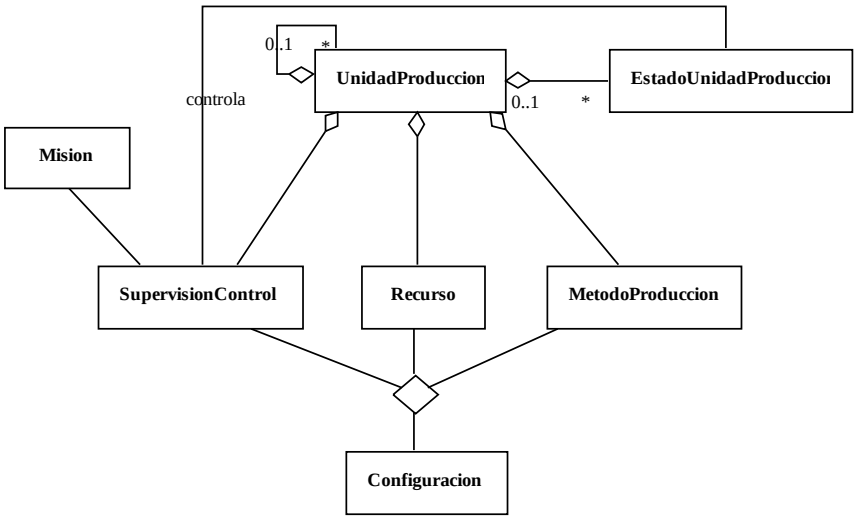


Figura 6.5: Unidad de Producción, relación con la configuración y su estado actual

En la figura también se aprecia que la misión está asociada a la clase *Supervisión y Control*, cuya función es la de mantener el estado de la unidad de producción dentro de los estados deseados posibles utilizando para ello algoritmos de control. Para cumplir con su función se relaciona con la clase Recurso, y con la clase *Método de Producción*, que describe la relación entre producto a obtener y orden de producción. La Supervisión tiene como objetivo monitorear la ejecución de un método de producción, el cual utiliza recursos que a su vez son monitoreados. La interacción resultante permite definir la clase *Configuración*, que describe como está organizada la UPA en un instante determinado para ejecutar una actividad. Esta configuración se expresa en términos de estado de los recursos, actividades a realizar y puntos de control (setpoints) donde el supervisor comparará la ejecución de tareas con las actividades programadas. La configuración se puede aplicar en un intervalo determinado, con un tiempo definido de inicio o final, o puede tener un tiempo inicial en que se aplica, y luego se cambia de acuerdo a las indicaciones del supervisor.

### 6.2.2. Supervisión y Control del Estado de la Unidad de Producción

Las teorías de control, supervisión y detección de eventos son útiles para llevar a cabo la función de monitoreo, y por lo tanto deben considerarse en la implementación del sistema de información que registra el comportamiento de la unidad de producción. Para poder supervisar el comportamiento de esta unidad de producción, es necesario conocer su estado interno, y para ello se recurre a las clases *Estado Unidad de Producción* y *SupervisiónControl* en una asociación que permite relacionar los mecanismos de control con los estados en que se encuentran los recursos y el proceso ejecutado. Para conocer mejor el estado de la unidad de producción, se separa el estado físico de los recursos a utilizar del estado lógico del proceso de producción (la ejecución de un método de producción). Para ello se utilizan las clases *Estado de Recurso* y *Estado de Proceso*. A su vez, el *Estado del Control* es parte de la Supervisión y Control de la UPA. Unas clases necesarias para controlar la evolución dinámica de la unidad de producción son las que implementan al *Supervisor*, *Detector de eventos* y *Actuador*, las cuales interactúan con el estado global de la unidad de producción y con el estado de las actividades de control, y son parte de la clase configuración y control, la cual actualiza la configuración de la planta.

Un supervisor acoplado a la Unidad de Producción que ejecute ciertas acciones puede cumplir su misión si conoce el estado de la Unidad de Producción. Este supervisor se describe como un sistema de eventos discretos, y tiene un estado interno, que se considera como el estado del control. En la figura 6.6 se observa que las clases que representan al estado del proceso y al estado del control están asociadas a una clase *DES*, diseñada para representar sistemas a eventos discretos de diferente naturaleza, cuyos componentes mas importantes son los Autómatas de estados finitos, y las redes de Petri, las cuales se describirán más adelante. Esta clase implementa además los

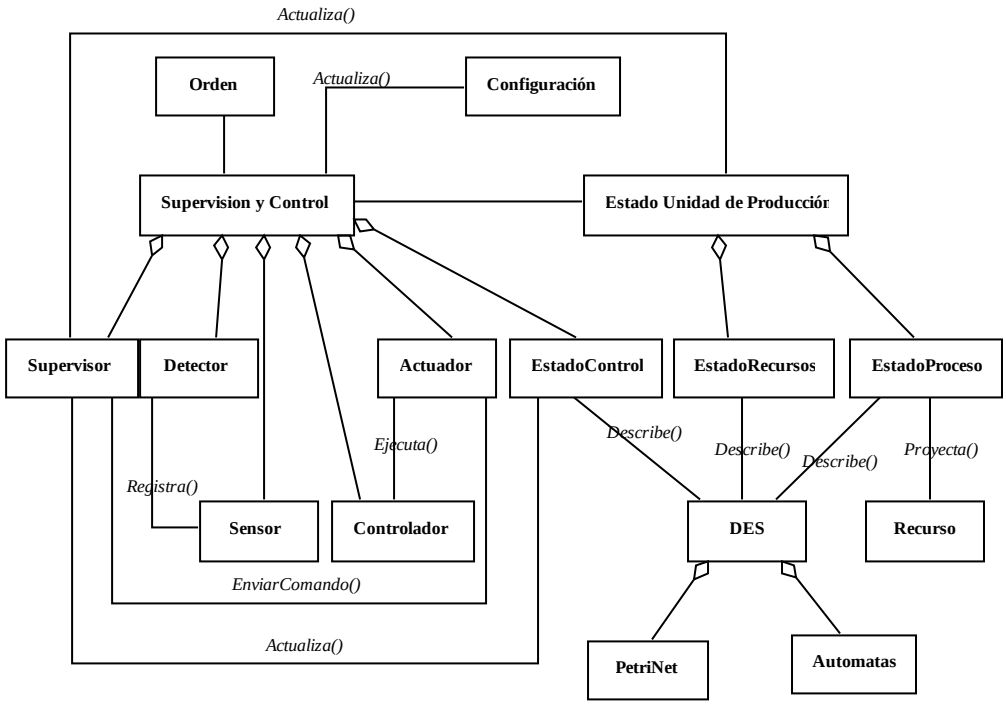


Figura 6.6: Diagrama de Clases de la Configuración Estática de la Unidad de producción

métodos de análisis de este tipo de sistemas, como lo es el análisis de alcanzabilidad, el manejo de transiciones y el control de conducta, expresado en control de secuencias de eventos. En lo que queda de esta sección se describirán los estados de los recursos, del proceso y del control.

Para poder implementar correctamente la clase *Supervisión y Control* es necesario definir y diseñar las clases que van a manejar la información sobre el *Estado de la Unidad de Producción*, que de acuerdo a la arquitectura PROSA y a la especificación de la conducta del Sistema Holónico, debe registrar la información asociada a sistemas a eventos discretos (DES), tal y como se describen en la sección tres. Este estado se desagrega en el *estado de los Recursos*, y en el *estado del Proceso*, el cual está asociado a la ejecución de una orden de producción. Los aspectos de modelado conceptual de ambos estados se describirá a continuación.

## 6.3. El estado de los recursos

Como *Recursos* se consideran no sólo el equipamiento para ejecutar las operaciones productivas, sino también la energía a utilizar, los servicios de transporte, y el recurso humano, los cuales son necesarios para ejecutar las actividades de producción. Un atributo en común de todos estos recursos lo es la unidad de medida, ya que establece como se va a cuantificar la cantidad o disponibilidad de un recurso. Otro atributo en común es el *Estado*, el cual está asociado a la clase *Estado Recurso*. Así mismo, esta clase Recurso implementa los métodos necesarios para representar la reserva un recurso determinado, su utilización durante un intervalo de tiempo y posteriormente su liberación para que vuelva a estar disponible en un futuro. Todos estos recursos implican restricciones sobre el proceso de producción, y de alguna manera se proyectan a una configuración definida para llevar a cabo un método de producción determinado. El conocer el Estado de la UPA, tanto en sus recursos como en el proceso que se lleva a cabo permite que el supervisor pueda controlar a esta unidad, y a su vez mantenga actualizada una imagen del proceso.

Para ubicar al estado de los recursos dentro del modelo de negocios de la unidad de producción, considérese el sub-objetivo *Actualizar estado de la Unidad de Producción*, el cual podría subdividirse en: actualizar estado del recurso y actualizar estado del proceso. Una breve descripción operacional tiene la siguiente forma: la clase *EstadoRecursos* recibe información de los estados de los recursos que están operando por medio de la clase *Supervisor*, y algunos recursos interactúan con la clase *Detector de eventos* a través del *Sensor*, el cual a su vez es un recurso especial, que está midiendo variables de los recursos asociados, así como del proceso de transformación. Esta clase informa al supervisor sobre la ocurrencia de eventos que cambien a este estado, con el fin de que este ejecute algunas acciones para mantener el proceso con el comportamiento deseado. Otros atributos y operaciones de esta clase conforman a su vez

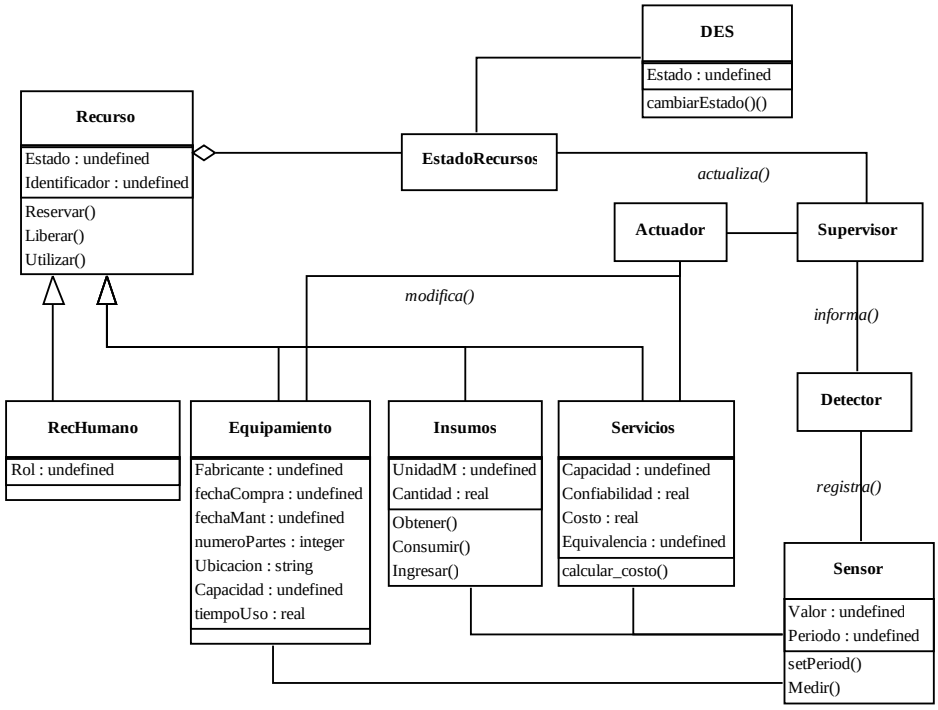


Figura 6.7: El Estado de los Recursos

otras clases, cuyo diagrama UML estático se puede observar en la figura 6.7. Como se puede observar en el diagrama, la clase *EstadoRecursos* describe el estado en que se encuentra un determinado recurso. Como existen varios tipos de ellos, esta clase se considera como una plantilla para las clases que son mas específicas en describir la naturaleza del recurso al cual se le quiere describir su estado: *Servicios*, *RecHumano*, *Equipamiento*, cada una con su conjunto de atributos y operaciones específico a la naturaleza del recurso que están describiendo. Una descripción de la información de cada clase se establece a continuación.

- clase Insumos: Describe al estado de los insumos necesarios para ejecutar la orden de producción, manejando información sobre el material que está fluyendo sobre los otros recursos. Dentro de sus atributos están la unidad en que se mide este insumo (kg, litros, cm, u otras), y la cantidad disponible. Como operaciones asociadas a esta clase se tienen: obtenerCantidad() para reportar la

cantidad actual, consumir() para disminuir su cantidad, ingresar() para indicar que ingresa determinada cantidad que se agrega a la ya disponible. Es necesario aclarar que consumir() e ingresar() también pueden denotar unas tasas de flujo saliente o entrante del material, ya que su naturaleza puede ser continua.

- clase RecHumano: Describe el estado en que se encuentra el Recurso Humano asociado al proceso productivo. Se han establecido los atributos de identificador y rol asignado.

- clase Equipamiento: Maneja la información concerniente al equipamiento involucrado en el proceso de producción. Entre sus principales atributos se tienen *fabricante, fecha de compra, fecha mantenimiento, número de partes, ubicación*, los cuales describen al equipamiento, así como la capacidad que maneja este equipo (potencia eléctrica, peso máximo, potencia de carga, volumen), el valor asociado a esta capacidad y el tiempo de uso. Entre las operaciones que maneja se encuentra la de asignar este recurso y revisar su estado. Esta clase se relaciona con la clase *Sensor*, donde esta última ejecuta las funciones de medir algunas variables asociadas al equipamiento y al proceso que se desea controlar, y las va registrando en una estructura de datos. Además de las variables y los valores a medir, es importante conocer el período de medición del sensor, y un mecanismo para variarlo, como lo es *setPeriod()*, por ejemplo.

- clase Servicios: Maneja información relacionada con los servicios utilizados para llevar a cabo el proceso productivo. Dentro de sus atributos están la capacidad nominal, la confiabilidad del servicio, el costo por utilizar una unidad de servicio, y la equivalencia entre la unidad de servicio y el trabajo realizado. Esto permitirá ejecutar las operaciones de calcular el costo de uso del servicio. En general, cada equipamiento está asociado a la utilización de por lo menos un servicio, con el consiguiente costo asociado por hacer funcionar un equipo determinado.

### 6.3.1. Eventos en la operación de los Recursos

A cada uno de los recursos que pertenecen a la unidad de producción está asociada un objeto que implementa la imagen de este recurso, gestionada por su correspondiente agente. Estas clases se relacionan la clase que lleva a cabo la supervisión y control bajo diversos escenarios. En la figura 6.8 se puede observar una secuencia de eventos sobre como opera la supervisión de un recurso en particular, y tiene el siguiente orden: 1) un detector de eventos recibe datos periódicamente de un controlador, el cual a su vez los obtiene del proceso físico, el cual es resultado de una configuración física. 2) Cuando este recurso es supervisado, el detector de eventos indica al supervisor la ocurrencia de un evento. 3) El supervisor produce una acción que ejecuta el actuador, 4) notificando este un cambio en la configuración de la unidad de producción, 5) una vez confirmado el cambio en la configuración, el actuador informa al supervisor sobre el cambio efectuado, y 6) el Supervisor informa el nuevo estado en que se encuentra el recurso. Dentro de un sistema holónico, los holones recurso y los de orden interactúan produciendo así el proceso, cuyo estado se está supervisando. De manera que esta dinámica presentada no puede verse aislada del contexto de producción. Esta secuencia de eventos se puede representar por medio de un sistema de eventos discretos, cuya dinámica la se representa utilizando algunas estructuras matemáticas y formales.

## 6.4. El estado del proceso

El proceso de una orden de producción resulta de la relación entre los holones Orden y Recurso, donde el primero aplica un método de producción que requiere que reserve algunos recursos para poder ejecutar este método. La *clase EstadoProceso* indica entonces en que etapa y tiempo de ejecución se encuentra el método de producción, y proporciona una medida del producto que se ha obtenido, bien sea en términos de

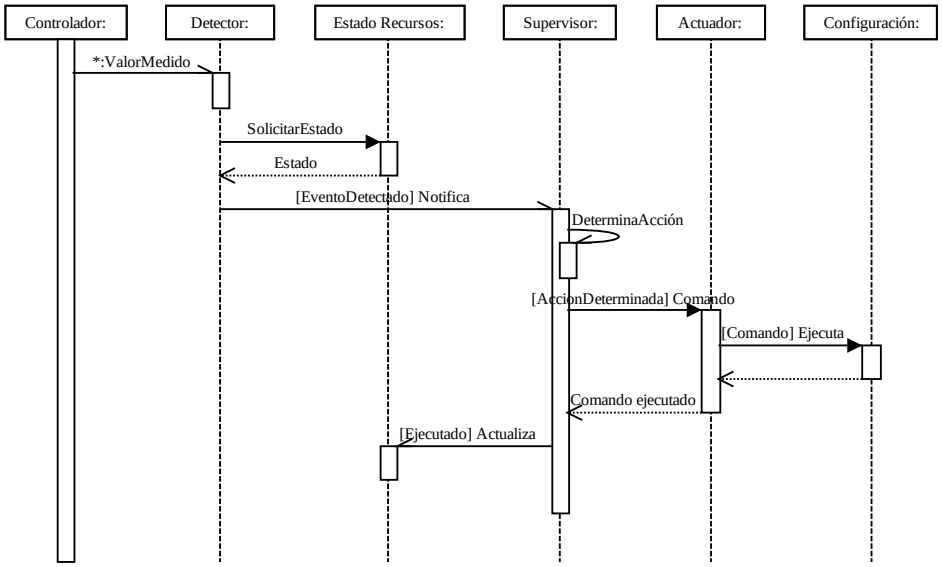


Figura 6.8: Secuencia de actividades dentro de un Holón Recurso

unidades absolutas o en términos de flujo por unidad de tiempo. El comportamiento dinámico de esta clase se describe por medio de la clase *DES*, sobre la cual se profundizará más adelante, y un método que constantemente se va a aplicar es el de *actualizar()*, donde se va a registrar la evolución sobre como se ejecuta la orden de producción. Para extender el potencial de representación de los sistemas a eventos discretos, se hace explícito el registro del tiempo como una variable o estructura de datos que indique en qué punto del tiempo ocurre un cambio de estado, a fin de llevar un registro de los eventos ocurridos.

La figura 6.9 ilustra como se relacionan las clases antes mencionadas con las clases que ejecutan la supervisión y el control de la unidad de producción: el supervisor, detector de eventos, sensor y actuador. Para poder monitorear el proceso, el sensor periódicamente está midiendo las variables del proceso productivo, y las notifica al detector de eventos, el cual establece la ocurrencia de algún evento que informará al supervisor. Este supervisor genera una orden de producción nueva, que en el caso de los sistemas continuos, implica aplicar nuevas leyes de control al proceso que ya se

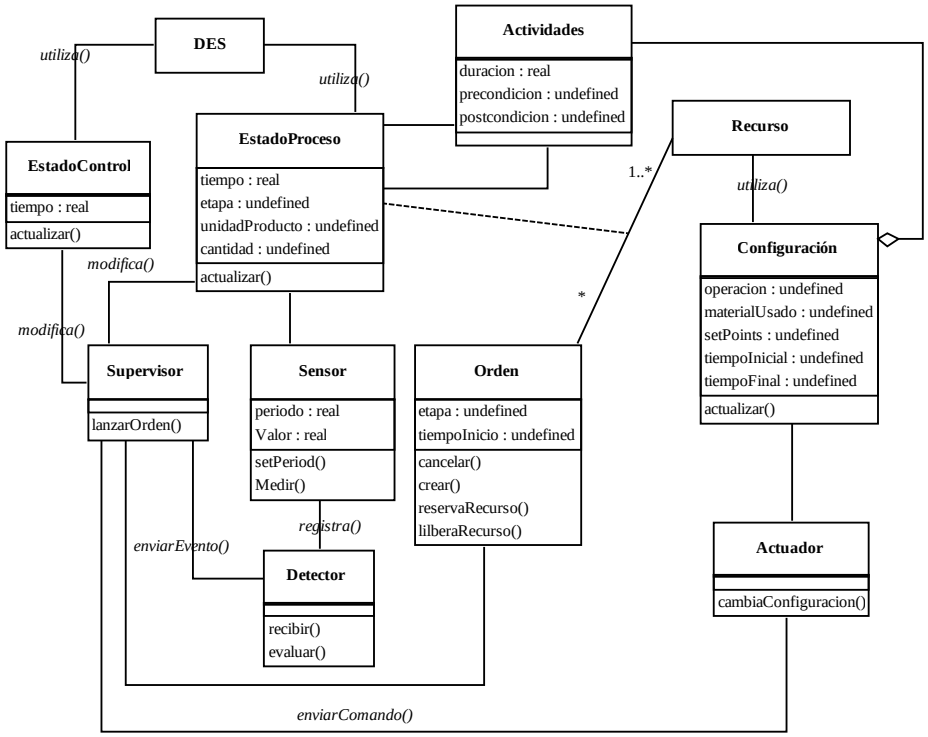


Figura 6.9: Clases utilizadas para describir el Estado del Proceso

venía ejecutando. Esta orden de producción nueva obviamente cancela la orden que se venía ejecutando hasta ahora. El supervisor cambia así la configuración del sistema por medio del actuador. La información asociada a las actividades cuya ejecución se está monitoreando se maneja por medio de la *clase actividades*, donde a cada tarea individual se le asocia un tiempo de duración (en caso de que sea predeterminada), precondiciones para ejecutar la tarea, y cuales tareas son habilitadas luego de su ejecución, es decir, las postcondiciones.

## 6.5. Modelado del mecanismo de supervisión

La configuración de la planta resulta de la relación entre el Método de Producción, el Holón orden, (Objetivo de producción para un producto dado) y la información

asociada a los recursos humanos, equipos, insumos y del avance del objetivo. El estado de cada uno de los componentes está enmarcado dentro del conjunto de modelos que describen el comportamiento de cada uno de los componentes que conforman la UPA guiada por el modelo PROSA. El comportamiento global resulta de la composición de la dinámica de los componentes particulares, y puede ser descrita como la composición de un DES, el cual se describió en la sección tres, en este documento, de la cual se vuelve a escribir la siguiente ecuación:

$$\Phi_k(\Phi_{k-1}(\ldots\Phi_1(\Phi_0(X_R, X_P, t_0, t_1))\ldots)) \tag{6.1}$$

Donde $X_R$ y $X_P$ son los estados de los recursos y del proceso de producción, respectivamente. La dinámica del control supervisorio determina cual es la función $\Phi_i$ a aplicar en el intervalo $(t_{i-1}, t_i)$, hasta que la evolución del sistema indique que el supervisor deba aplicar otra configuración que corresponda a la función $\Phi_j$, o que el sistema dinámico evolucione hasta llegar a ese estado. La aplicación de una configuración se asocia a la *generación de una orden de producción* por parte del supervisor, por lo tanto, cada vez que sea necesario el supervisor va a interactuar con los diversos holones de la unidad de producción, con el fin de introducir los cambios en la configuración que sean necesarios para cumplir con una meta de producción. La secuencia de esta interacción se ilustra en la figura 6.10, donde se observa el flujo de mensajes que intercambian cada una de estas clases. Vale la pena destacar que este proceso es cíclico, es decir, siempre se está monitoreando periódicamente la evolución del proceso, con el fin de asegurar la correcta supervisión de la orden de producción planeada.

Como se muestra en la figura 6.10, el proceso inicia cuando el *Supervisor* cancela la orden actual y en su lugar genera una nueva orden (tal y como funciona en los sistemas continuos de un solo producto), para lo cual interactúa con el *Holón Orden*,

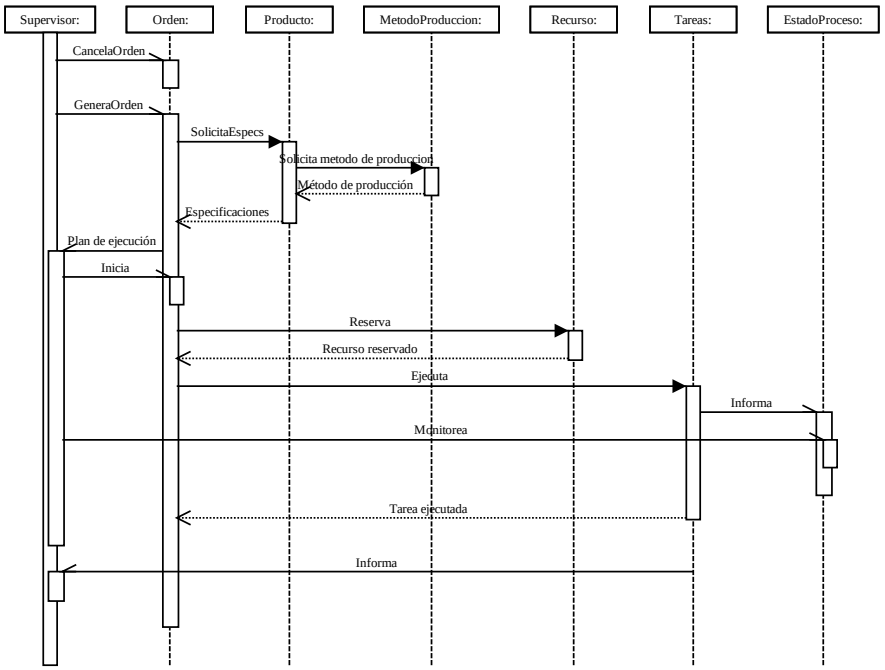


Figura 6.10: Diagrama de secuencia de las interacciones entre supervisor, holones y clases asociadas

informando sobre las nuevas necesidades de producción. Este holón solicita al *Holón Producto* sobre las especificaciones del producto a producir, y el holón producto a su vez indica el método de producción que se va a aplicar. Con esta información, el holón Orden informa al supervisor sobre cual es el plan de ejecución del proceso. Una vez el Supervisor inicia la ejecución del proceso, el holón Orden reserva los *Recursos* necesarios para ejecutar el método de producción seleccionado, y empieza a ejecutar tarea por tarea. El Supervisor monitorea el proceso y está actualizando el estado de la ejecución del proceso, hasta que sea necesario generar otra orden.

## 6.6. Modelado Conceptual de los Sistemas a Eventos Discretos (DES)

En las secciones tres y cuatro se había planteado que para ejercer acciones de control sobre un proceso industrial hay que representar su dinámica y sintetizar un supervisor para asegurar que el proceso a controlar exhiba una conducta deseada, y se habían planteado varios enfoques para ello, destacándose los Sistemas a Eventos Discretos (DES), donde cada estado corresponde a un modo de operación, mientras que los eventos se consideran como el cambio de un estado a otro. Esta dinámica se relaciona con la evolución discreta de los modos de operación (configuración) de la UPA, por lo que es necesario acoplar de alguna manera el DES con el comportamiento de estas variables. Si se etiqueta cada evento con un símbolo, el comportamiento de un proceso industrial se puede resumir por medio de una secuencia de símbolos, donde cada uno de ellos representan las transiciones que han ocurrido. Así, un comportamiento tiene la siguiente forma, $C = \sigma_0, \sigma_1, \ldots, \sigma_n$, donde $\sigma_i$ es el i-ésimo evento ocurrido. Si se consideran los eventos en el tiempo, la traza del comportamiento se puede expresar así: $Ct = \sigma_0(t_0), \sigma_1(t_1), \ldots, \sigma_n(t_n)$. Donde $\sigma_i(t_i)$ indica que el i-ésimo evento fue detectado en el tiempo $t_i$. Ahora se trata de llevar estos formalismos a un diseño orientado a objetos, para diseñar su posterior implementación en la arquitectura computacional de automatización.

Una manera de modelar los Sistemas a Eventos Discretos como un sistema orientado a objetos se muestra en el diagrama de clases de la figura 6.11, donde la clase principal *DES* se asocia a dos clases: por un lado está la clase que describe a los Autómatas de Estado Finito (AEF, de aquí en adelante), y por otro lado está la clase que describe a las redes de Petri, la cual es la *clase PetriNet*. Cada una de estas clases está diseñada para cumplir con la especificación formal de un autómata y una red de Petri, como se describirá más adelante.

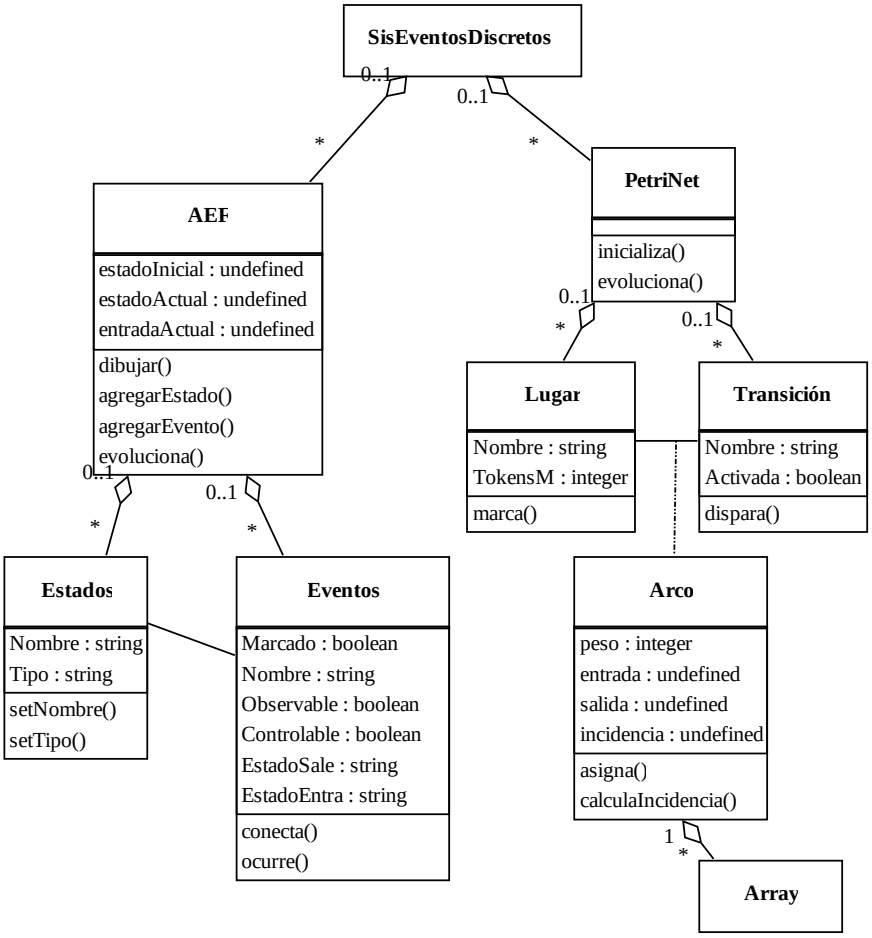


Figura 6.11: Diagrama de clases que implementa a un sistema de eventos discretos

La relación entre una Red de Petri y un Autómata de Estado Finito está en una herramienta matemática para analizar redes de Petri: el *Arbol de alcance*. El árbol de alcance es un tipo particular de Autómata de Estado Finito, que muestra las posibles transiciones que puedan ocurrir, así como los posibles estados en que pueda evolucionar una red de Petri. Por lo tanto es necesario considerar a un DES compuesto tanto por redes de Petri como Autómatas.

La clase AEF tiene como atributos el estado inicial, estado actual, el evento entrante actual y la matriz de incidencia, por medio de los cuales implementa la operación *evolución()*, la cual actualiza a nuevos estados dependiendo del evento introducido. Los estados y eventos discretos conforman dos subclases del mismo nombre, y asociados a la clase principal AEF. Los eventos tienen otros atributos adicionales que permitirán su posterior control: unos indicadores de que si son observables directamente, y a su vez si son controlables.

Para su implementación computacional, la clase PetriNet incorpora dos subclases: *Lugar y Transición* que reflejan al conjunto de lugares y de transiciones respectivamente. Estas dos sub-clases están asociadas por medio de la clase Arco, la cual relaciona lugares con transiciones, e implementa la función de asignación de pesos. Esta clase puede agrupar los pesos dentro de una matriz para así obtener la matriz de incidencia, que permite que la marcación de la red de Petri evolucione. Por lo tanto, para describir a una red de Petri se requiere de una representación matricial, y las operaciones que permiten hallar el arbol de alcance y luego describir la evolución de la red también se implementan como operaciones entre matrices con los vectores de marcación. Por esta razón la clase *Arco* está asociada a un vector de números enteros que conforman la clase *Array*.

## 6.7. Aspectos de implementación computacional

Una arquitectura holónica para supervisar y coordinar procesos continuos la proponen Altamiranda *et al* [3], donde proyectan desde el Sistema de Transformación (ST) a un sitio de supervisión central, donde se encuentran los objetos que reflejan el comportamiento de la unidad de producción, los servidores de reglas de producción, de aplicaciones y el servidor de interfaz Humano-Máquina (HMI). Las funciones que desarrolla este sitio central de supervisión son las de monitorear los eventos recibidos desde supervisores locales, toma de decisiones basado en la dinámica compuesta de los holones, transformar esas decisiones en metas para los sistemas de producción, reconfiguración de grupos y secuenciamiento de operaciones para alcanzar un estado final.

La propuesta establecida en esta tesis para la implantación de un sistema holónico supervisado se puede apreciar en la figura 6.12. Allí se puede observar que el sistema de transformación donde están los recursos físicos y el proceso industrial, es monitoreado por sensores los cuales pasan los valores de las mediciones a un sistema de información que contiene una base de datos de todas las mediciones. A esta base de datos están accediendo periódicamente los agentes detectores de eventos, para comunicar de la ocurrencia de eventos a los agentes que ejecutan las funciones de supervisión y control, los cuales actualizan la imagen del proceso físico que se está controlando, así como la imagen de todo el sistema productivo, representado en la arquitectura orientada a objetos la cual se ha descrito en el presente capítulo. Estos agentes se comunican con los gestionadores de recursos físicos y los actuadores para que ejecuten comandos. Los gestionadores de holones a su vez producen mensajes a los objetos que corresponden a la que corresponde a la configuración estática de la unidad de producción, tal y como se muestra en la parte superior del diagrama de clases.

Esta configuración de la unidad de producción también proporciona la estructura de

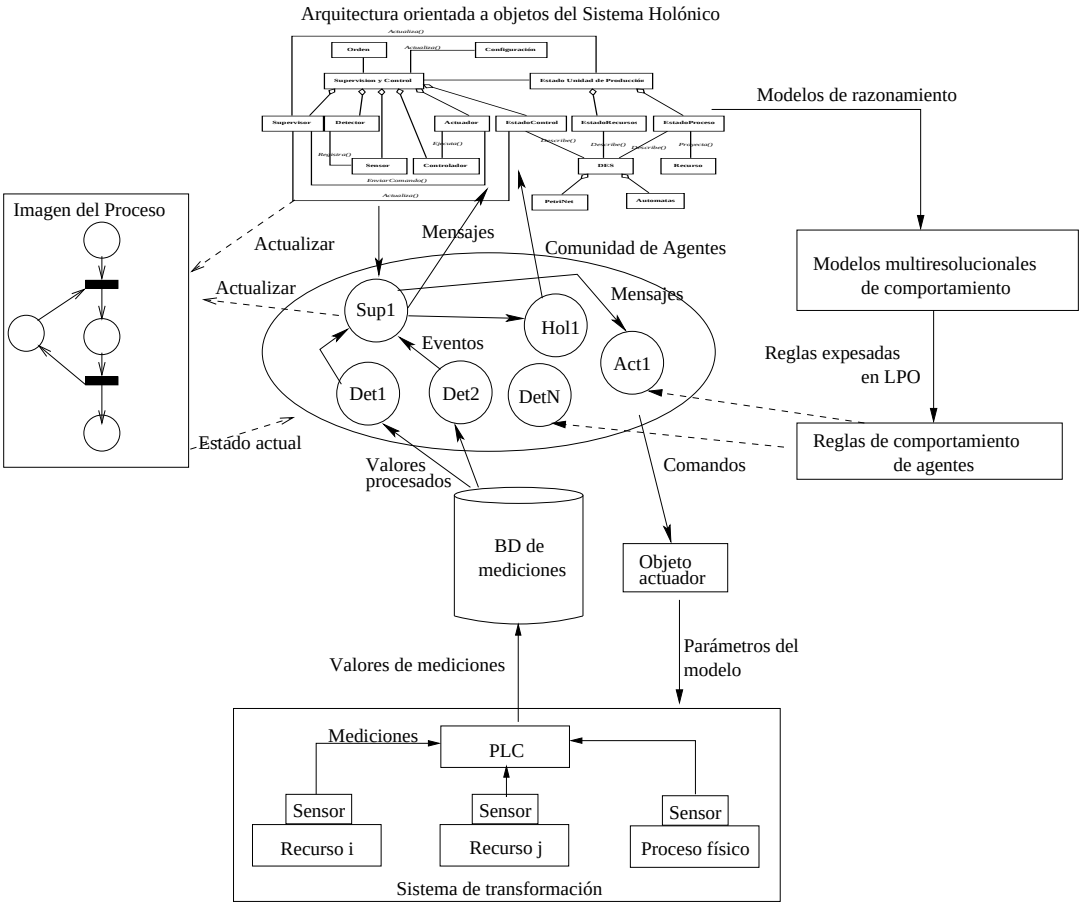


Figura 6.12: Esquema de implementación de la supervisión de un sistema de producción holónico

datos necesaria para almacenar los modelos multiresolucionales del comportamiento de los agentes, esencial para que estos agentes puedan decidir acciones una vez que observen determinados eventos. Para llevar a cabo las funciones de supervisión, es necesario que algunos de estos agentes pueda comunicarse de alguna forma con los dispositivos físicos a través de un PLC u otro tipo de controlador electrónico. De esta manera, pueden incidir sobre el proceso físico y llevarlo a buen término.

## 6.8. Conclusión

En este capítulo se propuso un esquema de diseño orientado a objetos para implementar el control supervisorio en sistemas holónicos basados en la arquitectura de

referencia PROSA, con un doble propósito: por un lado establecer una especificación inicial sobre como sería una implantación en un entorno real, y por otro lado proporciona las bases para construir un simulador de sistemas holónicos. Dentro de este esquema también se especifica la manera en que se van a incorporar los agentes, los cuales tienen una funcionalidad definida, y algunos de ellos obtendrían sus reglas de conducta a partir de los Sistemas a Eventos Discretos que describen la imagen de la Unidad de Producción. Este esquema de razonamiento se aplica en particular a los agentes que tienen la función de detectar eventos.

A pesar de tener claro un esquema de implementación del control supervisorio en sistemas de manufactura holónica, se requiere establecer la validez de este diseño. Una manera económica en cuanto a tiempo y recursos consiste en recurrir a la generación de mediciones simuladas obtenidas del modelo matemático del proceso físico, para observar la interacción de las interfaces implementadas como agentes con la representación de la imagen del proceso, y la traza de los cambios de la configuración de la imagen del sistema, y los cambios de los valores de los parámetros asociados al modelo matemático y así generar otro tipo de conducta. En el siguiente capítulo se describirán los principios que guían a esta metodología de modelado y simulación para validar el diseño que se acaba de presentar.

# Capítulo 7

# Modelado y Simulación para validar el control supervisorio en Unidades de Producción basadas en la arquitectura PROSA

En las secciones anteriores se ha definido la conducta y el mecanismo de supervisión de una Unidad de Producción Holónica inspirada en la arquitectura de automatización PROSA. Posteriormente se estableció su implementación utilizando tecnología de agentes, y se obtuvo como resultado del análisis orientado a objetos un conjunto de clases que representan al sistema holónico y su dinámica. Ahora es necesario evaluar la validez de este diseño, examinando su desempeño bajo las condiciones de un proceso real. Una forma de hacer esta evaluación consiste en desarrollar una implementación directa dentro de un entorno de manufactura continua real, pero esto conlleva problemas asociados al costo y al tiempo de desarrollo, ya que implica tener disponible todo el equipamiento necesario para ejecutar las labores de manufactura. Por lo tanto se requiere de otra forma de validar el control supervisorio de la unidad de producción

holónica. La forma más viable consiste en simular un modelo que contenga dentro de sus componentes a los objetos de una Unidad de Producción concebida bajo la arquitectura de referencia PROSA, junto con las leyes del proceso de producción y las especificaciones lógicas de la conducta de los agentes que van a gestionar cada holón. Por medio de la simulación, se obtiene el beneficio de que se puede repetir muchas veces y a bajo costo, además de abordar diferentes escenarios industriales.

Un esquema de esta validación propuesta se puede observar en la figura 7.1, donde el lado izquierdo de la gráfica muestra un esquema de supervisión de un proceso industrial a ser controlado, donde un Supervisor interactúa con dicho proceso por medio de un Detector de eventos y un Actuador. Este esquema se proyecta computacionalmente en el lado derecho de la gráfica, donde esta proyección resulta en un Simulador discreto y continuo que representa a las condiciones de operación y el modelo físico del proceso industrial, respectivamente; las mediciones que el sensor le transmite al Detector de eventos se simulan numéricamente; el Supervisor se representa por medio de un modelo de su conducta basado en sistemas a eventos discretos, mientras que el Detector y el Actuador se representan por sus respectivas especificaciones de comportamiento basadas en lógica. En este capítulo se describirá una metodología para validar el comportamiento de un proceso supervisado, acoplando sistemas de eventos discretos, sistemas de variable continua, sistemas multi-agente y plataformas de simulación.

Aparte de la economía de tiempo y recursos computacionales, hay otros motivos para recurrir al modelado y simulación como herramienta para analizar y verificar sistemas automatizados de manufactura, en particular aquellos sistemas híbridos para los cuales aún no se dispone de métodos analíticos, tal como lo afirman Lygeros, Godbole y Sastry [68]. Estos métodos analíticos sí existen para analizar sistemas a eventos discretos que representan las acciones del supervisor sobre un proceso, como aquellos basados en autómatas y redes de Petri, con lo son los árboles de alcance. Por tal mo-

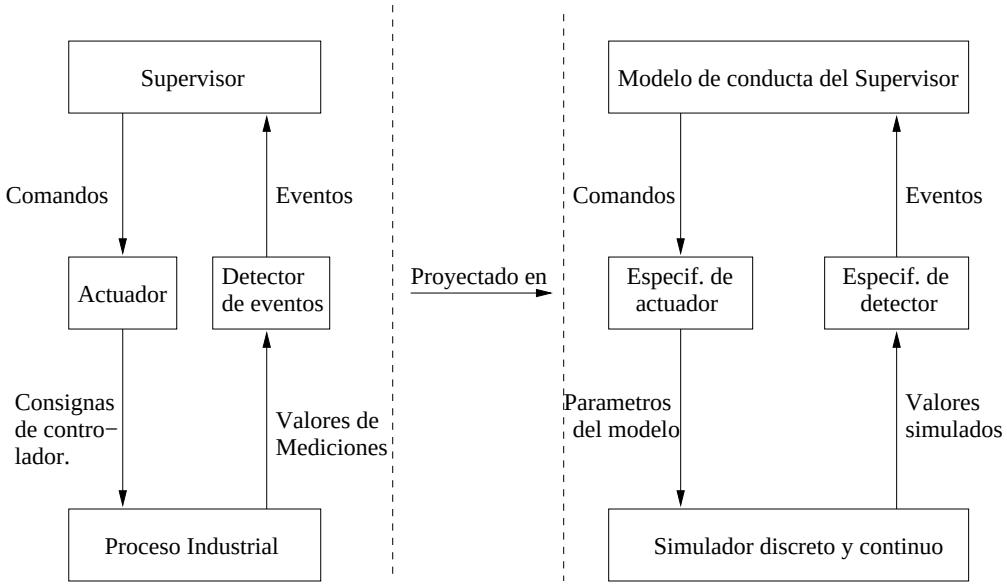


Figura 7.1: Proyección del sistema de control supervisorio

tivo, es conveniente recurrir a la simulación como herramienta para diseñar sistemas de control supervisorio sobre procesos continuos e híbridos, tal como lo sugieren también los trabajos de Zeigler *et al* [114]. Las siguientes definiciones permitirán aclarar algunos conceptos los cuales se van a utilizar con frecuencia en este capítulo.

**Definición 7.1. Modelado** Existen varias definiciones de modelado, a continuación se citan algunas de ellas. Zeigler *et al* [113] define Modelado como *el proceso de organizar el conocimiento que se tiene sobre un sistema*, estableciendo que el modelo es una de las entidades que define el marco de trabajo de modelado y simulación. Guasch *et al* [51] considera al modelado como el proceso de abstracción para describir las características de interés de un sistema. En general, el modelado representa de forma organizada una idea sobre un sistema, de una manera abstracta y utilizando técnicas matemáticas, gráficas u otras que permitan entender un concepto determinado.

**Definición 7.2. Simulación** Banks *et al* [4] definen que la Simulación es la imitación de la operación de un proceso o sistema real. Según Zeigler, Simulación significa la ejecución de un modelo matemático para así generar una conducta de salida se-

mejante al comportamiento de las variables del sistema en el tiempo [113], Fishwick [40], por su parte, propone que la Simulación es un proceso compuesto de tres fases iterativas: diseño de modelo, ejecución del modelo y análisis de la ejecución.

**Definición 7.3. Simulación a eventos discretos** Law y Kelton [66] establecen que la simulación a eventos discretos está relacionada con un sistema que evoluciona en el tiempo en una representación en la cual las variables de estado cambian instantáneamente en puntos separados en la escala del tiempo. Estos puntos en el tiempo son la ocurrencia de eventos.

Aplicando las definiciones anteriores sobre sistema, modelado y simulación a un sistema holónico supervisado de producción continua, se puede describir el funcionamiento de la Unidad de Producción basada en la arquitectura de referencia PROSA como un *conjunto de componentes que van a estar intercambiando mensajes* - tal cual como lo hacen los objetos en el paradigma de programación orientada a objetos -, y sobre este conjunto de componentes se va a ejecutar un algoritmo de simulación que genere una conducta discreta y continua de acuerdo a unas condiciones iniciales definidas.

Dentro de este capítulo primero se describirán los formalismos que proporciona la Teoría de Modelado y Simulación, así como la simulación de sistemas multi-agente. Luego se planteará un método para validar mediante simulación un sistema industrial interactuando con agentes que tienen funciones específicas de gestión, supervisión y control. Luego se presentan detalles de implementación, generando así el comportamiento de la dinámica discreta y continua del proceso a supervisar, planteado en el capítulo tres de esta tesis, y apoyado por algunos ejemplos.

## 7.1. Formalismos de modelado y simulación

Existen varios formalismos para especificar diversos tipos de sistemas, los cuales se condensan en una teoría sobre modelado y simulación propuesta por Zeigler *et al* [113]. Estos formalismos describen sistemas continuos con ecuaciones diferenciales (DESS), sistemas en tiempo discreto (DTSS), sistemas a eventos discretos (DEVS) y los sistemas que acoplan eventos discretos con sistemas continuos (DEV-DESS). La gran ventaja que se tiene al considerar estos formalismos es que aparte de ayudar a entender la Teoría de Modelado y Simulación, separan la especificación de sistemas de la implementación del mismo, lo que libera al diseñador del modelo de las influencias de las particularidades de una plataforma computacional a la hora de implementar.

La tabla 7.1 resume los formalismos más utilizados en modelado y simulación. De aquí en adelante, se extenderá la descripción del formalismo a Eventos Discretos (DEVS), el cual es la base para las simulaciones efectuadas que describen el comportamiento de una Unidad de Producción.

Tabla 7.1: Formalismos de Modelado y Simulación [113]

| Formalismo | Sistema | Modelo típico | Simulador |
|---|---|---|---|
| DESS | Continuo | $q'(t) = aq(t) + bx(t)$ | Integrador numérico |
| DTSS | Discreto | $q(k+1) = a * q(k) + b * x(k)$ | Algoritmo recursivo |
| DEVS | Discreto | Tiempo y Estado en el siguiente evento | Procesador de listas de eventos |

### 7.1.1. Especificación de Sistemas de Eventos Discretos (DEVS)

Este formalismo describe a los sistemas guiados por eventos, tanto eventos externos que origina el entorno, como eventos internos generados por los elementos que componen al sistema. DEVS se concentra en el tiempo restante al próximo evento, y es simulado por un procesador de eventos, el cual engloba aquellos sistemas descritos por Redes de Petri, GSMP, autómatas, algebra min-max, entre otros. El formalismo

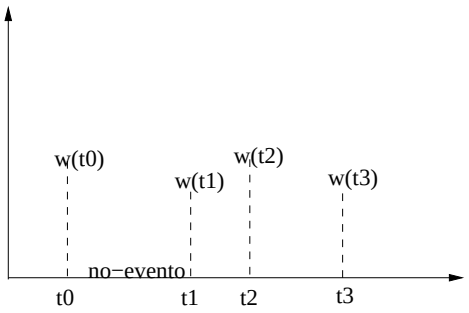


Figura 7.2: Ejemplo de un segmento de eventos

DEVS permite simular sistemas especificados bajo DTSS, y proporciona la base para sistemas de control a eventos discretos. Dado que pueden existir variables continuas en un sistema dado, entre un evento y otro hay se supone que hay un *segmento* de números reales, razón por la cual es necesario definir lo que es un segmento de eventos.

**Definición 7.4. Segmento de eventos** Tenemos $w : [t_0, t_n]$ que es un intervalo definido en $\Re$. Para los puntos $t_0, t_1, \ldots, t_n$ el valor $w(t_i)$ pertenece a un conjunto cualquiera $A$, y es $\oslash$ (no hay evento) para $w(t)$ cuando $t$ es otro punto. La figura 7.2 muestra un ejemplo de un segmento de eventos, y las zonas de no-evento. Así, $t_0$ y $t_1$ son eventos, y el intervalo comprendido entre esos puntos corresponde a una zona de no-evento ($\oslash$).

Puesto que un sistema se puede concebir como un conjunto de componentes de diversa naturaleza (discreta o continua) intercambiando mensajes, es necesario describir matemáticamente a un componente DEVS atómico, para luego describir la interacción de varios componentes de diverso tipo. De acuerdo a la definición de Zeigler, un componente DEVS atómico es una estructura que tiene la siguiente forma:

$$M = (X_M, S, Y_M, \delta_{int}, \delta_{ext}, \lambda, ta) \tag{7.1}$$

Donde:

$X_M$ es el conjunto de eventos de entrada

$S$ es el conjunto de estados

$Y_M$ es el conjunto de eventos de salida

$\delta_{int} : S \rightarrow S$ es la función de transición interna

$\delta_{ext} : Q \times X \rightarrow S$ es la función de transición externa, donde $Q$ es el conjunto total de estados que se define como $Q = (s, e)|s \in S,\ 0 \leq e \leq ta(s)$,

$e$ es el tiempo transcurrido desde la última transición

$\lambda : S \rightarrow Y$ es la función de salida

$ta : S \rightarrow \Re^+ + \{0\}$ es el conjunto de los reales positivos incluyendo al cero, que representa la duración hasta el siguiente evento

### 7.1.2. Relación entre DEVS atómico y la definición de Sistema Dinámico

De acuerdo a la definición de sistema dinámico que propone Fishwick [40], mediante la cual se describe un proceso industrial, mencionado en el capítulo tres de este documento, procedemos a relacionar un sistema dinámico con una estructura DEVS. La relación se establece de la siguiente manera, donde los elementos del sistema dinámico tienen el subíndice "D" para facilitar su identificación:

- La base temporal $T$ es el conjunto de los números reales $\Re$
- $X_D = X_M \cup \{\oslash\}$ es decir, el conjunto de entrada del sistema dinámico es el conjunto de entrada de la especificación DEVS mas el símbolo $\oslash$ que significa no-evento.
- $Y_D = Y_M \cup \{\oslash\}$ igual para los conjuntos de salida.
- $Q = \{(s, e)|s \in S,\ 0 \leq e \leq ta(s)\}$ indica que el conjunto de estados del sistema dinámico corresponde a la pareja del estado especificado en DEVS con

un número real $e$ que nos indica el tiempo transcurrido desde el último evento.

- Los eventos de entrada admisibles corresponde al conjunto de todos los segmentos especificados en DEVS sobre $X$ y $T$.

- Para cualquier segmento de entrada $w :< t_1, t_2 > \rightarrow X$ y estado $q = (s, e)$ en el tiempo $t_1$ la función de transición de estados $\Delta$ se define así:

  $\Delta(q, w < t_1, t_2 >) = (s, e + t_2)$ si $e + t_2 < ta(s)$ y $w$ es el segmento $\oslash$ (no hay eventos)

  $\Delta(q, w < t_1, t_2 >) = \Delta((\delta_{int}(s), 0), w < t_1 + ta(s) - e, t_2 >)$ ocurrió un evento interno o programado

- La función de salida $\Lambda$ del sistema dinámico se define así: $\Lambda((s, e), x) = \lambda(s)$

- Para un segmento de entrada $w$ que no contiene eventos, y no ocurre un evento interno en el intervalo $< t_1, t_2 >$ entonces el estado permanece sin cambios, pero se actualiza el tiempo transcurrido.

- Si un *evento interno* programado por la función de avance de tiempo $ta()$ ocurre en el intervalo $< t_1, t_2 >$ y no hay evento de entrada antes de ese evento interno, entonces la función de transición interna define un nuevo estado, al tiempo transcurrido $e$ se le asigna el valor 0, y el resto del segmento de entrada se actualiza a este nuevo estado.

- Si el segmento de entrada $w$ define un evento externo antes de que ocurra un evento interno, entonces la función de transición externa es la que se aplica y al tiempo transcurrido $e$ se le asigna el valor 0. El resto del segmento de entrada se aplica a este nuevo estado.

### 7.1.3. Esquema de simulación DEVS acoplado

Hasta ahora se ha descrito la estructura de un componente DEVS atómico, pero un sistema de mayor complejidad puede constar de varios componentes DEVS acoplados, incluso con componentes descritos con formalismos DESS o DTSS. De esta manera se pueden representar procesos más complejos. Estos modelos que se componen de componentes acoplados se formalizan de la siguiente manera, con la estructura $DN$.

$$DN =< X, Y, M, EIC, EOC, IC, SELECT > \tag{7.2}$$

Donde:

$X$ es el conjunto de eventos de entrada

$Y$ es el conjunto de eventos de salida

$M$ es el conjunto de componentes DEVS

$EIC \subseteq DN.IN \times M.IN$ es la relación de acoplamiento externo de entradas

$EOC \subseteq M.OUT \times DN.OUT$ es la relación de acoplamiento externo de salidas

$IC$ es la función de acoplamiento interno de entradas

$SELECT : 2^M - \oslash \rightarrow M$ es el selector de componente para el siguiente evento

$M.IN$ y $M.OUT$ son los puertos de entrada y salida de cada componente, respectivamente

$DN.IN$ y $DN.OUT$ son los puertos de entrada y salida del modelo acoplado, respectivamente

La figura 7.3 muestra un esquema de un sistema con componentes acoplados: dos de ellos son DEVS, mientras que hay un componente continuo y otro que funciona

basado en ecuaciones en diferencia. También se pueden apreciar los puertos de cada componente, y los puertos externos del sistema, que coinciden con algunos de los componentes. La descripción formal para acoplar eventos discretos con sistemas continuos se puede encontrar en la teoría propuesta por Zeigler *et al* [113].

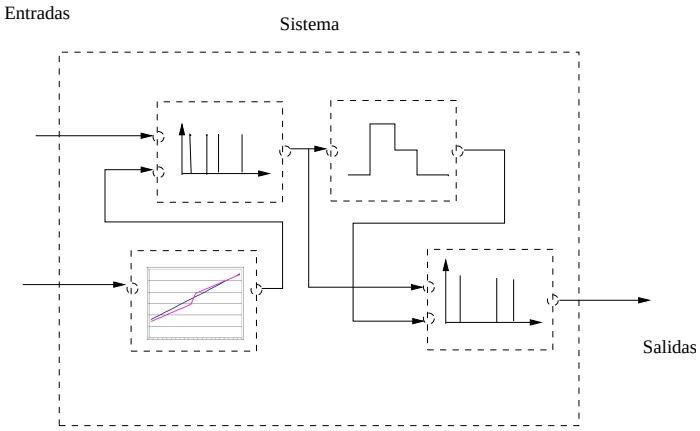


Figura 7.3: Sistema representado con componentes DEVS acoplados

El desempeño del Control Supervisorio de la Unidad de Producción Holónica, cuya dinámica ha sido explicada en las secciones tres y cuatro, puede evaluarse por medio de un modelo de simulación basado en el formalismo DEVS, cuya implementación computacional se hace por medio de una red de componentes que intercambian mensajes, como componentes DEVS, DTSS y DESS acoplados. De esta manera, se puede representar tanto cada recurso físico como el flujo de material y de información que se intercambia, representado por modelos físicos en ecuaciones diferenciales o estocásticas, así como las etapas lógicas de un proceso industrial, las cuales se representan como sistemas a eventos discretos, descrita mediante sistemas a eventos discretos.

Para un componente atómico basado en DEVS, la simulación consiste en actualizar el estado del sistema cada vez que ocurre un evento, producir la salida correspondiente y generar el próximo evento con la función $ta()$ para evento interno, o producir una transición si ocurre un evento externo. El algoritmo general se puede observar en la figura 7.4, donde se aprecia que el algoritmo progresa en pasos discretos de tiempo,

correspondientes a la variable $T_a$, la cual contiene el tiempo que va a transcurrir hasta el evento más próximo. Cuando este componente atómico está acoplado con otros formando un sistema de mayor complejidad, como el que aparece en la figura 7.5 el algoritmo básico se puede describir en los siguientes pasos:

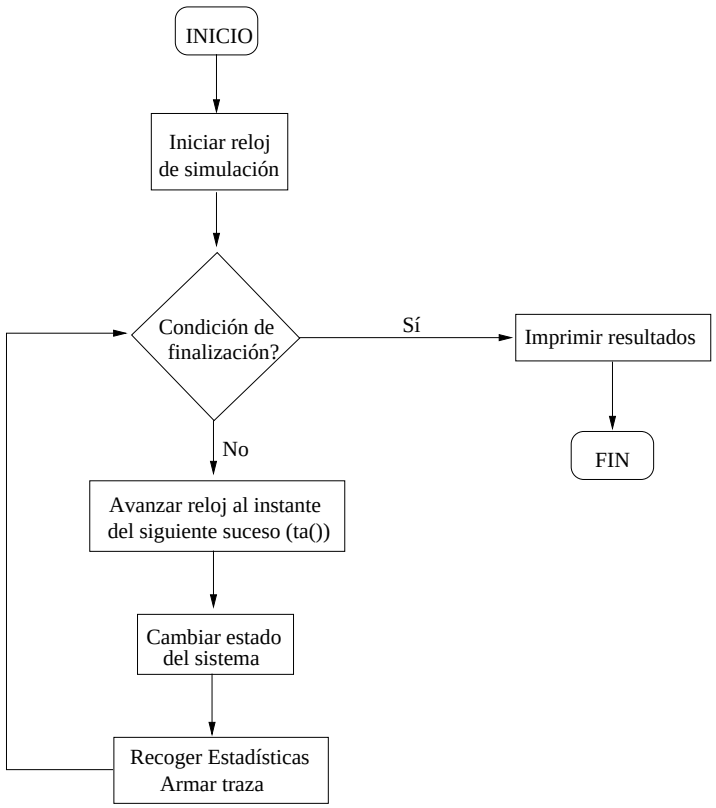


Figura 7.4: Algoritmo general simular sistemas guiados por eventos, propuesta de Ríos *et al* [95]

.

1. Dado un tiempo de simulación $t_a$ buscar en cada componente el tiempo que corresponde al evento próximo a ocurrir $t_i$, seleccionar el componente $i$ cuyo evento sea el más próximo a $t_a$.

2. Avanzar el tiempo de simulación $t_a$ hacia $t_i$ y ejecutar la función de transición interna del componente $i$.

3. Propagar el evento de salida de $i$ hacia los otros componentes que estén relacionados directamente, y ejecutar las transiciones externas correspondientes.

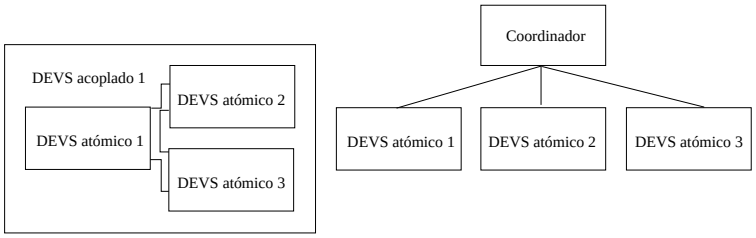


Figura 7.5: Simulación de componentes DEVS acoplados

### 7.1.4. Mecanismos de simulación de sistemas a eventos discretos

Los formalismos de modelado DTSS, DESS, DEVS y DEVS acoplados nos sirven para especificar las características de los simuladores que implementen cada uno de esos formalismos, por medio de plataformas computacionales para simulación. Todo modelo a implementar en un simulador tiene *elementos* y *mecanismos de funcionamiento*. Entre los elementos tenemos las entidades, los atributos, las colas y los eventos. Las entidades se asocian con aquellos mensajes que circulan a través de un sistema, y pueden ser temporales o permanentes, las marcas en una red de Petri son un ejemplo de entidades temporales, mientras que los componentes que representan recursos del sistema son entidades permanentes, como lo es un equipamiento. Las actividades son tareas o acciones que tienen lugar en el sistema. Los eventos pueden ser no condicionados, o sea que ocurren planificadamente, o condicionados, que depende su ocurrencia de que se presenten ciertas condiciones.

Un simulador que funciona bajo el formalismo de eventos discretos puede implementar tres mecanismos diferentes de funcionamiento de acuerdo a Guasch *et al* [51], que se distinguen en la manera como escogen al siguiente evento, con la suposición de que el sistema no cambiará su estado entre un evento y el otro, partiendo del algoritmo DEVS presentado anteriormente. Estos mecanismos se describen a continuación:

- **Programación de eventos**, que consiste en mostrar como evoluciona el mo-

delo a medida que se activan los eventos. No distingue a los eventos condicionados de los no condicionados. El simulador gira en torno a una lista de eventos pendientes, de la cual extrae los eventos procesados e inserta eventos generados. Este mecanismo puede implementarse por medio de hojas de cálculo, o mediante aplicaciones que funcionan como adiciones a hojas de cálculo, como Crystalball.

- **Interacción de procesos**, que consiste en mostrar la evolución de las entidades temporales del modelo. Diferencia las entidades temporales de las permanentes, estas últimas son bloques de alto nivel, gestionados y simulados independientemente. Entre las plataformas de simulación que trabajan de esta manera están ARENA, Automod, GLIDER y GALATEA.

- **Exploración de actividades**, se concentra en actividades, y en las condiciones que permitan que ocurra una actividad, se parece conceptualmente al funcionamiento de una Red de Petri. Usa incremento de tiempo fijo. Visual Object Net es un simulador de Redes de Petri, que funciona bajo esta filosofía, al igual que PetriSim. CPN Tools también funciona con redes de Petri coloreadas,

Para los sistemas continuos, el proceso de simulación es diferente y en este caso el tiempo avanza a través de intervalos de igual longitud y las ecuaciones diferenciales se integran entre los límites inferior y superior de cada intervalo. Para ello es necesario establecer las condiciones iniciales cuando se ejecuta la simulación.

Cuando se simula el comportamiento de un sistema que tiene tanto componentes continuos como componentes que involucran eventos discretos, es importante la sincronización entre los eventos y la evolución continua registrada en cada intervalo. Entre un evento y otro se deben integrar las ecuaciones del sistema continuo, para mantener la inercia de las variables a medida que evoluciona el sistema.

### 7.1.5. Simuladores que implementan DEVS

Se han desarrollado varias implementaciones basadas en el formalismo DEVS para simular sistemas de naturaleza discreta. DEVSJava, PowerDEVS y CD++ [54] son un ejemplo de simuladores orientados a la composición de componentes. Arena [59] es un software orientado a la interacción de procesos y el modelo se construye ensamblando componentes dentro de una red, algunos de los cuales se comportan como generadores de eventos discretos. GLIDER [31] es un lenguaje de simulación desarrollado en la Universidad de Los Andes, donde un sistema se describe por medio de una red de objetos que intercambian mensajes asociados a la ocurrencia de eventos aunque también maneja variables continuas expresadas como ecuaciones diferenciales.

Últimamente se han llevado a cabo intentos para agregar agentes inteligentes a componentes DEVS, para obtener objetos que puedan decidir en tiempo de ejecución sobre como interactuar con los componentes del sistema, inclusive llevando a un cambio de estructuras. AgeDEVS es una implementación sobre DEVS que incluye componentes con un estado mental interno, GALATEA es un lenguaje de simulación desarrollado para implementar una teoría de simulación de sistemas multi-agente dentro de una plataforma de simulación basada en java.

### 7.1.6. Simulación de Sistemas Multi-agente

Además de la implementación del esquema de supervisión de un HMS utilizando agentes, es necesario validar el diseño basado en la especificación de comportamiento de estos agentes interactuando con el simulador a eventos discretos, tal y como se planteó al inicio de esta sección. Con los agentes se puede simular el comportamiento de entidades que *razonan* y colaboran entre sí para lograr ciertas metas, a la vez que influyen sobre el sistema modificando su comportamiento, y a su vez también son afectados por la evolución de la planta. A continuación se muestra una teoría

de simulación para integrar agentes a una plataforma de simulación, lo cual es la especificación de la plataforma de simulación que se va a utilizar para validar la propuesta de control supervisorio en sistemas holónicos.

Una teoría de simulación de sistemas multi-agente basados en lógica, que proporciona el marco referencial adecuado para aplicarla a proyectos de modelado inteligente de sistemas de diversa naturaleza fue propuesta por Dávila, Tucci y Uzcátegui [28]. Esta teoría se basa en una extensión al formalismo de Sistemas Multi-agente, propuesto por Ferber y Muller [37] y también en la especificación lógica de un agente racional, propuesta por Kowalski, la cual se describió en la sección cinco de este documento. Una implementación del mecanismo de razonamiento de agentes basados en programación lógica se pueden consultar en los trabajos de Kowalski, Sadri y Dávila [65, 33], donde se pueden describir reglas para observar, reglas que expresan creencias y reglas de procedimiento, entre otras.

Una adaptación del algoritmo general a eventos discretos para implementar la simulación multi-agente con detección de eventos y modelos multi-resolucionales se muestra en la figura 7.6, donde se propone que a partir de la detección de eventos se actualiza un sistema a eventos discretos, lo cual hará factible que otro agente decida acciones a partir de la nueva representación del estado del sistema. La detección de eventos tiene lugar principalmente sobre variables continuas, para las cuales se simula su medición. Este algoritmo es una adaptación hecha a partir del algoritmo general de simulación guiada por eventos discretos, como lo proponen Ríos *et al* [95]. La diferencia fundamental está en el hecho de que el algoritmo avanza según el intervalo de observación de cada agente, en particular de aquellos agentes que asumen el rol de detector de eventos, y que están observando la evolución de las variables del sistema. Entre cada observación del agente las variables continuas evolucionan según su modelo matemático, y de acuerdo a los valores de estas variables es que cada agente decide la ocurrencia de algún evento, de acuerdo a sus reglas de decisión.

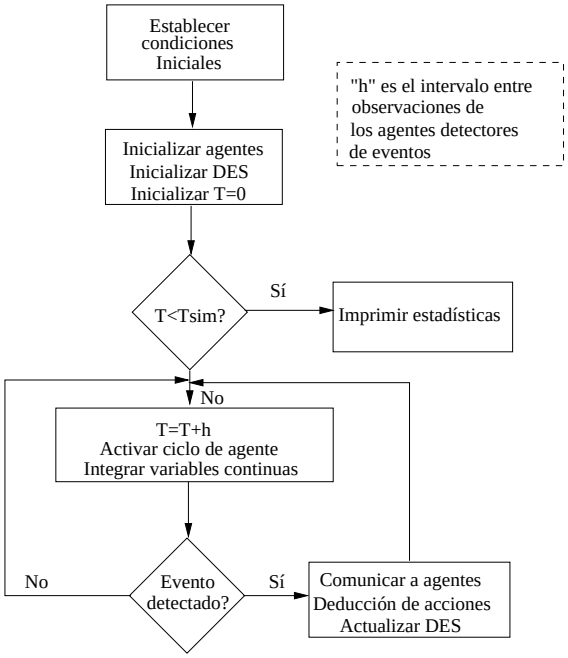


Figura 7.6: Algoritmo general para la simulación multi-agente

En cuanto al Sistema de Eventos Discretos (DES) que describe el estado global del proceso a simular, es actualizado por el agente que implementa al supervisor, el cual previamente ha recibido información de ocurrencia de eventos por parte del agente detector. Así mismo, este agente comunica el nuevo estado a los agentes que están administrando los recursos, productos y órdenes, para que decidan y ejecuten las acciones correspondientes. La simulación termina cuando el tiempo de simulación sobrepasa un tiempo establecido de antemano o cuando se han cumplido otras condiciones de terminación. Obsérvese que este algoritmo simula la secuencia de eventos presentada en las figuras 6.8 y 6.10, del capítulo anterior.

La plataforma de simulación GALATEA (acrónimo de Glider with Autonomous Logic-Based Agents) implementa la interacción de componentes DEVS con agentes racionales, de manera que en un simulador se pueden obtener ambas características. Por esta razón se utilizará esta plataforma de simulación en la validación inicial del diseño propuesto para la supervisión de unidades de producción diseñadas con base en la arquitectura de referencia PROSA para automatización. En la siguiente sección

se planteará el método para simular una unidad de producción supervisada y basada en agentes. Posteriormente se introducirán agentes desarrollados en la plataforma JADE para implementar los gestores de holones; supervisor y detector de eventos, *independientes del modelo a eventos discretos*, para proporcionar un comportamiento aproximado de estos agentes a la hora de interactuar con entornos de producción reales, solo que estos entornos estarían simulados en los sistemas a eventos discretos que describen los diversos casos de estudio.

A continuación se presenta entonces una metodología general para obtener modelos de simulación basados en componentes DEVS, que implementan el Control Supervisorio de Sistemas Holónicos basados en la arquitectura de automatización PROSA y que incorporan agentes inteligentes.

## 7.2. Metodología para generar modelos de simulación de sistemas holónicos supervisados

El objetivo principal de la simulación es reproducir la conducta de un Sistema Holónico supervisado, cuyos holones están concebidos según la arquitectura de referencia PROSA. Se incorporan agentes en la simulación para representar las decisiones de los holones y del mecanismo que implementa el control supervisorio. El Control Supervisorio funciona de la siguiente manera: cercano al piso de planta está el proceso industrial, cuyas variables más importantes se miden continuamente por medio de sensores y controladores, los cuales dejan un registro de las mediciones efectuadas. Estos datos los utiliza el *Detector de eventos* para indicar al *Supervisor* que han ocurrido eventos, y luego el Supervisor toma decisiones para permitir que la *planta* evolucione libremente, de acuerdo a las especificaciones del proceso, actuando únicamente para inhibir conductas no deseadas, produce comandos a los controladores asociados al proceso, para que aseguren que la trayectoria de las variables se mantengan dentro

de un comportamiento ideal. Debido a la dificultad para montar este experimento en un entorno industrial real, es necesaria una validación de la dinámica de su comportamiento utilizando el modelado y la simulación. Por lo tanto se requiere disponer de un modelo matemático que representa al proceso industrial, salidas que simulan los procesos de medición y agentes de software que asuman el rol de supervisor, detector de eventos, gestionadores de holones y actuador se diseñan para llevar a cabo funciones que mantengan la trayectoria de las variables dentro de las especificaciones de diseño iniciales. En esta sección se presentarán algunos aspectos de la metodología propuesta, la cual es aporte de esta tesis doctoral.

### 7.2.1. Trabajos similares desarrollados en esta área

Algunos aspectos que dan origen al método propuesto anteriormente (SPC, Holones basados en PROSA, Sistemas discretos y continuos, Supervisión, Sistemas multi-agente cuyo razonamiento se basa en modelos multi-resolucionales) se pueden encontrar en diversos trabajos que se han publicado sobre el tema de simulación de procesos industriales. Con diferencias dentro del entorno industrial, como el tipo de proceso, o diferente manera de concebir los procesos de producción; o con aspectos diferentes a los mencionados anteriormente. A continuación se resumen los trabajos más importantes publicados recientemente en el área. Leitao [67] describe el comportamiento de un sistema holónico por medio de redes de Petri, evaluado por medio de indicadores tales como throughput (relación de entrada/salida), uso de recursos, entre otros. Varios trabajos de evaluación del desempeño de sistemas de eventos discretos también se han llevado a cabo, uno de ellos consiste en simular sistemas de control híbrido proyectando variables continuas a un espacio de estados discretos, a partir del cual se sintetiza un controlador aplicando una secuencia de entradas de control, como el trabajo de Cembellín [17]. Una implementación DEVS de red de Petri se puede consultar en el trabajo sobre el lenguaje CD++ de Jacques y Wainer [54], a manera de

ejemplo. Una simulación de un sistema dinámico incrustada dentro de un mecanismo de control es la propuesta de Millán y O'Young [74] para sintetizar sistemas supervisores. Por otra parte, Aceves *et al* [1] proponen un lenguaje orientado a procesos continuos, y cuya implementación produce como salida un código de simulación. Una propuesta de evaluar el desempeño de procesos representados por redes de Petri es la de Jiménez *et al* [56], que utiliza el concepto de redes de Petri estocásticas y cadenas de Markov, otro enfoque consiste en proyectar sistemas discretos a sistemas a eventos discretos, como el de Parra y Chacón [89], avance preliminar de este trabajo doctoral. Hsieh [53] utiliza la simulación para conformar holarquías en un sistema holónico, representando a los recursos y productos por medio de redes de Petri. Por otro lado, nuestra propuesta se basa en la interacción conjunta de todos los aspectos para obtener un procedimiento que permita implementar y evaluar computacionalmente un mecanismo de supervisión de unidades de producción bajo el enfoque holónico de automatización. Una aplicación cuyo caso de estudio es un sistema hidroneumático utilizando el modelo multi-resolucional extendido, lógica difusa, agentes y simulación multi-agente la presentan Parra *et al* [87, 86, 88].

### 7.2.2. Derivación del método de modelado y simulación

El procedimiento para validar el comportamiento dinámico de un sistema supervisado se puede apreciar en la figura 7.7, la cual muestra tres niveles conceptuales, donde el nivel central es la aplicación de conceptos y métodos para supervisar una unidad de producción descrita por medio de holones, y es un enlace entre la descripción del sistema de transformación, y el esquema de validación utilizando modelado y simulación. Los elementos que son el punto de partida son dos: uno de ellos es la descripción del proceso industrial a partir del cual se obtiene un *modelo físico*, expresado por medio de un sistema de ecuaciones diferenciales o algebráicas en variable continua (SDVC, de aquí en adelante), acoplada a algunos sistemas de eventos discretos, como autó-

matas o redes de Petri. El otro elemento de partida es la arquitectura de referencia de la organización industrial y los objetivos de producción, los cuales especifican al Supervisor. Este Supervisor y el modelo físico estableceran el modelo de conducta, el cual a su vez establecerá los mecanismos de decisión, detección de eventos y comandos para que el proceso se mantenga dentro de las condiciones deseadas de operación. A continuación se detallarán los pasos necesarios hasta llegar a una solución computacional que implemente la dinámica del proceso industrial supervisado. En el nivel de aplicación de conceptos y métodos, un punto de partida necesario son los modelos multi-resolucionales, que permiten deducir acciones a partir de hechos observados.

### 7.2.3. Definir Regiones de Operación, Estados y Eventos

Este paso permite obtener un Sistema a Eventos Discretos (DES, de aquí en adelante) a partir del modelo físico en variable continua. Para ello se aplica una función de proyección de un vector $\Re^n$ a un valor discreto, de la manera propuesta por Chacón, De Sarrazín y Khodr [20] dentro de un marco para modelar la dinámica de varios subsistemas acoplados. Este vector de valores discretos definiría un intervalo de una o mas variables que contendría una *región de operación*. Esta región de operación a su vez define a un *Estado*, el cual es una variable discreta. Para deducir en que región de operación se encuentra un proceso, se utiliza el modelo linguístico de Mamdani, de una manera similar al mostrado en la figura 3.12, donde las entradas son las lecturas de variables continuas y la salida es un estado discreto. El cambio de un estado a otro, es decir, el cambio de valores de las variables dentro de una región de operación que indique que se encuentran en otra región, implica la ocurrencia de un *Evento*. Un DES se compone de estados y eventos, y puede representarse por medio de una Máquina de Estados Finitos, una red de Petri o por otras estructuras lógico-matemáticas.

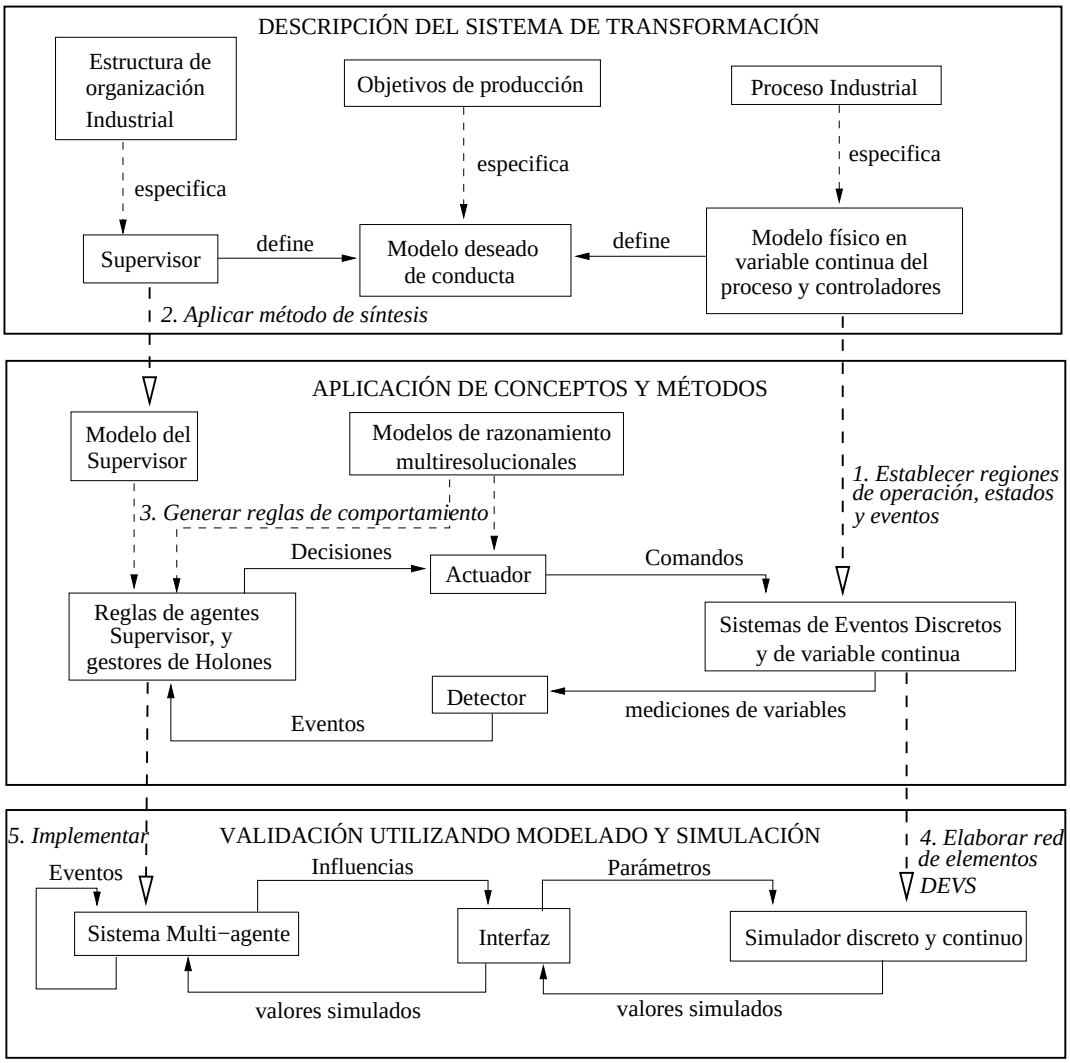


Figura 7.7: Método para validar el control supervisorio de un HMS

### 7.2.4. Aplicar método de Síntesis de Supervisor

Teniendo una especificación del supervisor, en cuanto a mantener una secuencia de eventos deseada, o evitar una secuencia de eventos no deseada, se puede aplicar un método de *Sintesis de Sistemas Supervisores* si además se dispone de la dinámica del sistema descrita por medio de un DES. A este DES se le aplican las reglas de detección de eventos para obtener el *estado del proceso*, mientras que con las reglas de la supervisión aplicadas a este estado del proceso se sintetiza un supervisor, que acoplado al DES a controlar nos permite obtener un *sistema a eventos discreto supervisado*.

Existen varios métodos para sintetizar un sistema supervisor, y depende de la manera en que se haya descrito al DES. Si el DES se ha descrito con un autómata de estado finito, el método estará basado en la teoría de lenguajes formales propuesta por Wonham y Ramadge [93], donde el supervisor obtenido será otro DES acoplado al DES que describe la planta, y cuya función será la de inhibir *eventos controlables*. Si se utiliza una red de Petri la manera más directa de sintetizar un supervisor consiste en aplicar restricciones lineales, introduciendo lugares adicionados denominados "de holgura" como lo proponen Moody y Antsaklis [76] y así ejercer control sobre la ocurrencia de transiciones asociadas a eventos. En caso de que no se cumpla el *criterio de admisibilidad*, se manipularán algebráicamente las restricciones para sintetizar un supervisor que si cumpla con el criterio de admisibilidad.

### 7.2.5. Generar reglas para agentes

Para implementar al supervisor que se haya sintetizado es necesario disponer de la tecnología de información adecuada, por tal motivo se recurre a la tecnología de agentes, ya que su naturaleza distribuída la hace apropiada para su aplicación en entornos de producción complejos, que también están distribuidos sobre la organización industrial. El mecanismo de razonamiento de cada uno estará basado en lógica de

primer orden (LPO, de aquí en adelante) ya que permite especificar el razonamiento de la manera más parecida al lenguaje natural, y establecer comportamientos tanto reactivos como proactivos, de acuerdo a la definición de Kowalski [65], los cuales permiten reaccionar ante cambios en el entorno y planificar sus acciones sobre tales cambios. Kowalski propone que el agente está interactuando constantemente con su entorno, y a partir de los hechos que observa razona y produce influencias (acciones que debe ejecutar) para inducir eventos posteriores dentro de este entorno. El esquema de razonamiento se puede apreciar en la figura 5.2, donde se muestra la distinción entre el razonamiento reactivo y el proactivo, donde el primero aparea observaciones con metas para obtener acciones aplicando razonamiento hacia adelante, y el segundo obtiene acciones a partir de reducir metas a sub-metas, en un proceso más complejo que implica razonamiento hacia adelante y hacia atrás.

Complementar el esquema de representación de reglas de agente con los modelos multi-resolucionales de Sanz es uno de los resultados de este trabajo doctoral, y permite generar reglas de conducta para agentes. La arquitectura de razonamiento para generar estas reglas de conducta, basada en los modelos multi-resolucionales de Sanz [98] es la más apropiada para especificar el funcionamiento de los agentes, puesto que permite obtener un curso de acciones a través de la aplicación de funciones de abstracción y concreción a hechos observados, tanto a nivel cualitativo como su proyección hacia un modelo numérico, que en nuestro trabajo corresponde al modelo físico del proceso de producción. Para proporcionar mayor robustez al proceso de supervisión de procesos, se utiliza el modelo de Sanz extendido, publicado como avance de nuestro trabajo doctoral [88], el cual proporciona además mecanismos para supervisar procesos expuestos a la ocurrencia de fallas, por intermedio del acoplamiento de un estado del autómata global, que indica el estado general del proceso (normal, en falla, reparación, etc) con el autómata que describe la conducta del proceso industrial particular. Estas reglas permiten generar las conductas de los agentes Detector de

Eventos, Supervisor, Actuador, además de los gestores de los Holones de Recurso y Producto.

### 7.2.6. Definir e implementar Red de componentes basados en DEVS

Teniendo un mecanismo de supervisión basado en agentes que observan sistemas discretos, que reflejan una dinámica discreta y continua, es necesario evaluar su validez. Esto se puede lograr de maneras: implementando directamente sobre un entorno real, o simulando este entorno. La última forma permite economizar tiempo, recursos a la vez que se evalúan diversos escenarios, por lo tanto es necesario aplicar un enfoque de simulación orientada a eventos discretos y que nos permita responder preguntas sobre que porcentaje de tiempo el sistema está en un estado determinado, por ejemplo, o cual es el valor de una variable mientras se detecta el evento que cambia de una región de operación a otra. El formalismo de simulación basado en eventos discretos (DEVS, de aquí en adelante) proporciona la base para simular sistemas de control a eventos discretos y permite interactuar con sistemas de variable continua por medio de una red de componentes que intercambian mensajes como se había mencionado anteriormente, con la teoría de Zeigler *et al* [113], con el cual se puede utilizar para encontrar trayectorias de controladores discretos, como se muestra en la figura 7.8.

La implementación de estos simuladores de sistemas de eventos discretos se realiza por medio de una lista de eventos futuros, como proponen Law y Kelton [66] o por la interacción de procesos propuesta por Guasch *et al* [51] es un medio para medir eficiencia del sistema en cuanto al grado de utilización de equipamiento, si hay o no represamiento de insumos, probabilidad de ocurrencia de un evento en un tiempo dado, entre otros aspectos. Estos modelos que representan a sistemas de eventos discretos son dinámicos, estocásticos y discretos, donde las variables de estado cambian

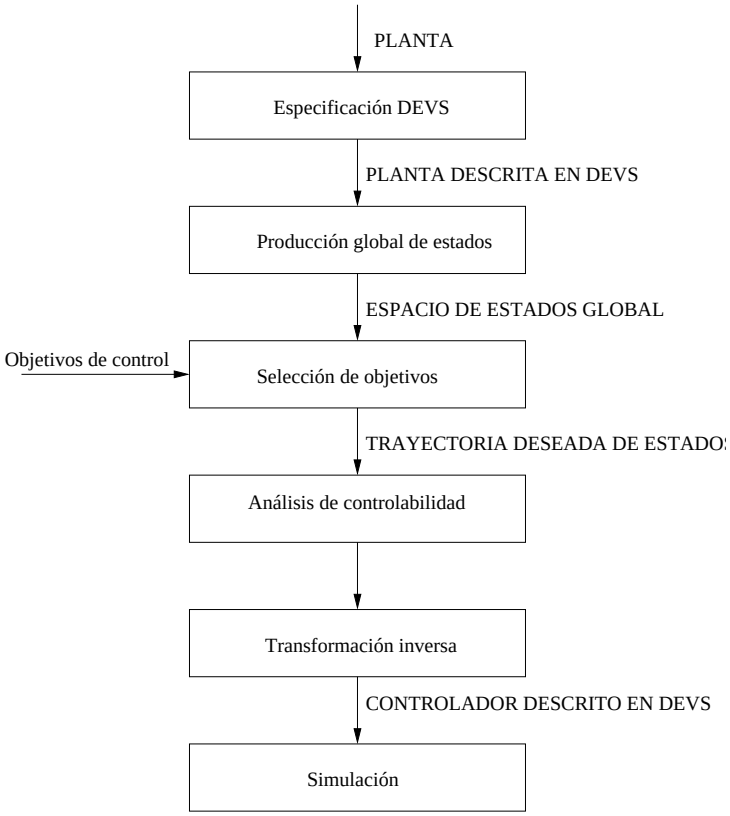


Figura 7.8: Simulación para encontrar trayectorias de controladores

de valor en instantes no predecibles de tiempo. Un método para obtener la red de componentes basados en DEVS se propone en el trabajo de Parra y Chacón [89], el cual se complementa en este trabajo doctoral.

### 7.2.7. Implementar Sistema Multiagente

Para una unidad de producción autónoma cuyo proceso de transformación o distribución es indivisible, se han identificado varios agentes que cumplen funciones diferentes pero complementarias entre sí: detección de eventos, aplicación de reglas de supervisión para inducir nuevos eventos, y ejecución de los mismos, además de los gestores de cada Holón Recurso. Dado que esta unidad de producción se concebimos como un *Holón* y puesto que los sistemas holónicos son autosimiliares, esta estructura de agentes también tenderá a repetirse en otros niveles dentro de un sistema producti-

vo. Estos agentes conforman un sistema multiagente que implementa las capacidades de razonamiento de un holón de producción, e interactarían con un simulador que representa al sistema de eventos discretos por medio de una interfaz. Estos últimos elementos: el sistema multiagente, la interfaz y el simulador DEVS ejecutarían el modelo de proceso industrial supervisado, y permitirían establecer la validez del diseño de la supervisión de procesos.

## 7.3. Implementación del Simulador

El simulador que se va a describir en esta sección corresponde a una implementación de la solución computacional que simula el comportamiento de un sistema holónico supervisado, lo que a su vez es una solución al problema central de este trabajo doctoral, puesto que aplica una metodología para obtener un modelo matemático y lógico que reproduce la conducta de un proceso industrial gobernado por el sistema holónico, de acuerdo a unas metas de producción y unas condiciones de operación determinadas. En la figura 7.7, corresponde a la sección de *validación utilizando modelado y simulación*, la cual contiene el sistema multi-agente que simula la conducta del supervisor, detectores de eventos y gestores de holones, así como las comunicaciones entre estos agentes. Además está el simulador del proceso industrial, que evoluciona de acuerdo a sus leyes físicas y a las acciones de los agentes que modifican sus parámetros de operación. Este simulador se basa en la filosofía DEVS acoplada, donde se combinan componentes DEVS y DESS, el mecanismo de simulación corresponde a la interacción de procesos, donde cada componente corresponde a un *recurso* de la Unidad de Producción, y los mensajes que circulan por la red de componentes corresponden a insumos e información. La lógica del proceso está representada por un sistema a eventos discretos, bien sea una máquina de estados finitos o una red de Petri. A la representación mediante DES del proceso que tiene lugar en la Unidad de produc-

ción se le asocia un componente DEVS a cada estado o lugar, y otro diferente para representar las transiciones.

Como plataforma de implementación se escogió al lenguaje de simulación *galatea*, por las siguientes razones: está basado en el lenguaje de programación Java, que le proporciona gran portabilidad y facilidad en la orientación a objetos, y la otra razón consiste en que implementa DEVS e incorpora agentes, por medio de clases Java especiales. De manera que para simular la conducta de una Unidad de Producción no hay que preocuparse de detalles de simulación: solo del modelo físico y lógico de recursos y proceso, y la especificación del comportamiento de cada agente. Un esquema que muestra la implementación de este simulador se muestra en la figura 7.9, donde aparecen en capas diferentes los objetos que representan a los recursos, al proceso y a los agentes que intervienen en la simulación.

Una de las capas corresponde al simulador DEVS y DESS (componentes discretos y contínuos intercambiando mensajes), que simula la evolución del proceso y de los recursos que se están utilizando, y se muestran en la parte inferior de la gráfica. Las salidas de este simulador corresponden a las mediciones generadas a partir de la evolución de las variables contínuas, las cuales se almacenan en una base de datos para su trazabilidad. En otra de las capas se encuentra la comunidad de agentes, en la cual los agentes que tienen asignado el rol de detección de eventos leen de esta base de datos y se comunican con los agentes supervisores, quienes actualizan la imagen del proceso y envían mensajes a los agentes gestionadores de holones sobre los posibles cambios en la configuración de los recursos para continuar con el plan de producción. En otra capa se encuentran los objetos asociados al sistema holónico de producción, quienes reciben los mensajes de los agentes y actualizan sus estados internos así como la configuración de la planta.

Para proporcionar independencia de los agentes con respecto al modelo, se puede optar

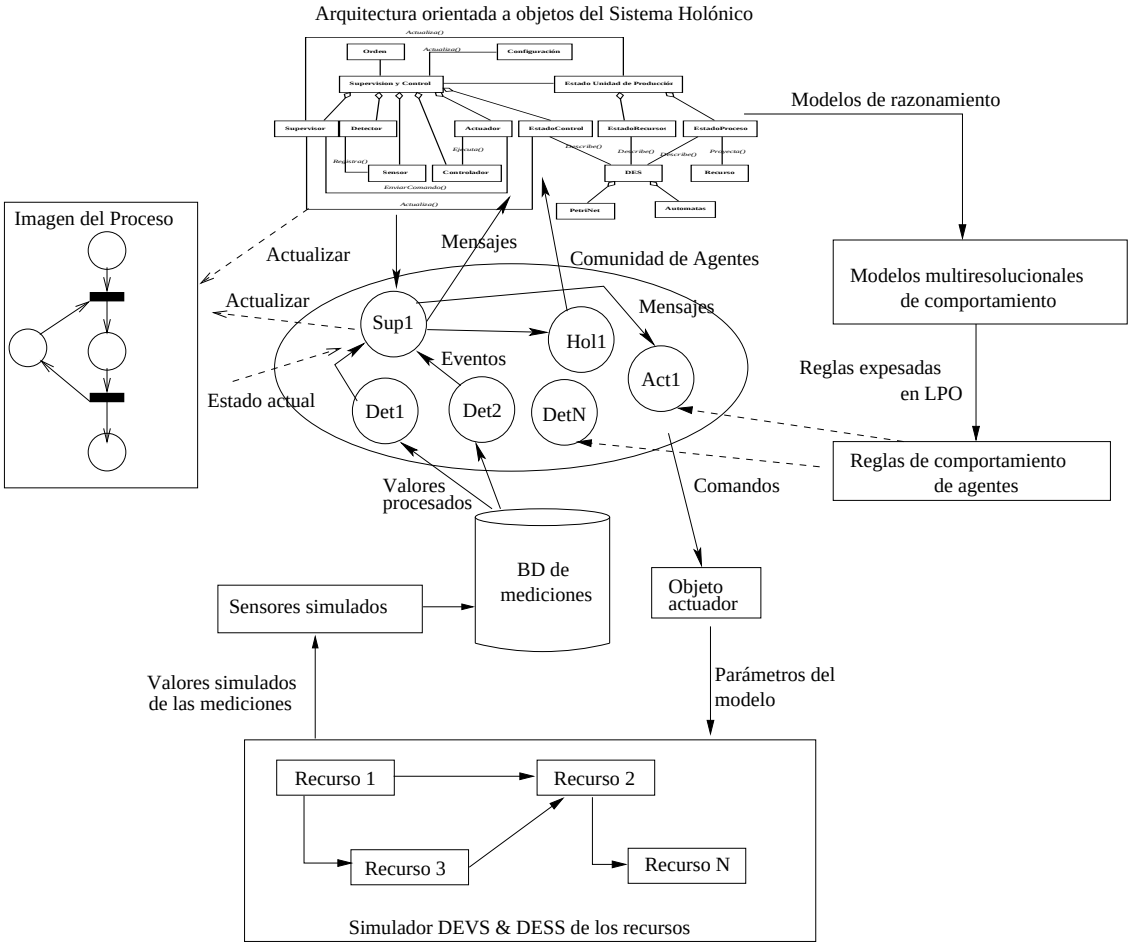


Figura 7.9: Esquema del simulador multi-agente para validar supervisión del sistema holónico

por utilizar un sistema multi-agente acoplado a un simulador DEVS, por medio de una interfaz en lenguaje Java, que permite que los agentes implementados en la plataforma multi-agente JADE puedan observar algunas variables de los objetos de simulación (recursos y mediciones del proceso) y tomar decisiones al respecto. Además, con la plataforma JADE los agentes pueden intercambiar información entre sí, e incluso negociar por medio del mecanismo contract-net para hacer más realista la simulación. Existe un agente asociado a cada holón dentro de la unidad de producción, es decir, uno para cada recurso, otro para gestionar la orden, otro maneja las especificaciones del producto, y además de ellos, están los agentes encargados de ejecutar la supervisión y control de la orden de producción, y la detección de eventos del proceso industrial.

El algoritmo general de simulación de la Unidad de Producción se basa en la propuesta presentada en la figura 7.6, y tiene los siguientes pasos:

1. Inicializar las variables globales de la unidad de producción, el tiempo de simulación y el reloj de simulación.

2. Inicializar los componentes asociados a cada recurso, un estado inicial y unos valores iniciales para el proceso fisico.

3. El sistema a eventos discretos también se inicializa, estableciendo un estado o marcación inicial, según el tipo de representación utilizada.

4. Establecer el período para que cada agente observe su entorno. Este período puede ser diferente para cada agente según su función. De hecho, un agente detector de eventos debe chequear su entorno más frecuentemente que un supervisor, o un agente gestionador de recurso.

5. Establecer la configuración inicial de la planta.

6. Generar la activación de componentes en el sistema, para que empiecen a evolucionar las variables asociadas al proceso físico.

7. Mientras que el reloj de simulación sea mayor que el tiempo de simulación hacer:

   *a*) Modificar las variables asociadas al componente donde tiene lugar el evento más próximo, integrando desde el tiempo 0 hasta el tiempo actual ($\Delta_T$).

   *b*) Procesar observación de variables continuas para el detector de eventos.

   *c*) Si se detecta un evento, iniciar diálogo entre agentes detector de eventos y supervisor.

   *d*) Iniciar diálogos entre supervisor y otros agentes, de acuerdo al evento detectado.

   *e*) Si es necesario, cambiar configuración de la planta.

   *f*) Actualizar reloj de simulación.

8. Imprimir estadísticas asociadas a cada recurso, la traza de las variables continuas, los diálogos entre agentes, la traza de control que hace seguimiento a la configuración y los estados discretos del proceso (secuencia de eventos o árbol de marcación)

## 7.4. Experimentación

Diversos experimentos de simulación se han ejecutado para verificar el método propuesto y la implementación computacional. Para este documento se van a presentar dos ejemplos, aunque se han desarrollado algunos más. Un ejemplo académico consiste en un sistema hidroneumático supervisado, mientras que un ejemplo más elaborado consiste en la logística de un Central Azucarero (CAZ).

### 7.4.1. Sistema hidroneumático supervisado

En el *anexo B* se muestra una aplicación a un ejemplo académico que consiste en controlar un sistema hidroneumático supervisado. El énfasis que se hace en este anexo

es el de mostrar la metodología de modelado y simulación que se presentó en la sección 7.2 de este capítulo. Este sistema se supervisa para mantener los valores de dos variables dentro de un rango de valores, cambiando la configuración del sistema cada vez que sea necesario. El enfoque de modelado del sistema a eventos discretos se basa en redes de Petri, mientras que la síntesis del supervisor se aplica utilizando arcos inhibidores. La simulación incorpora agentes que desempeñan las funciones del supervisor, detector de eventos y actuador. Una variante de este ejemplo modelando el proceso por medio de un autómata global para el estado del proceso y otro autómata local para indicar las regiones de operación, previa detección difusa de eventos, se publicó por Parra, Colina y Chacón en el marco de la validación de un sistema de control supervisorio inteligente [86, 87]. Como resultados de este experimento se tienen las trazas de variable continua y la marcación de la red de Petri.

### 7.4.2. Logística de aprovisionamiento de un CAZ

Este ejemplo es más elaborado, y se describe en el *anexo C*, el cual resume resultados de experimentos para simular el aprovisionamiento continuo de un Central Azucarero (CAZ) [85]. El énfasis de este ejemplo está en la descripción de todo el sistema bajo la filosofía holónica, inspirada en la arquitectura PROSA. El modelado de sistemas discretos se plantea utilizando redes de Petri acopladas, y su simulación se lleva a cabo proyectando a una red de componentes que intercambian mensajes. El objetivo de la simulación en este contexto es el de comparar diversos enfoques de coordinación de unidades de producción, como es la heterarquía absoluta (sin coordinación explícita); la jerarquía absoluta, donde la coordinación es centralizada; y el sistema holónico, donde se coordinan proceso y recursos por medio de una red de Petri.

Una vez desarrollados estos ejemplos, se demuestra que efectivamente es factible validar el comportamiento de sistemas holónicos supervisados por medio de la simulación de sistemas a eventos discretos. Esto se puede considerar como el origen de posterio-

res investigaciones, lo cual se resumirá en la sección que corresponde a conclusiones y recomendaciones.

## 7.5. Conclusión

Como aporte de este capítulo se tiene la derivación de una metodología que permite validar el comportamiento de unidades de producción supervisadas por medio de la combinación de un simulador diseñado bajo el formalismo de eventos discretos (DEVS) y de variable continua, acoplado a un sistema multi-agente que implementa las funciones de detección de eventos, gestión de las unidades de producción y supervisión, entre otras. Además, se hicieron algunos experimentos de simulación utilizando plataformas de simulación orientadas a agentes, como Galatea, y también con plataformas multi-agente independientes del simulador, como JADE. Algunos de estos experimentos fueron publicados en eventos internacionales, y proporcionan las bases para comprobar la validez de las hipótesis de trabajo propuestas.

# Capítulo 8

# Conclusiones y Recomendaciones

Durante el desarrollo de la investigación que concluyó en este trabajo doctoral se obtuvieron resultados, algunos de ellos provechosos, al igual que se experimentaron no pocas dificultades para lograr su culminación. Estos resultados constituyen las conclusiones que son el resultado de esta investigación y que presentamos en los siguientes párrafos.

Como resultado principal, se verificó la hipótesis de trabajo mencionada al inicio de este trabajo doctoral, donde se logró simular la conducta dinámica discreta y continua de una Unidad de Producción concebida bajo la arquitectura de automatización holónica PROSA, gestionada por agentes y supervisada por un sistema a eventos discretos, y se derivó un método para obtener el modelo a simular a partir de la especificación de la organización y el modelo numérico del proceso industrial a controlar. Este método se basa en el modelado de sistemas discretos y continuos, y simulación de sistemas multi-agente utilizando el formalismo DEVS. El mecanismo de supervisión de la unidad de producción holónica actúa sobre la orden de producción, tomando información que simula datos tomados en tiempo real del proceso físico y de la configuración de los recursos que pertenecen a esta unidad de producción, generando posteriores configuraciones de la misma unidad de producción, de manera que se

cumplan las metas establecidas de especificación y cantidad de producto.

Este mecanismo de supervisión se generó combinando diversos aspectos de sistemas dinámicos híbridos, métodos de síntesis de supervisores y esquemas de razonamiento basados en modelos multiresolucionales, y se aplica para ejercer control sobre la dinámica del sistema holónico, siguiendo la propuesta de descripción de Chacón *et al.* La implementación de la parte de razonamiento de los holones que conforman la Unidad de Producción se hizo mediante tecnología de agentes, los cuales son parte activa dentro de la metodología de modelado y simulación del control supervisorio en sistemas de producción continua. En detalle, los resultados obtenidos se muestran a continuación.

## 8.1. Análisis de resultados obtenidos

Como resultados de la investigación llevada a cabo, podemos mencionar los siguientes:

- Se generó un método que valida el control supervisorio implementado en un Sistema de Manufactura de procesos continuos. La novedad del método consiste en que hace uso de agentes que agregan inteligencia y capacidades de decisión y comunicación para controlar el proceso con base en metas y una programación dada, y una representación en Sistemas a Eventos Discretos del estado del sistema. Además, permite evaluar no solo la lógica del control supervisorio, sino su desempeño, lo que lo aproxima a los sistemas supervisores reales en cuanto a su trazabilidad.

- Se obtuvo una solución de computación para implementar este método que incorpora una imagen del proceso físico y una imagen discreta del proceso simuladas sincronizadamente bajo el paradigma DEVS. Los agentes interactúan con el proceso decidiendo en base a su imagen discreta, y toman decisiones sobre la configuración de la planta. Esta implementación produce trazas no solo

de las variables del proceso, sino de los estados cualitativos en que se encuentra el mismo y además brinda la capacidad de hacer un seguimiento a la evolución de la configuración del sistema y a los diálogos que mantienen los agentes entre sí y con los objetos del sistema.

- Uno de los aportes de la tesis consiste en la combinación del esquema de supervisión con el modelo multiresolucional de razonamiento, para obtener reglas de conducta del agente que implementan a un supervisor. Como novedad obtenida de este trabajo, se logró extender al modelo multiresolucional de Sanz para obtener una imagen local y global del proceso a supervisar, brindandole así mayor robustez al sistema supervisor.

- Se produjo un esquema de implantación del Control Supervisorio para una Unidad de Producción Continua concebida bajo la arquitectura holónica PROSA, utilizando los conceptos actuales para la arquitectura de computación y de telecomunicaciones. Por lo tanto, este esquema tiene vigencia y puede actualizarse con mayor facilidad.

- Se obtuvo un diseño basado en el Lenguaje Unificado de Modelado que obtiene el estado del proceso y los recursos del sistema holónico, y que incorpora a los agentes gestores de los holones bajo la arquitectura PROSA, los cuales interactúan con una clase especial denominada Supervisión y Control, cuya misión es la de ejercer el control supervisorio sobre este sistema holónico.

- Como consecuencia del diseño antes mencionado, se obtuvo un conjunto de clases en lenguaje java que implementan a un DES, que funciona con máquinas de estado finito y con redes de Petri, es decir, tiene doble funcionalidad y sirve para registrar la evolución de un sistema real, simularlo y también generar trayectorias de eventos.

- Aunque el uso de tecnologías de agente en control no es nuevo, si lo es la generación de reglas de comportamiento de los agentes a partir de la dinámica de un sistema a eventos discretos supervisado y el esquema conmutado para carga de reglas, el cual permite que los agentes cambien su comportamiento para adaptarse a nuevas condiciones en su entorno. Así como la interacción entre los agentes gestionadores de holones PROSA con los agentes basados en funciones: supervisor, detector de eventos y actuador.

- Para profundizar más en esta dirección recomendamos simular de la misma manera la coordinación de varios holones, para evaluar cuanto influyen los procesos de negociación, por ejemplo, y también la composición de la dinámica de los estados de cada unidad de producción en una dinámica global que represente regiones de operación.

- Se efectuaron experimentos de validación y salieron exitosos. Aunque el ejemplo seleccionado es académico, representa en sus aspectos a un sistema de producción continua. Este mismo esquema de validación funciona para sistemas de mayor complejidad, siempre y cuando se sigan los siguientes lineamientos: a) modelar correctamente el fenómeno físico de transformación, b) representar bajo esquema holónico al sistema complejo, de forma bottom-up, es decir, desde unidades de producción simple hasta conformar una unidad más compleja c) representar adecuadamente la dinámica de la conducta de este sistema bajo este mismo esquema bottom-up y d) Definir una ontología para la negociación entre unidades de producción. Algunos de estos aspectos se trataron en este trabajo doctoral, otros solamente se mencionaron.

## 8.2. Indicaciones para futuros trabajos

A su vez, para la posterior continuación de esta tesis, así como una guía para futuras publicaciones, podemos establecer las siguientes recomendaciones:

- Considerar el control supervisorio sobre la arquitectura holónica propuesta por Deen y Fletcher, lo que implica encontrar una manera de describir la conducta de una Unidad de Producción vista bajo esta arquitectura.

- Profundizar sobre generación automática de reglas de comportamiento de agentes a partir de sistemas a eventos discretos, observando las pre y post condiciones del disparo de transiciones dentro de la red de Petri que modela la conducta de todo el proceso supervisado, y luego establecer las reglas para disparar las transiciones del supervisor, siempre y cuando estas sean controlables. Es necesario un mecanismo para diferenciar eventos no sólo para aquellos controlables y no controlables, sino que también si son ejecutables por agentes o por otras entidades.

- Ensayar con la generación de trayectorias para sistemas supervisados no solo con DES, sino con agentes, para introducir explícitamente el tiempo, y para facilitar el establecimiento de puntos de chequeo del proceso.

# Bibliografía


[1] L. Acebes, R. Alves, A. Merino, and C. De Prada. Un entorno de modelado inteligente y simulación distribuída de plantas de proceso. *Revista Iberoamericana de Automática e Informática Industrial*, 1:42–58, 2004.

[2] Emmanuel Adam. *Modèle d'organisation multi-agent pour l'aide au travail coopératif dans les processus d'enterprise: application aux systèmes administratifs complexes*. PhD thesis, Université de Valenciennes et du Hainaut-Cambrèsis. Discipline: Informatique, 2000.

[3] E. Altamiranda, E. Chacón, and E. Colina. A holonic architecture for continuous processes supervision and coordination. *WSEAS Transactions on Systems*, 4:941–945, 2005.

[4] J. Banks, J. Carson, D. Nelson, and E. Nicol. *Discrete-Event System Simulation*. Prentice Hall International Series in Industrial and Systems Engineering, 1999.

[5] Pascal Blanc. *Pilotage par Approche Holónique d'un Systema de Production de Vitres de Securite Feuilletees*. PhD thesis, Université de Nantes, 2006.

[6] L. Bongaerts, H. Van Brussel, J. Wyns, P. Valckenaers, and T. Van Ginderachter. A conceptual framework for holonic manufacturing : Identification of manufacturing holons. *Journal of manufacturing systems*, 18:35–52, 1999.

[7] L. Bongaerts, H. Van Brussel, J. Wyns, P. Valckenaers, and P. Peeters. Designing holonic manufacturing systems. *Robotics and computer-integrated manufacturing*, 14:455–464, 1998.

[8] Luc Bongaerts. *Integration of Scheduling and Control in Holonic Manufacturing Systems*. PhD thesis, Katholieke Universiteit Leuven, 1998.

[9] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, 1999.

[10] H. Van Brussel, J. Wyns, P. Valckenaers, L. Bongaerts, and P. Peeters. Reference architecture for holonic manufacturing systems. *Computers in Industry*, 37:255–274, 1998.

[11] S. Bussmann, N. Jennings, and M. Wooldridge. *Multiagent Systems for Manufacturing Control. A design Methodology*. Springer Verlag, 2004.

[12] P. Caines and S. Wang. Cocolog: A conditional observer and controller logic for finite machines. Technical Report, 1994.

[13] G. Caire, T. Trucco, F. Bellifemine, and G. Rimmasa. Jade tutorial using. Technical Report, 2002.

[14] Giovanni Caire. Jade programming for beginners. Technical Report, 2003.

[15] N. Carver and V. Lesser. The evolution of blackboard control architectures. CMPSCI Technical Report 92-71, 1992.

[16] C. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Kluwer Academic Publishers, 1999.

[17] Antonio Cembellín. Simulación de sistemas de control híbrido. *Memorias de las XXIV Jornadas de Automática, comité español de automática*, 2003.

[18] E. Chacón, I. Besembel, and J. Hennet. Coordination and optimisation in oil and gas production complexes. *Computers in Industry*, 53:17–37, 2004.

[19] E. Chacón and G. DeSarrazin. Automatización de sistemas de producción. Reporte Técnico, Universidad de Los Andes, 2004.

[20] Edgar Chacón, Gisela De Sarrazin, and Yanira Khodr. Coupled dynamics for industrial complex system. *ISA Transactions*, 4:305–319, 2001.

[21] Edgar Chacón, Ferenc Szigeti, and Oscar Camacho. Integral automation of industrial complexes based on hybrid systems. *Nonlinear Analysis Theory, Methods and Applications*, 47:1561–1570, 2001.

[22] J. Christensen. Holonic manufacturing systems: Initial architecture and standards directions. In *First European Conference on Holonic Manufacturing Systems*. European HMS Consortium, Hanover, Germany, 1994.

[23] CIMOSA. Cimosa: A primer on key concepts, purpose and business value. http://cimosa.cnt.pl/Docs/Primer/primer0.htm, 2000.

[24] A. Cockburn. Using goal-based use cases. *JOOP*, 10:56–62, 1997.

[25] A. Collinot, A. Drogoul, and P. Benhamou. Agent-oriented design of soccer robot team. In *Second Internacional Conference on Multi Agent Systems ICMAS'96*, pages 41–47,. AAAI Press, 1996.

[26] A. Colombo, R. Neubert, and B. Sussmann. A coloured petri net-based approach towards a formal specification of agent-controlled production systems. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 2002.

[27] V. Dalmoro and J. Palazzo. Pera - arquitectura de referencia de empresa purdue. universidade federal do rio grande do sul. Technical Report, 2000.

[28] J. Davila, K. Tucci, and M. Uzcategui. Towards a logic-based, multi-agent simulation theory. In *International Conference on Modeling, Simulation and Neural Networks MSNN'2000*, pages 199–215, 2000.

[29] C. Dellarocas and M. Klein. Civil agent societies: Tools for inventing open agent-mediated electronic marketplaces. In *ACM Conference on Electronic Commerce - EC99 at IJCAI'99*, 1999.

[30] F. DiCesare and A. Desroches. Modeling, control and performance analysis of automated manufacturing systems using petri nets. *Control and Dynamic Systems*, 47:121–172, 1991.

[31] C. Domingo, G. Tonnella, and M. Sananes. *GLIDER's Reference Manual*. Universidad de Los Andes, 1996.

[32] J. Dávila and M. Uzcátegui. Galatea: a multi-agent simulation platform. In *International Conference on Modeling, Simulation and Neural Networks MSNN'2000*, 2000.

[33] Jacinto Dávila. *Agents in Logic Programming*. PhD thesis, University of London, 1997.

[34] Engormix. Cosechadoras de caña de azucar case ih. cosechadoras serie a. http://www.engormix.com/cosechadorascanaazucarcasesarticulos1144-AGR.htm, 2006.

[35] EURESCOM. Message: Methodology for engineering systems of software agents. Technical Report, 2001.

[36] Peynman Faratin. *Automated Service Negotiation between Autonomous Computational Agents*. PhD thesis, Queen Mary and Westfield College, London, 2000.

[37] J. Ferber and J. Müller. Influences and reactions: a model of situated multiagent systems. In *ICMAS 96*, pages 72–79, 1996.

[38] T. Finin, Y. Labrou, and J. Mayfield. Kqml as an agent communication language. *Software agents*, pages 291–316, 1997.

[39] T. Fischer. *Batch Control System; Design, Application and Implementation.* Instrument Society of America, 1990.

[40] Paul Fishwick. *Simulation Model Design and Execution. Building digital worlds.* Prentice Hall International Series in Industrial and Systems Engineering, 1995.

[41] M. Fletcher and M. Deen. Fault-tolerant holonic manufacturing systems. *Concurrency and computation: Practice and Experience*, 13:43–70, 2001.

[42] Foundation for Intelligent Physical Agents. Fipa acl message structure specification. SC00061G Technical Report, web: http://www.fipa.org/specs/fipa00061/index.html, 2000.

[43] John P. Van Gigch. *System Design Modeling and Metamodeling.* Plenun Press, 1991.

[44] A. Giret and V. Botti. Aplicaciones de los sistemas multiagente (sma) en industria. In *Agentes inteligentes: Sistemas Multiagentes y aplicaciones*, pages 19–82. Editorial Club Universitario, 2002.

[45] Adriana Giret. *ANEMONA: Una metodología multi-agente para sistemas holónicos de fabricación.* PhD thesis, Universidad Politécnica de Valencia, 2005.

[46] J. Gómez and J. Pavón. Meta-modelling in agent oriented software engineering. In *(th Ibero-American Conference in AI (Iberamia 2002). Advances in Artificial Intelligence*, pages 606–615,, 2002.

[47] C. Golaszewski and P. Ramadge. Control of discrete event processes with forced events. In *IEEE Proceedings of the 26th Conference on Decision and Control*, 1987.

[48] David Gouyon. *Controle par le produit des systemes d'execution de la production: apport des techniques de synthèse*. PhD thesis, Département de Formation Doctorale Automatique, Université Henri Poincaré, 2004.

[49] The OMG group. The unified modeling language, 1998.

[50] The OMG group. Omg systems modeling language (omg sysml), version 1.0, 2007.

[51] A. Guasch, M. Piera, J. Casanovas, and J. Figueras. *Theory of Modelling and Simulation. 2nd. Edition.* Academic Press, 2000.

[52] Anders Hellgren. *On the Implementation of Discrete Event Supervisory Control*. PhD thesis, Chalmers University of Technology, Göteborg, Sweden, 2002.

[53] Fu-Shiung Hsieh. Holarchy formation and optimization in holonic manufacturing systems with contract net. *Automatica*, 44:959–970, 2008.

[54] C. Jacques and G. Wainer. Using the cd++ devs toolkit on develop petri nets. In *2002 summer Computer Simulation Conference*, 2002.

[55] Nick Jennings. *Cooperation in industrial multi-agent systems*. World Scientific, 1994.

[56] E. Jiménez, M. Pérez, and F. Sanz. Modelado y simulación de sistemas logísticos y de producción mediante redes de petri. *Revista Iberoamericana de Automática e Informática Industrial*, 2:39–53, 2005.

[57] Charlotta Johnsson. Recipe-based batch control using high-level grafchart. Master's thesis, Department of Automatic Control, Lund Institute of Technology, 1997.

[58] Charlotta Johnsson. *A Graphical Language for Batch Control.* PhD thesis, Lund Institute of Technology, Sweden, 1999.

[59] D. Kelton, R. Sadowski, and D. Sadowski. *Simulation with Arena.* McGraw-Hill Series in Industrial Engineering and Management Science, 1998.

[60] E. Kendall, M. Malkoun, and C. Jiang. A methodology for developing agent based systems. *Distributed Artificial Intelligence - Architecture and Modelling*, 1087:85–99, 1996.

[61] D. Kinny and M. Georgeff. *Modelling and Design of Multi-Agent Systems.* Springer-verlag, 1997.

[62] A. Koestler. *The Ghost in the Machine.* Arkana Books, 1967.

[63] Dilip Kotak, Shaohong Wu, Martin Fleetwood, and Hiroshi Tamoto. Agent-based holonic design and operations environment for distributed manufacturing. *Computers in industry*, 52:95–108, 2003.

[64] Robert Kowalski. How to be artificially intelligent, 2005.

[65] Robert Kowalski and Fariba Sadri. Towards a unified agent architecture that combine rationality with reactivity. In *LID'96 workshop on Logic in databases*, 1996.

[66] Averill Law and David Kelton. *Simulation Modeling and Analysis.* McGraw-Hill Series in Industrial Engineering and Management Science, 2000.

[67] Paulo Jorge Pinto Leitao. *An Agile and Adaptive Holonic Architecture for Manufacturing Control*. PhD thesis, Department of Electrotechnical Engineering, Instituto Politécnico da Braganca, 2004.

[68] J. Lygeros, D. Godbole, and S. Sastry. Simulation as a tool for hybrid system design. In *Fifth IEEE conference on AI, Simulation and Planning in High-Autonomy Systems*, pages 6–12, 1994.

[69] John Lygeros. *Hierarchical, Hybrid Control of Large Scale Systems*. PhD thesis, Universidad of California, Berkeley, 1996.

[70] E. Mamdani. Application of fuzzy algorithms for control of simple dynamic plant. *Proceedings of the IEEE*, 121:1585–1588, 1974.

[71] B. Mandelbrot. *Los objetos fractales. Forma, azar y dimensión*. Tusquets Editores, S.A, 1993.

[72] F. Maturana, W. Shen, and D. Norrie. Metamorph: An adaptive agent-based architecture for intelligent manufacturing. *International Journal of Production Research*, 37(10):2159–2174, 1999.

[73] S. Miles, M. Joy, and M. Luck. Designing agent-oriented systems by analysing agent interactions. *Agent-Oriented Software Engineering*, Vol. 1657, 2001.

[74] J. Millan and S. D. O'Young. Online discrete event supervisory control of hybrid dynamical systems using embedded simulation. In *2006 8th International Workshop on Discrete Event Systems*, pages 137–142, 2006.

[75] J. Müller. The design of intelligent agents: a layered approach. In *Volume 1177 of Lecture notes in artificial intelligence*. Springer Verlag, 1996.

[76] J. Moody and P. Antsaklis. *Supervisory Control of Discrete Event Systems using Petri nets*. Kluwer Academic Publishers, 1998.

[77] M. Morari and A. Bemporad. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35 (3):407–427, 1999.

[78] G. Morel, P. Valckenaers, J. Faure, C. Pereira, and C. Diedrich. Manufacturing plant control challenges and issues. *Control Engineering Practice*, 15:1321–1331, 2007.

[79] R. Mott. *Applied Fluid Mechanics*. Maxwell Macmillan International Editions, 1990.

[80] Tadao Murata. Petri nets, properties, analysis and applications. *Proceedings of the IEEE*, 77, 1989.

[81] J. Mylopoulos, M. Kolp, and J. Castro. Uml for agent-oriented software development: the tropos proposal. In *4th Internacional Conference on the Unified Modeling Language UML'01*, 2001.

[82] A. Nerode and W. Kohn. Models for hybrid systems: Automata, topologies, stability, 1993.

[83] K. Ogata. *Ingeniería de Control Moderna*. Editorial Mc-Graw Hill, 1993.

[84] M. Ortín, J. García, B. Moros, and J. Nicolás. El modelo de negocio como base del modelo de requisitos. In *Actas de las Jornadas de Ingeniería de Requisitos Aplicada, Sevilla*, 2001.

[85] C. Parra and E. Chacón. Simulación de la logística de aprovisionamiento de un central azucarero (caz). In *Memorias del XIII Congreso Latinoamericano de Control Automático (CLCA), Mérida*, 2008.

[86] C. Parra, E. Colina, and E. Chacón. Intelligent supervisory control design framework for fault exposed processes. *International Journal of Circuits, Systems and Signal Processing*, 3:251–258, 2007.

[87] C. Parra, E. Colina, and E. Chacón. Intelligent supervisory control of continuous processes exposed to faults: a practical approach. In *MATH'08: Proceedings of the American Conference on Applied Mathematics*, pages 252–258, 2008.

[88] C. Parra, E. Colina Morles, and E. Chacón. Design framework for intelligent supervision of industrial processes. *WSEAS Transactions on Systems*, 7(7):616–625, 2008.

[89] Carlos Parra and Edgar Chacón. Evaluación del desempeño de procesos de manufactura proyectando redes de petri hacia redes de eventos discretos. In *XII Latin-American Congress in Automatic Control*, 2006.

[90] James Peterson. *Petri Net Theory and the Modeling of Systems*. Prentice-Hall International, 1981.

[91] M. E. Porter. *Competitive Advantage: Creating and Sustaining Superior Performance*. Free Press, 1985.

[92] PABADIS Project. Plant automation based on distributed systems. revolutionising plant automation - the pabadis approach. Technical Report, 2003.

[93] J. Ramadge and M. Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal of Control and Optimization*, 25:206–230, 1987.

[94] A. Ritter, W. Baum, M. Hopf, and E. Westkamper. Agentification for production systems. In *Second Internacional Workshop on Integration of Specification Techniques for Applications in Engineering*, 2002.

[95] D. Ríos, S. Ríos, and J. Martín. *Simulación. Métodos y Aplicaciones*. Editorial AlfaOmega Ra-Ma, 2000.

[96] J. Rosenschein and G. Zlotkin. Mechanisms for automated negotiation in state oriented domains. *Journal of Artificial Intelligence Research*, 5:163–238, 1996.

[97] S. Russel and P. Norvig. *Inteligencia Artificial: un enfoque moderno*. Prentice Hall, 1995.

[98] Rafael Sanz. *Arquitectura de Control Inteligente de Procesos*. PhD thesis, 1991.

[99] R. Sarrate and J. Aguilar. Vigilancia de un proceso a partir de la detección de eventos significativos. In *XXIV Jornadas de Automática. Universidad de León. España*, 2003.

[100] Ramón Sarrate. *Supervisió Intelligent de Processos Dinàmics basada en Esdeveniments*. PhD thesis, 2002.

[101] M. Shaw and A. Whinston. A distributed knowledge-based approach to flexible automation: The contract net framework. *International Journal of Flexible Manufacturing Systems*, 1:85–104, 1988.

[102] M. Silva. *Las redes de Petri en la automática y la informática*. Editorial AC, 1985.

[103] Jean Marcelo Simao. *A Contribution to the development of HMS simulation tool and proposition of a Meta-model for Holonic Control*. PhD thesis, Centre de Recherche en Automatique de Nancy (CRAN), 2005.

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

[105] César Torrico. Implementacao de controle supervisório de sistemas a eventos discretos aplicado a processos de manufatura. Master's thesis, Universidade Federal de Santa Catarina, Florianópolis, Brasil, 1999.

[106] P. Valckenaers, H. Van Brussel, Hadeli, O. Bochmann, B. Saint German, and C. Zamfirescu. On the design of emergent systems: an investigation on integra-

tion and interoperability issues. *Engineering Applications of Artificial Intelligence*, 16:377–393, 2003.

[107] Paul Valckenaers, Hadeli, Bart Saint Germain, Paul Verstraete, and Hendrik Van Brussel. Emergent short-term forecasting through ant colony engineering in coordination and control systems. *Advancen Engineering Informatics*, 20:261–278, 2006.

[108] G. Weiss. *Multiagent Systems: A modern approach to Distributed Artificial Intelligence*. MIT Press, 1999.

[109] M. Wooldridge, N. Jennings, and D. Kinny. The gaia methodology for agent-oriented analysis and design. *Journal of Autonomous Agents and Multi-Agents Systems*, 15, 2000.

[110] Jo Wyns. *Reference Architecture for Holonic Manufacturing Systems*. PhD thesis, Katholieke Universiteit Leuven, 1999.

[111] L. Zadeh. Outline of a new approach of the analysis of complex systems and dcision processes. *IEEE transactions of Systems, Man and Cybernetics*, 3:28–44, 1973.

[112] F. Zambonelli, N. Jennings, and M. Wooldridge. Developing multiagent systems: the gaia methodology. *ACM Transactions on Software Engineering and Methodology*, 12:317–370, 2003.

[113] B. Zeigler, H. Praehofer, and T. Kim. *Theory of Modelling and Simulation. 2nd. Edition.* Academic Press, 2000.

[114] B. Zeigler, H. Sang Song, T. Gon Kim, and H. Praehofer. Devs framework for modelling, simulation, analysis and design of hybrid sistems. *LectureNotes In Computer Science; Hybrid Systems II*, 999:529–551, 1995.

[115] M. Zhou and F. DiCesare. *Petri Net Synthesis for Discrete Event Control of Manufacturing Systemas*. Kluwer Academic Publishers, 1993.

[116] L. Zérega, T. Hernández, and J. Valladares. Evaluación de 14 variedades de cana de azúcar en dos suelos afectados por sales bajo condiciones de umbraculo. *Caña de azúcar*, 9:81–98, 1991.

# Anexo A

# Ejemplo de síntesis de un Supervisor utilizando redes de Petri

Un ejemplo académico para ilustrar la aplicación de los pasos de síntesis de un supervisor basado en el enfoque de redes de Petri se presenta a continuación: se trata de una máquina sujeta a fallas. La máquina se utiliza para procesar piezas desde una cola de entrada, y las partes terminadas se mueven a una cola de salida por un vehículo automatizado (AGV). La máquina puede fallar y dañar una pieza durante su operación, lo que implica que las partes dañadas se mueven a otra cola por otro AGV. Este comportamiento se captura en el modelo de la planta mostrado en la figura A.1.

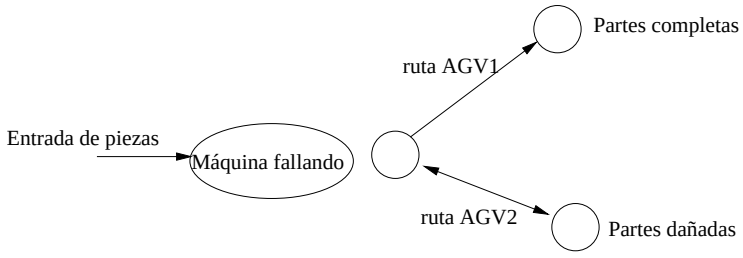


Figura A.1: Proceso de piezas con una máquina sujeta a fallas

La red de Petri que representa a cada máquina se representa en la figura A.2. El proceso tiene dos transiciones no controlables: t2, puesto que el controlador no puede

forzar a que termine un proceso de una pieza, y t6 que representa la falla de la máquina. Existe espacio para un solo AGV a la vez, de manera que solo uno de ellos puede acercarse a la máquina y retirar la pieza, sea completa o dañada. Así mismo, hay espacio solamente para una pieza, y si la máquina falla, hay que esperar a que la máquina se repare antes de empezar a trabajar sobre una nueva pieza.

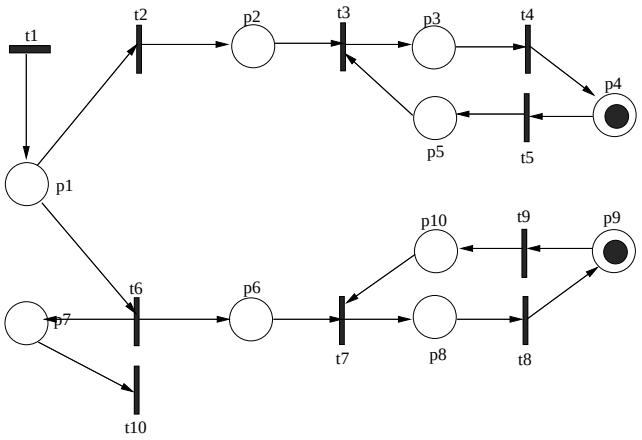


Figura A.2: Red de Petri que esquematiza al ejemplo

Las especificaciones anteriores llevan a las siguientes restricciones lineales:

$$\mu_5 + \mu_{10} \leq 1 \tag{A.1}$$

$$\mu_2 + \mu_6 \leq 1 \tag{A.2}$$

$$\mu_1 + \mu_7 \leq 1 \tag{A.3}$$

Donde $\mu_i$ representa la marcación que existe en el lugar $i$, y cada desigualdad indica que la suma de determinadas marcaciones no debe superar el valor indicado, que en este caso es de 1. Por ejemplo, si $\mu_2$ tiene marca, entonces no la debe tener $\mu_6$. A partir de estas restricciones, se sintetizará un Control Supevisorio que no permita que la marcación de la red viole las mencionadas desigualdades. La descripción de cada lugar en la red de Petri se muestra en la tabla A.1:

Tabla A.1: Descripción de los lugares de la red de Petri del ejemplo.

| Lugar | Descripción |
|---|---|
| P1 | Máquina ocupada procesando pieza |
| P2 | Pieza esperando a ser transportada a cola de piezas completadas |
| P3 | Pieza se está transportando por AGV1 |
| P4 | AGV1 está disponible |
| P5 | AGV1 está en posición de tomar la pieza de la máquina |
| P6 | Pieza está esperando para transferirse a cola de piezas dañadas |
| P7 | Máquina esperando a ser reparada |
| P8 | Pieza se está transportando a cola de piezas dañadas |
| P9 | AGV2 está libre |
| P10 | AGV2 está en posición de tomar la pieza dañada de la máquina |

La matriz de incidencia que describe el comportamiento de este modelo es:

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

La matriz que contiene las transiciones no controlables $D_{uc}$ corresponde a las columnas dos y seis de la matriz $D_p$. Las restricciones A.1 a A.3 se representan en forma matricial de la siguiente manera:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \mu_p \leq \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Aplicando el criterio de admisibilidad se tiene:

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

Puede observarse claramente que la segunda fila no cumple con el criterio de admisibilidad, y corresponde a la restricción A.2. Por lo tanto deben manipularse las restricciones para que se cumpla el criterio de que $l^T Duc \leq 0$ implique admisibilidad. Primero se extienden las de la matriz de transiciones no controlables extendida con las que expresan el criterio de admisibilidad, resultando en la siguiente matriz:

$$\begin{bmatrix} D_{uc} \\ LD_{uc} \end{bmatrix}$$

Una operación sobre las filas de esta matriz consiste sumar la fila 1 (columna dos y seis de la matriz de incidencia) con la fila 12 (la que no cumple el criterio de admisibilidad), para dejar sus elementos con valor nulo. Esa operación hecha sobre una fila corresponde a agregar $\mu_1$ a la segunda restricción: $\mu_2 + \mu_6 \leq 1$. De manera que esta restricción se convierte en $\mu_1 + \mu_2 + \mu_6 \leq 1$. Dado el conjunto admisible de restricciones, el supervisor se obtiene mediante la siguiente operación:

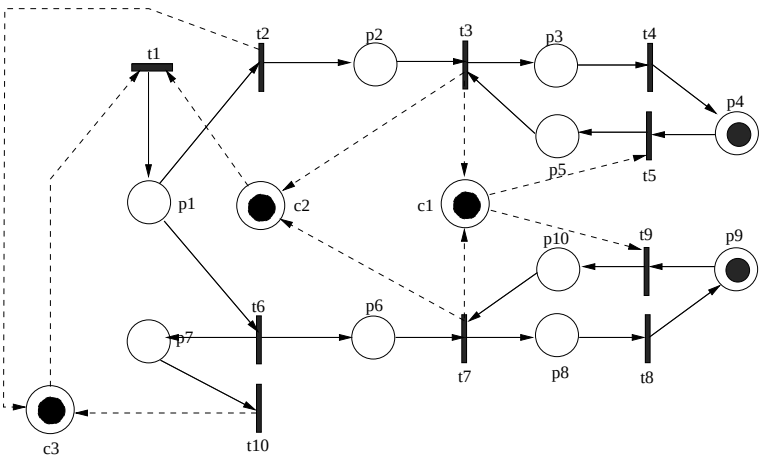


Figura A.3: Red de Petri del sistema supervisado

$$D_c = L'D_p = -(R_1 + R_2L)D_p = \begin{bmatrix} 0 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Y luego se calcula la marcación inicial a los lugares que corresponden a la acción del supervisor:

$$\mu_{c0} = b' - L'\mu_{p0} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

En la figura A.3 se puede apreciar el supervisor que se agrega a la red de Petri, en forma de tres lugares C1, C2 y C3 conectados a las transiciones ya existentes. La linea punteada indica las transiciones adicionales que se agregan para cumplir con las restricciones A.1, A.2 y A.3. También se puede apreciar la marcación inicial de la red, tanto en los lugares asociados al proceso como en los asociados al supervisor. Los lugares que representan la acción del supervisor habilitan ciertas transiciones, y a su vez algunas transiciones permiten que estos lugares adquieran marcación nuevamente.

# Anexo B

# Aplicación de la metodología de modelado y simulación a un Sistema Hidroneumático Supervisado

Un caso de estudio para diseñar y evaluar el control supervisorio sobre un proceso continuo se tiene en el problema del control de un *sistema hidroneumático*. El sistema que se quiere supervisar tiene los siguientes componentes: un depósito de agua de forma cilíndrica, el cual contiene agua y aire, un tanque de almacenamiento externo, una motobomba y un compresor. El objetivo global del sistema consiste en suministrar agua a los usuarios, manteniendo los niveles de su depósito y presión de aire dentro del rango de valores adecuado. La presión del aire determina la calidad de suministro a la demanda de agua de los usuarios del servicio, por lo tanto se debe controlar también. Si el nivel de agua desciende mas allá del valor se debe bombear agua por medio de una motobomba, que permite un flujo de entrada al depósito, mientras que si sube mas allá del valor máximo, se detiene el bombeo. El esquema que representa a este sistema se puede observar en la figura B.1, donde se pueden apreciar los componentes y las variables que se miden para estos subsistemas. Para

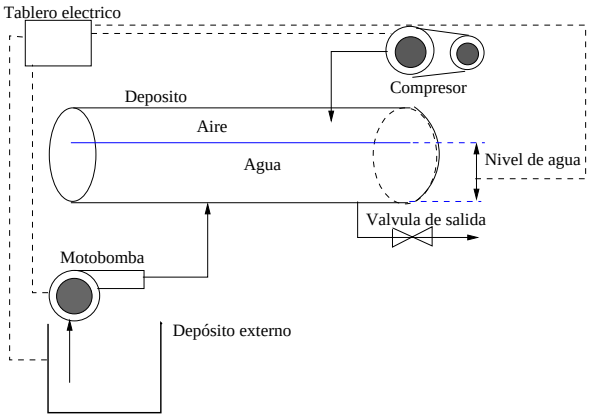


Figura B.1: Sistema de surtidor de agua hidroneumático

ello se asumirá una descripción holónica de este sistema cuyo proceso de control de niveles de agua y presión estará supervisado. Los holones recurso estarán gestionados por agentes, así como el supervisor y detector de eventos. La evolución dinámica de este sistema holónico se validará por medio del modelado y simulación multi-agente.

Algunas características del sistema, sus componentes y las restricciones de diseño que se deben cumplir son las siguientes:

- El depósito principal tiene la forma de un cilindro acostado.
- La presión debe estar entre 2 y 6 $kg - f/cm^2$.
- El nivel del depósito debe oscilar entre un valor mínimo y máximo.
- La bomba no debe succionar en vacío, por lo tanto si el nivel del depósito externo está por debajo del mínimo permitido ésta debe apagarse.
- Las demandas de agua por parte de los usurios son de naturaleza aleatoria.
- La bomba debe apagarse cuando el nivel del depósito principal alcance un valor máximo.

- Si la presión desciende mas allá del valor mínimo se inyecta aire por medio de un compresor hasta que supere este nivel, luego del cual se apaga para que la presión siga dependiendo del nivel de agua.

- Si el nivel del tanque externo desciende mas allá del mínimo permitido, no se puede encender la bomba de succión.

- La cantidad de aire encerrado dentro del depósito principal tiende a disminuir con el tiempo, ya que parte del aire se escapa con el agua que se entrega a la demanda.

Con esta especificación inicial del sistema, se procede entonces a aplicar la metodología propuesta de modelado y simulación para determinar la validez del control supervisorio propuesto para este sistema holónico.

## B.1. Fase de Aplicación del Método de Diseño

En esta sección se presenta una idea aproximada del comportamiento de la unidad de producción autónoma a lazo abierto, es decir, sin intervención de los mecanismos de control, y luego se diseña el mecanismo de control para supervisar su conducta y una especificación del modelo de razonamiento para implementar al supervisor. Esta descripción del comportamiento a lazo abierto tiene tres componentes fundamentales: un modelo matemático en variable continua que describe al proceso continuo, una descripción organizativa del sistema a estudiar, y un modelo matemático que describa la lógica de las operaciones de la planta.

### B.1.1. Modelo en variable continua

Para describir el comportamiento físico del sistema se consideran como variables de estado los niveles de agua en los dos depósitos y la presión en el tanque cilíndri-

co acostado. La evolución dinámica de estas variables se representa por medio de ecuaciones diferenciales ordinarias, y su formulación se presenta a continuación:

**Niveles de los depósitos.** Existen dos depósitos cilíndricos, donde el principal tiene la forma de un cilindro acostado, con radio $r$. La ecuación diferencial que relaciona el nivel de agua con los flujos de entrada y salida son:

$$h_1'(t) = \left\{ \frac{\lambda_1(t) - \mu_1(h_1(t))}{r^2 \cos^{-1}\left(\frac{r-h_1}{r}\right) - (r-h_1)\sqrt{2rh_1 - h_1^2}} \right\}$$

Donde $h_1'(t)$ es la tasa de cambio del nivel de agua en el tanque y $h_1$ es la altura instantánea, en un instante determinado. Los flujos de entrada y de salida los describimos por medio de las siguientes expresiones:

$$\lambda_1(t) = \begin{cases} k \text{ si } \mathrm{h}_1(t) \leq h_{max} \\ 0 \text{ en caso contrario} \end{cases}$$

El depósito externo también tiene forma cilíndrica con radio $R$, y su ecuación para el nivel es:

$$h_2'(t) = \frac{\lambda_2(t) - \mu_2(h(t))}{\pi R^2}$$

**La bomba hidráulica.** Se dispone de una bomba para llevar agua al depósito principal desde el depósito externo, la cual está gobernada por la ecuación de carga de la bomba, y por la potencia nominal.

$$w = \frac{550HP}{H}$$

Donde $H$ es la carga de la bomba, $HP$ es la potencia nominal de la bomba, $w$ es el flujo másico de agua, $\rho$ es la densidad del agua. Dado que la densidad del agua es 1 $Kg/m^3$, este flujo másico es equivalente al flujo de entrada $\lambda(t)$ descrito anteriormente. Este

flujo másico de entrada es equivalente al volumen de agua entrante multiplicado por su densidad $\rho$. Dado que $\rho$ es 1 $kg/dm^3$, $w$ $Kg/s$ equivale a $\lambda(t)/1000$ $m^3/s$. Más referencias sobre este fenómeno se pueden encontrar en Mott [79].

**La presión en el tanque principal.** El aire es un fluido compresible, por lo tanto un volumen determinado de aire se puede confinar a un volumen menor y tiene el efecto físico de que aumenta la presión resultante. Su comportamiento se puede aproximar con la siguiente ecuación, suponiendo que el aire se comporte como un gas ideal:

$$P'(t) = P_0 \frac{-V'(t)}{V_t}$$

Donde $P_0$ es la presión inicial, y $V_t$ es el volumen ocupado en el tiempo $t$. Suponiendo que el proceso sea isotérmico, la temperaturas se mantiene constante, de manera si disminuye el volumen disponible para el aire, entonces su presión aumentará y viceversa. Cada vez que se entrega un $m^3$ de agua, se pierden aproximadamente 22 $dm^3$ de aire, de manera que a largo plazo la presión debe disminuir.

### B.1.2. Regiones de Operación y proyección a variables discretas

Para facilitar la proyección de las de variables continuas a un conjunto de estados discretos, es necesario definir unas regiones de operación, donde los valores de las variables continuas como presión y niveles de agua se proyecten a un valor discreto. Además el sistema tiene unas variables discretas las cuales representan el estado de la bomba (encendida, apagada) y el estado del compresor (encendido, apagado). La figura B.2 muestra las regiones de operación para la presión del tanque acostado, existen tres regiones en todo tiempo $t$: presión baja, cuando es inferior a $1{,}41kg -$

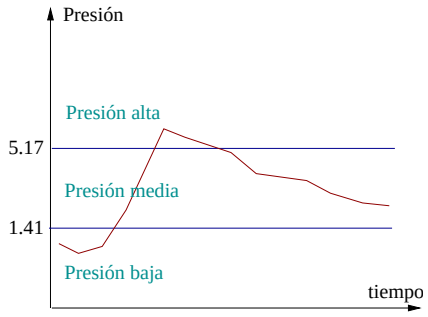


Figura B.2: Regiones de operación para la presión del tanque principal

$f/dm^3$, presión media, cuando está entre 1,41 y 5,17$kg - f/dm^3$, y presión alta, cuando excede a este último valor. Para las otras variables también existen regiones de operación similares, las cuales se establecen a continuación:

- Nivel del tanque principal: tiene tres regiones de operación: nivel bajo, cuando la altura dle nivel del tanque es menor de 0.65 m., nivel medio cuando se encuentra entre 0.65 y 1.25m, y nivel alto cuando su altura supera los 1.25m.
- Nivel del tanque externo: también tiene tres regiones de operación: nivel bajo, medio y alto. Los valores que delimitan estas regiones son 1.5 y 2.2 m.
- Presión del tanque acostado, ya fue explicada anteriormente.
- Motobomba: encendida y apagada.
- Compresor: encendido y apagado.

Cada elemento del sistema tiene sus propias regiones de operación de forma independiente, de manera que cualquier combinación de estados pueden ocurrir. El cambio de un estado a otro implica la ocurrencia de un evento, aunque ciertos eventos no pueden ocurrir si alguna variable se encuentra en un estado dado. Por ejemplo, la motobomba apagada no permite que ocurra un ascenso en el nivel de agua que lleve de estado medio a alto, para el tanque principal. Una vez identificados las regiones de

operación del sistema, se puede plantear el sistema hidroneumático como un sistema de eventos discretos.

## B.2. Modelado de la lógica del proceso como un sistema de eventos discretos

Para ejecutar una aplicación de supervisión de un proceso es necesario conocer un modelo de la dinámica discreta del mismo. La presencia de condiciones de operación y de estados sugieren que el proceso es de naturaleza discreta, así existan variables continuas. Para describir esta dinámica discreta se pueden utilizar dos enfoques: o bien hacer una descripción por medio de autómatas o por medio de redes de Petri. Debido a la operación concurrente de componentes en este sistema, se escogió el método de las redes de Petri, ya que permite determinar el estado de cada subsistema independientemente de acuerdo a la posición de la marcación dentro de su grafo, y el diseño de un controlador para la planta no aumenta significativamente la complejidad de la red. Una representación basada en autómatas se puede encontrar en la publicación de Parra, Colina y Chacón sobre Control Supervisorio Inteligente [86].

La red de Petri que describe a los componentes del sistema se muestra en la figura B.3. Expresa el estado en que se pueden encontrar los recursos tales como la motobomba y el compresor (encendido, apagado), asociándolos a un lugar en la red. Así como el estado en que puedan encontrarse los niveles de agua y presión en los depósitos se asocia con otros lugares, describiendo así las etapas del proceso de control de nivel y presión. El significado de cada lugar y transición se describirá mas adelante.

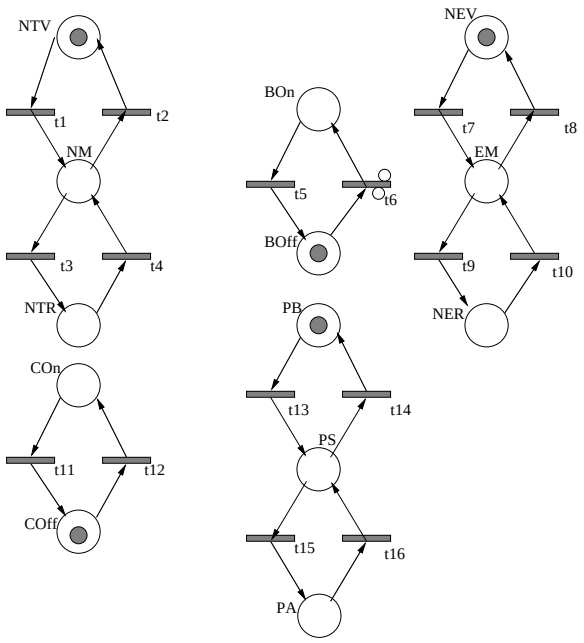


Figura B.3: Red de Petri que representa a los componentes del sistema hidroneumático

### B.2.1. Lugares y transiciones asociados al proceso.

Los lugares están asociados con el estado de los depósitos, bomba y compresor, mientras que las transiciones indican la ocurrencia de eventos que modifican un estado, o tareas que se llevan a cabo para alcanzar un estado determinado. Las siguientes tablas describen cada uno de ellos, respectivamente.

## B.3. Síntesis del Sistema Supervisor

Para el caso de ejemplo, las transiciones que indican cambio de estado de niveles y presión son *no controlables*, lo que dificulta aplicar el esquema de síntesis en redes de Petri, propuesto por Moody y Antsaklis [76]. De manera que es necesario un procedimiento que permita obtener un supervisor, y este consiste en apoyarse en la teoría de lenguajes formales, planteada por Wonham y Ramadge [93], donde una secuencia de eventos se considera un lenguaje formal. Un supervisor puede reaccionar de acuerdo a una secuencia determinada, para inducir eventos que produzcan una

Tabla B.1: Descripción de los lugares de la red de Petri del caso de estudio.

| Lugar | Descripción |
|---|---|
| NTV | Nivel de depósito principal inferior al mínimo |
| NM | Nivel del depósito principal medio |
| NTR | Nivel de depósito principal superior al máximo |
| BOn | Bomba encendida |
| BOff | Bomba apagada |
| NEV | Depósito externo con nivel por debajo el mínimo |
| ME | Nivel del depósito externo medio |
| NER | Depósito externo con nivel encima del máximo |
| PB | Presión del depósito interno baja |
| PS | Presión del depósito interno media |
| PA | Presión del depósito interno alta |
| COn | Compresor encendido |
| COff | Compresor apagado |

Tabla B.2: Descripción de las transiciones de la red de Petri que describen al sistema discreto.

| Tran. | Descripción |
|---|---|
| t1 | Nivel de depósito principal pasa a medio |
| t2 | Nivel de depósito principal pasa a bajo |
| t3 | Nivel de depósito principal pasa a alto |
| t4 | Nivel de depósito principal pasa a medio |
| t5 | Pasa de lugar BOn a BOff, apaga bomba |
| t6 | Pasa de lugar BOff a BOn, enciende bomba |
| t7 | Depósito externo en nivel medio |
| t8 | Depósito externo en nivel bajo |
| t9 | Depósito externo en nivel alto |
| t10 | Depósito externo en nivel medio |
| t12 | Se enciende el compresor |
| t13 | Presión pasa a estado medio |
| t14 | Presión pasa a estado bajo |
| t15 | Presión pasa a estado alto |
| t16 | Presión pasa a estado medio |

secuencia nueva, o mantenga una trayectoria de eventos.

### B.3.1. Síntesis basada en lenguajes de eventos

Para obtener un Supervisor que permita evitar condiciones que lleven a un estado no deseado es la de introducir las siguientes restricciones: cada vez que se dispare la transición 3, que indica la llegada al estado de rebosamiento en el depósito interno, se habilite el disparo de la transición 5 por medio del *lugar de control* $C1$, la cual abre la bomba de succión, y viceversa, con las transiciones 2 y 6, que se asocia a apagar la bomba por medio del lugar $C2$. Igual ocurre cuando se dispara la transición 15, que indica que la presión del depósito externo ascendió mas allá del máximo permitido, lo que lleva a habilitar la transición t11, que apaga al compresor, al igual que con las transiciones 14 y 12, para encenderlo. El sistema del tanque externo no está bajo control, pero conocer su estado interno posibilita tomar medidas para no bombear cuando su nivel desciende del mínimo. Para ello se recurre a un *arco inhibitorio* para impedir que se accione la bomba succionadora desde el lugar $C5$, activado por la transición 8, cuando el nivel vuelve a la normalidad, se activa el lugar $C6$ por medio de la transición 7, levantando la restricción sobre encender la bomba. En la figura B.4 se puede apreciar como queda el sistema discreto, una vez acoplada la acción del supervisor a los sistemas discretos que representan los recursos y el estado del proceso.

### B.3.2. Efectos del supervisor sobre la configuración del sistema.

El mecanismo de supervisión que hemos establecido para controlar al sistema hidroneumático tiene unos efectos en la configuración del equipamiento del sistema, donde cada configuración puede estar asociada a uno o más estados globales. En nuestro

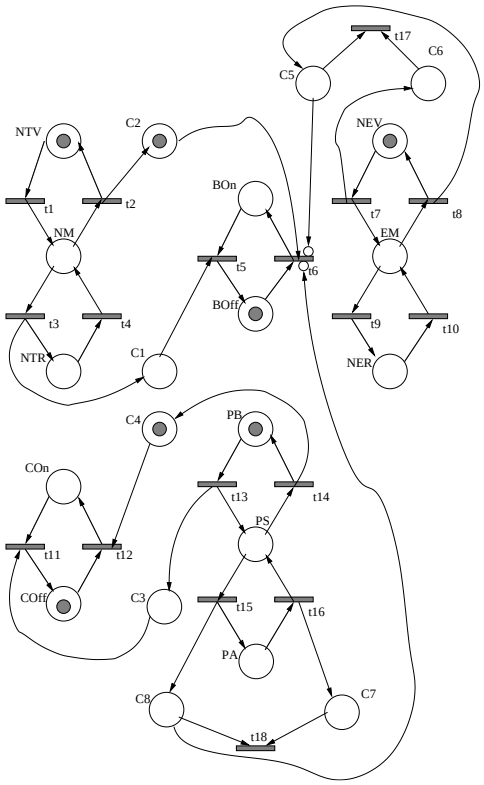


Figura B.4: Red de Petri del Sistema supervisado

Tabla B.3: Lugares que corresponden a la acción del Supervisor.

| Lugar | Acción del Supervisor |
|---|---|
| C1 | Apagar la motobomba cuando se rebasa nivel máximo del tanque principal |
| C2 | Encender la motobomba cuando se desciende mas allá del nivel mínimo |
| C3 | Apagar el compresor |
| C4 | Encender el compresor |
| C5 | Prevenir que no se encienda la bomba si no hay nivel en el tanque externo |
| C6 | Levantar la prevención de no encender la bomba |
| C7 | Prevenir que no se encienda la bomba si la presión es alta |
| C8 | Levantar la prevención de no encender la bomba debido a la presión |

caso de estudio los estados posibles en que puede encontrarse el sistema son finitos, pero debido a que existen varios componentes y estados posibles para cada uno de ellos, es difícil enumerarlos todos en una tabla. Para eso, es necesario establecer unas convenciones que permitan describirlos de una manera clara. El punto de partida para estas convenciones es el vector de marcación de la red de Petri que describe la dinámica del sistema, sin considerar a las medidas de control que tome el supervisor. Esto arroja el siguiente vector:

$$[\mathrm{L}_1\ L_2\ L_3\ L_4\ L_5\ L_6\ L_7\ L_8\ L_9\ L_{10}\ L_{11}\ L_{12}\ L_{13}]^T$$

Donde los tres primeros elementos del vector de marcación corresponden al nivel del depósito cilíndrico acostado, los elementos 4 y 5 corresponden al estado en que se encuentra la bomba, los elementos 6,7 y 8 describen el estado del tanque externo, los estados 9 y 10 corresponden al compresor y finalmente los últimos tres elementos los asociamos al estado de la presión en el tanque acostado. El valor de cada elemento puede ser $O$ o 1, de manera que habría cerca de $2^{13} = 8192$ estados posibles, aunque esta cantidad decrecerá cuando se consideren las características físicas de la planta. Por ejemplo, si se considera solamente el estado del nivel del tanque externo, el vector de marcación quedaría como $[L_1\ L_2\ L_3]^T$, pero la marca de la red (valor=1) solo podría estar en alguno de esos elementos únicamente, ya que así se describió la dinámica del estado del nivel del tanque externo: bajo nivel, nivel medio o alto nivel. De manera que los valores posibles serían [100], [010] y [001]. Igual razonamiento se aplica para el tanque externo y la presión del tanque acostado. La bomba y el compresor solo podrían tener como estados a [10] y [01]. Entonces habría un total de $3 \times 2 \times 3 \times 2 \times 3 = 108$ posibles estados de la planta, para cuatro configuraciones diferentes de la planta (estados de la bomba y compresor).

El estado del proceso y la configuración de los recursos del sistema hidroneumático se pueden etiquetar como $Q_{xyz}$, donde $x$ es un dígito que indica los estados del tanque acostado junto con la configuración de la bomba. El estado del tanque externo se

denota por $y$ y el dígito $z$ describe en que estado se encuentra la presión del tanque junto con la configuración de los compresores. La siguiente tabla muestra por ejemplo como serían los subíndices del estado $Q_x$, dado el vector de marcación que corresponde al nivel del tanque acostado y la bomba que controla ese nivel, junto con una breve descripción.

Tabla B.4: Estados que genera el nivel del tanque principal y la bomba.

| Marcación | Estado | Descripción |
|---|---|---|
| 1 0 0 0 1 | $Q_{0yz}$ | Nivel bajo y bomba apagada |
| 0 1 0 0 1 | $Q_{1yz}$ | Nivel medio y bomba apagada |
| 0 0 1 0 1 | $Q_{2yz}$ | Nivel alto y bomba apagada |
| 1 0 0 1 0 | $Q_{3yz}$ | Nivel bajo y bomba encendida |
| 0 1 0 1 0 | $Q_{4yz}$ | Nivel medio y bomba encendida |
| 0 0 1 1 0 | $Q_{5yz}$ | Nivel alto y bomba encendida |

La tabla B.4 sirve para explicar la numeración de los otros dos subíndices: 0,1 y 2 para describir el estado del nivel del tanque externo, y 0 a 5 para el estado de la presión en el tanque acostado y la configuración del compresor. Así, por ejemplo, un estado $Q_{223}$ se interpreta como: nivel de tanque acostado alto, bomba apagada, nivel de tanque externo alto, presión alta y compresor apagado.

Las entradas posibles que pueden ocurrir en el sistema para cambiar la configuración del equipamiento y así inducir eventos posteriores son cuatro: $X_1$ para encender la bomba, $X_2$ para apagarla, $X_3$ para encender el compresor y $X_4$ para apagarlo, los eventos inducidos son $t_5, t_6, t_{11}$ y $t_{12}$ respectivamente. Los demás eventos ocurren debido a cambios en las regiones de operación del sistema y por lo tanto deben ser detectados, ya que responden a la física de la planta, es decir, son eventos no controlables. Por otro lado, las transiciones $t_{17}$ y $t_{18}$ describen una acción que toma el supervisor para dejar de inhibir cambios en la configuración, por lo tanto no afectan la misma.

## B.4. Especificación de la conducta del Supervisor

Con el fin de implementar el comportamiento racional del Supervisor se utilizará tecnología de agentes, debido a que su naturaleza distribuida la hace adecuada para este tipo de aplicaciones. En nuestro caso particular se han identificado tres roles que pueden implementarse por medio de agentes: el *agente detector de eventos*, el *agente supervisor* y el *agente actuador*. El comportamiento de cada agente puede representarse por medio de un grafo dirigido con tres tipos de elementos: los hechos observables, los hechos ejecutables y los hechos desplegables, estos últimos llevan a otros hechos ejecutables o desplegables. La especificación de la conducta tiene reglas rectivas y proactivas. Las reglas de tipo *si ... entonces* sirven para especificar el razonamiento reactivo, o sea observar el entorno y deducir metas. Mientras que las reglas de tipo *para ... haga* representan los procedimientos para reducir esa meta a acciones ejecutables. Las siguientes reglas muestran un ejemplo de cada uno de este tipo de reglas.

```
1. si comando(X), X = 'abre bomba'
entonces abrir-bomba.
2. para abrir-bomba haga
enviar-señal, chequear-bomba, confirmar.
```

## B.5. Validación utilizando Modelado y Simulación

Para validar la conducta del sistema hidroneumático supervisado y gestionado por agentes, el cual combina dinámica de variables continuas con la dinámica discreta de sus condiciones de operación, se debe implementar su modelo matemático en una plataforma de simulación, para lo cual se escogió la plataforma Galatea [32]. Esta plataforma de simulación multi-agente implementa la especificación de un lenguaje de simulación basado en el formalismo DEVS y también permite interactuar con agentes,

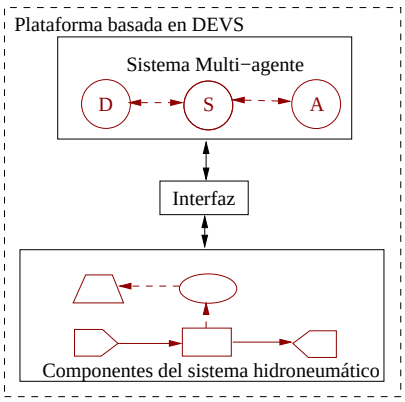


Figura B.5: Representación en red de nodos del sistema discreto y continuo

y está desarrollada en lenguaje Java. La figura B.5 muestra la red de componentes que representan a los procesos que tienen lugar en la planta, así como los agentes, donde estos últimos intervienen sobre los procesos por medio de una interfaz, intercambiando mensajes. En el caso de ejemplo, los mensajes que van a circular a través de la red son mensajes que indican la marcación en cada lugar de la red de Petri, así como comandos y resultados de observaciones sobre variables continuas.

### B.5.1. Condiciones iniciales.

Para ejecutar la simulación se establecieron las siguientes condiciones iniciales:

- Dimensiones del tanque externo: radio es 2.1 m, altura es de 4.4 m. Su nivel inicial es de 2.8 m.
- Tanque principal: radio es de 0.9 m, longitud es de 3.2 m, nivel inicial es de 0.5 m.
- Presión dentro del tanque acostado es de 2.067 $kg - m^2$.
- Flujo de salida por defecto es de 0.01 $m^3/seg$.
- Capacidad del compresor es de 2 $dm^3/seg$.

- Capacidad de la motobomba es de $0{,}1 m^3/seg$.
- Tanto el compresor como la motobomba inician apagados.
- Horizonte de simulación: 7200 unidades de tiempo (segundos).

La marcación inicial de la Red de Petri que representa a todo el proceso industrial, mas el estado interno del supervisor es:

$$M = [0\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^T$$

Los primeros trece elementos del vector corresponden al estado de los cinco componentes del sistema, descritos anteriormente. Las restantes ocho posiciones están corresponden al estado interno del supervisor.

### B.5.2. Análisis de resultados

Luego de ejecutar el modelo con las condiciones iniciales antes establecidas, y dada la especificación de la conducta de los agentes, se pudo hacer seguimiento a la ejecución de tres maneras: la traza de los valores de tiempo, nivel y presión, la marcación de la Red de Petri del proceso, y la traza de control del sistema.

**Traza de variables continuas.** Como salida de la simulación del modelo propuesto se analizaron dos variables: el nivel de líquido en el tanque acostado y la presión dentro del mismo. En la figura B.6 se puede apreciar como evoluciona el nivel de agua dentro del tanque, durante un intervalo de 600 segundos. La pendiente descendente corresponde a la demanda, donde la configuración del sistema indica que no ocurre bombeo, mientras que la ascendente corresponde a la configuración donde tiene lugar el bombeo. Los cambios de pendientes son consecuencia de la acción del agente supervisor y el actuador.

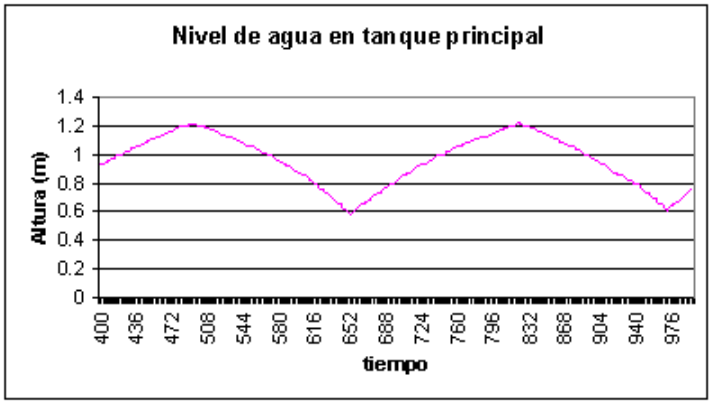


Figura B.6: Evolución del nivel de líquido en el tanque acostado

**Traza de las marcaciones de la Red de Petri.** Para evaluar el comportamiento de la lógica del sistema se recurre a la traza de las marcaciones en la red de Petri, las cuales modifica el agente supervisor cada vez que ocurre un evento. Dentro de esta traza también se indica el momento en que ocurre o se detecta una transición.

**Traza de configuración** Una manera compacta de observar como cambia la configuración del equipamiento cuando el proceso industrial evoluciona en el tiempo es la *traza de configuración*, la cual consiste en una sucesión de tuplas de tres elementos, $(N, Q, X)$ donde $N \in Z$ es el ordinal de eventos ocurridos, siendo 0 es el evento inicial y $n$ el evento final. $Q$ indica una configuración particular del sistema, que fue explicada en la sección anterior. $X = X_c \cup X_{nc}$ puede indicar la entrada que se aplica al sistema para inducir eventos posteriores (controlables) que generalmente se asocian a cambios en la configuración del sistema, o la ocurrencia de un evento no controlable debido a la física del proceso. La simulación a eventos discretos del sistema hidroneumático generó una traza de la configuración y estados del sistema. A través de sucesivas replicaciones, se encuentra que la secuencia es la misma, ya que el orden de los eventos y cambios en la configuración se mantiene, lo que cambia es el tiempo en que ocurre cada uno de estos eventos. Un fragmento de esta traza se puede ver a continuación:

Tabla B.5: Traza de marcaciones de red de Petri durante un intervalo pequeño.

| T. | $Q_x$ | $Q_y$ | $Q_z$ | Supervisor |
|---|---|---|---|---|
| 0 | 0 1 0 0 1 | 0 1 0 | 0 1 0 1 0 | 0 0 0 0 0 0 0 0 0 |
| 0 | Observé | Tr2 | | |
| 3 | 1 0 0 0 1 | 0 1 0 | 0 1 0 1 0 | 0 0 0 0 0 0 0 0 0 |
| 3 | Ejecuté | Tr6 | | |
| 4 | 1 0 0 1 0 | 0 1 0 | 0 1 0 1 0 | 0 0 0 0 0 0 0 0 0 |
| 30 | 1 0 0 1 0 | 0 1 0 | 0 1 0 1 0 | 0 0 0 0 0 0 0 0 0 |
| 30 | Observé | Tr1 | | |
| 30 | 0 1 0 1 0 | 0 1 0 | 0 1 0 1 0 | 0 0 0 0 0 0 0 0 0 |
| 171 | 0 1 0 1 0 | 0 1 0 | 0 1 0 1 0 | 0 0 0 0 0 0 0 0 0 |
| 171 | Observé | Tr15 | | |
| 171 | 0 1 0 1 0 | 0 1 0 | 0 1 0 0 1 | 0 0 0 0 0 0 0 0 1 |
| 180 | 0 1 0 1 0 | 0 1 0 | 0 1 0 0 1 | 0 0 0 0 0 0 0 0 1 |
| 180 | Observé | Tr3 | | |
| 180 | 0 0 1 1 0 | 0 1 0 | 0 1 0 0 1 | 0 0 0 0 0 0 0 0 1 |
| 183 | 0 0 1 1 0 | 0 1 0 | 0 1 0 0 1 | 1 0 0 0 0 0 0 0 1 |
| 183 | Ejecuté | Tr5 | | |
| 200 | 0 0 1 0 1 | 0 1 0 | 0 1 0 0 1 | 0 0 0 0 0 0 0 0 1 |
| 200 | Observé | Tr4 | | |
| 211 | 0 1 0 0 1 | 0 1 0 | 0 1 0 0 1 | 0 0 0 0 0 0 0 0 1 |
| 211 | Observé | Tr16 | | |
| 213 | 0 1 0 0 1 | 0 1 0 | 0 1 0 1 0 | 0 0 0 0 0 0 0 1 1 |
| 215 | 0 1 0 0 1 | 0 1 0 | 0 1 0 1 0 | 0 0 0 0 0 0 0 1 1 |
| 215 | Observé | Tr18 | | |
| ... | | | | |

$0Q_{414} \xrightarrow{t_2} 1Q_{314} \overset{X_2(t_6)}{\rightarrow} 2Q_{014} \xrightarrow{t_1} 3Q_{114} \xrightarrow{t_{15}}$
$4Q_{115} \xrightarrow{t_3} 5Q_{215} \overset{X_1(t_5)}{\rightarrow} 6Q_{515} \xrightarrow{t_4} 7Q_{415} \ldots$

Esta secuencia indica que en el evento inicial el nivel del tanque acostado era medio, bomba apagada, nivel de tanque externo medio, presión de tanque acostado media, compresor apagado, luego ocurre el evento $t_2$, ocasionando que el estado del sistema reflejara un nivel de tanque acostado bajo, manteniéndose las demás cualidades, luego del evento $t_1$ se genera la acción de control $X_2$ que lleva a su vez al evento $t_6$, donde cambia la configuración del sistema, estando bajo el nivel del tanque acostado y encendiendo el compresor, ...y así sucesivamente. Es importante destacar que a partir de esta traza se puede hacer un seguimiento a la configuración del equipamiento, que se manifiesta con las entradas $X_1$, $X_2$, $X_3$ y $X_4$.

# Anexo C

# Simulación de la logística de aprovisionamiento de un Central Azucarero concebido como un Sistema Holónico Supervisado

En este ejemplo se trata sobre el problema de aprovisionar un Central Azucarero (CAZ), en particular de lo complejo que puede resultar el asegurar un flujo discreto de insumos de manera que una planta procesadora de azúcar pueda operar de manera continua, produciendo lotes de azúcar. El flujo de insumos para la planta debe ser permanente, y de acuerdo a la capacidad nominal instalada de proceso. La figura C.1 muestra la cadena de valor asociada a este proceso, de acuerdo al concepto de Porter [91] sobre la cadena de valor en una organización. Tres aspectos fundamentales se pueden identificar en la figura C.1: recoger caña, procesar caña y distribuir azúcar. Cada uno de los componentes principales de esta cadena de valor tienen actividades complementarias, como lo es la facturación, mantenimiento industrial, entre otras. Para efectos del ejemplo, la actividad principal de recoger caña tiene a su vez tres

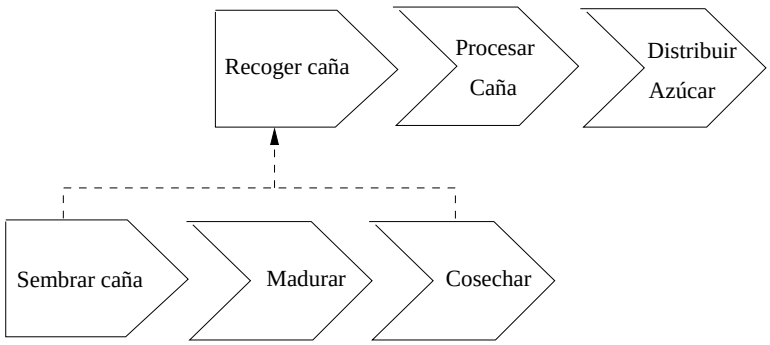


Figura C.1: Cadena de valor de una CAZ

actividades que son fundamentales: sembrar caña, madurar y cosechar. Un CAZ tiene a su disposición a un conjunto de granjas, las cuales están sembradas de caña y tienen una extensión variable. Para cosechar esta caña hay que esperar a que esté madura, y se asignen los recursos necesarios para su cosecha: máquinas cosechadoras y camiones transportadores, los cuales tienen un costo de uso asumido por cada granja. Dado que la planta industrial dentro de la CAZ una vez iniciada no puede detenerse a menos que incurra en costos inaceptables, se debe tener una cadencia de entrega de caña que posibilite que la planta esté en funcionamiento permanente.

## C.1. Especificación del problema

Dada una capacidad de procesamiento del CAZ de 700 ton/día, se requiere una cadencia de 290 toneladas de caña cortada cada hora. Puesto que un camión estándar puede transportar 40 toneladas de caña en un solo viaje, se requiere de un camión cargado entrando al CAZ cada ocho minutos, aproximadamente. El problema principal está en *como lograr esta cadencia*, dado que a) las granjas no maduran todas al mismo tiempo, lo que implica que no todas están disponibles para cosechar a la vez. b) la cantidad de cosechadoras y camiones asignados a cada granja producen cadencias distintas, dado que el tiempo de viaje de los camiones al CAZ son variables, c) la caña cortada tiene una cantidad finita de tiempo antes de descomponerse (ocho horas), de

modo que una vez cosechada necesariamente debe transportarse, y d) ningùn camión cargado con caña se sincroniza con los demás para llegar a la central, ya que salen de granjas diferentes. Por otro lado, no basta con lograr el flujo de entrada al CAZ, sino que también se requiere operar con mínimos costos, tanto para cada granja como para el CAZ.

Desde el punto de vista de la automatización industrial y el control, este es un problema de coordinación, asignación de recursos y de planificación. De manera que deben analizarse los siguientes escenarios:

1. Sin coordinación. Cada granja es una unidad independiente, la cual alquila sus cosechadoras y camiones, hasta un tope máximo, determinado por el flujo de caña que desee proporcionar, cada vez que haya madurado la caña.

2. Coordinación ejercida desde el CAZ, se asignan cosechadoras y camiones para obtener una cadencia constante, cada vez que en una finca madura la caña. Esta coordinación funciona de la misma manera como el control centralizado actúa sobre un proceso determinado.

3. Coordinación supervisada a partir de la interacción de *Holones*. Combinando los enfoques de automatización basados en *Sistemas holónicos* y en las teorías de control supervisorio, se propone diseñar un supervisor que asigne cosechadora y camión de acuerdo al potencial disponible y capacidades de transporte, previniendo asignaciones no deseadas. Es muy similar al anterior, solo que considera a los holones como unidades independientes las cuales son la granja, el frente de corte, el transporte y la planta procesadora.

Dada la gran cantidad de variables que describen un modelo de asignación de recursos para la operación de una CAZ, lo que dificulta resolver el problema planteado analíticamente, se va a utilizar modelado y simulación, de manera que cada experimento

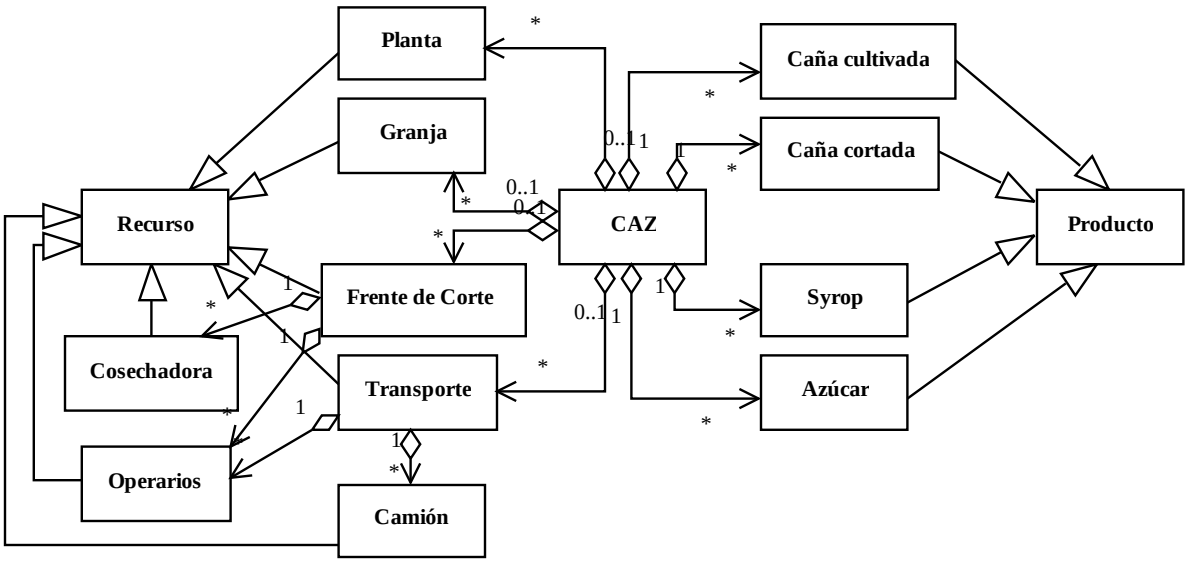


Figura C.2: Recursos de una Central Azucarera

de simulación refleje los escenarios planteados anteriormente, y ayude a optimizar un modelo de despacho óptimo que surta al CAZ con un flujo constante de insumos para su proceso.

## C.2. El Central Azucarero visto como un Sistema Holónico

Considerando el CAZ como un sistema holónico basado en la arquitectura de referencia PROSA [10, 6, 7], propuesta por Van Brussel *et al* en 1998, se establece la siguiente representación de los recursos del Central Azucarero utilizando un diagrama de clases, el cual se muestra en la figura C.2

Como se puede observar, los holones recurso con que cuenta el CAZ son de cuatro tipos: Granja, Frente de Corte, Planta y Transporte, los cuales son una especialización del holón Recurso en sí. El holón Frente de Corte cuenta a su vez con uno o varios

recursos como lo son la Cosechadora y el Recurso Humano. El Holón de Transporte tiene como recursos a los camiones y el recurso humano. Los productos que maneja el CAZ son la caña cultivada, caña cortada y syrop, este último es un subproducto antes de obtener el azúcar. La asignación de recursos para mantener al Holón Planta operando es una labor crítica, y en este documento se proporciona una solución basada en la simulación a eventos discretos para determinar el mejor escenario en que el Holón Planta reciba un flujo de caña constante. La caña en bruto es un producto que proporciona el holón granja, mientras que el holón de Frente de Corte proporciona caña cortada. La caña cortada es un recurso para el Holón Planta, el tiene la misión de producir azúcar.

La dinámica del sistema holónico se representa utilizando el enfoque de Chacón *et al* [18], basado en redes de Petri. Dado que el CAZ se compone de muchos holones granja, corte, transporte, y estos a su vez manejan recursos y diversas etapas de producción, se requiere una representación simplificada, para evitar una red de Petri difícil de tratar computacionalmente. Una representación que extrae una estructura básica para este sistema es la que se aprecia en la figura C.3, la cual representa el proceso de cosecha de caña y su transporte al CAZ. Esta estructura es válida para una sola granja, con unas etapas predefinidas. Los recursos de cosechadora y camión, que son parte del frente de corte y transporte, se representan como marcaciones en los lugares que corresponden a la actividad de los recursos. Esta estructura se repite y se agrega para las demás granjas. Un mecanismo de supervisión se requiere para poder coordinar toda esa cantidad de holones. Un enfoque para derivar supervisores se puede obtener de la teoría propuesta por [76], basado en redes de Petri.

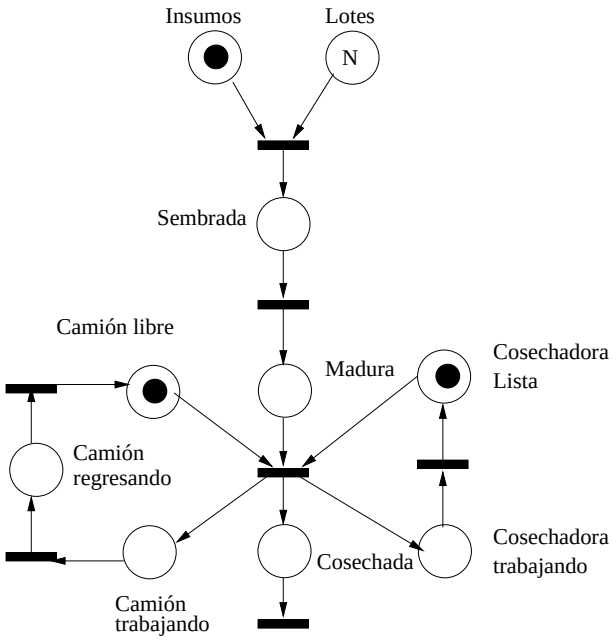


Figura C.3: Elemento estructurante de la dinámica del CAZ

## C.3. Simulación como un sistema a eventos discretos y análisis de resultados

Los supuestos del modelo son iguales para todos los escenarios, de manera que estos variarán solamente la manera en que se asignan los recursos. El lenguaje de simulación utilizado es Glider [31], el cual es un lenguaje orientado a la simulación mediante la construcción de una red de componentes que intercambian mensajes. La siguiente lista resume los supuestos para ejecutar las simulaciones, tomados algunos de ellos de fuentes sobre la explotación de caña [34] y [116]:

- Total de fincas a sembrar: 28
- Extensión de las fincas: se distribuye uniformemente entre 100 y 500 hectáreas
- Tiempo de siembra: 4 horas/hectárea
- Maduración: seis meses luego de finalizar la siembra. (4320 horas)

- Fin de disponibilidad: tres meses luego de madurar para la caña de corta maduración. (2160 horas)
- Potencial de cosecha: 64 toneladas de caña/hectárea
- Distancia a la central: uniformemente entre 20 y 200 kilómetros.
- Tipo de caña: un 40 por ciento de la cosecha es caña de corta maduración, la caña de maduración tardía es el 60 por ciento restante.
- Rendimiento de la cosechadora: 80 toneladas/hora
- Rendimiento del camión: 40 toneladas/viaje
- Cosechadora trabaja solamente si hay al menos un camión esperando por la caña a ser transportada.
- Cantidad de replicaciones: 8, para evitar sesgos en los resultados y analizar el promedio de las variables de salida.

### C.3.1. Potenciales de producción y fincas de caña

El potencial productor de las fincas de caña que pueden producir en un momento dado es válido para los tres escenarios, el resultado de toneladas de caña que están en condiciones de cosecharse. Como puede observarse, al inicio no todas las fincas están disponibles a la vez; solamente en un período breve de 50 días todas las fincas están disponibles para cosechar. Las fincas dejan de estar disponibles cuando la caña de maduración temprana se inunda, o cuando se agota la caña debido a la cosecha. Esta gráfica se comporta igual para todos los escenarios posibles. Se contabilizan las toneladas de caña que han madurado y están disponibles para cosechar. Se puede observar en el promedio de 8 replicaciones que a partir de las 7960 horas cierta

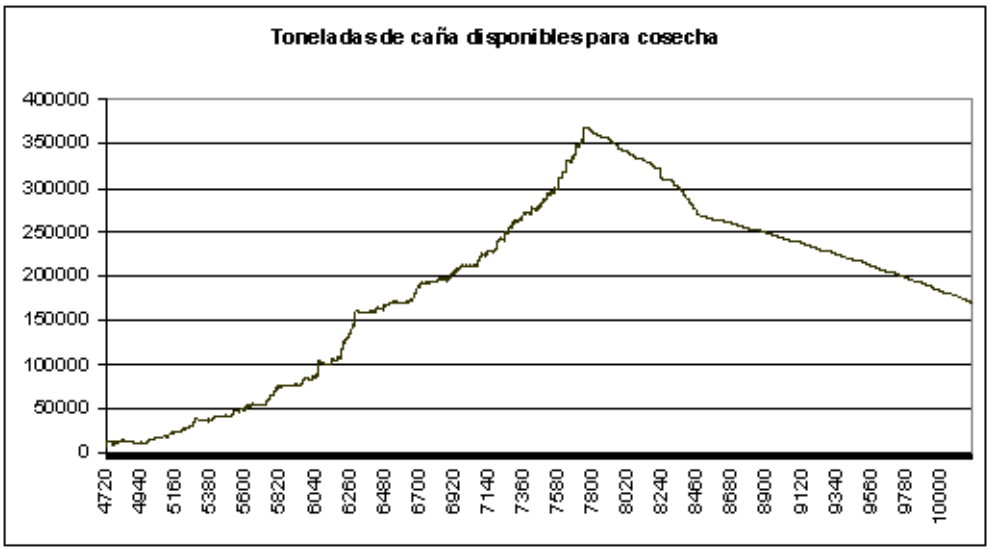


Figura C.4: Caña disponible para cosechar

cantidad de caña deja de estar disponible, debido a que las fincas de maduración temprana se inundan o dañan, dejando de estar disponible la caña que contienen.

El potencial de caña a cosechar depende de la cantidad de fincas disponibles para cosechar, y la cantidad de toneladas que quedan en cada finca, se puede apreciar en la figura C.4 el comportamiento de esta variable. Este potencial exhibe el mismo comportamiento para los tres escenarios considerados. Cuando existe re-siembra, el potencial de caña a cosechar va a estar oscilando entre unos valores mínimo y máximo.

### C.3.2. Capacidades de cosecha y transporte

Los potenciales en cuanto a la capacidad de cosecha y transporte se muestran a continuación, con gráficas comparativas que incluyen todos los escenarios. El comportamiento del potencial de cosecha se muestra en la figura C.5, para los escenarios sin coordinación, coordinación centralizada y distribuida (holónica). Para todos los escenarios la capacidad de cosecha nominal máxima es de 800 ton/hora, y las cosechadoras deben contar con el camión transportador al lado para poder ejecutar su trabajo y en ocasiones estos estarán realizando viajes hasta la Central Azucarera, por tal motivo

esta capacidad de cosecha disminuirá cuando las cosechadoras estén asignadas a una finca en particular, ya que no estarán disponibles para efectuar labores de cosecha. Así mismo, la capacidad de cosecha por asignar aumentará cuando las granjas se vayan agotando, ya que así liberan cosechadoras.

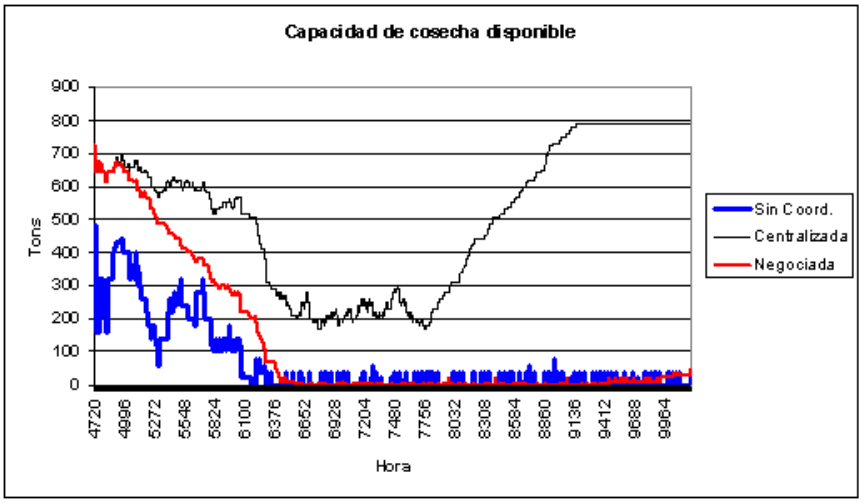


Figura C.5: Capacidad de cosecha para los tres escenarios de coordinación

Igual análisis se lleva a cabo para la capacidad de transporte. La cantidad de camiones ideal para una granja depende de las cosechadoras asignadas, ya que para mantener una cadencia de trabajo se requiere que haya entre dos y tres camiones por cosechadora asignada, ya que así permitirá que esta última trabaje, mientras los camiones van y regresan de la central. Por otro lado, asignar más de un camión a una cosechadora trae costos por espera ociosa, debido a que solamente un camión podrá cargar lo que arroja la cosechadora. Los demás camiones asignados deben esperar. Este costo de espera ociosa puede compensar el costo que implica traer un camión desde el CAZ o desde otra granja, cada vez que la cosechadora finalice un trabajo. El comportamiento de la capacidad de transporte puede apreciar en la figura C.6.

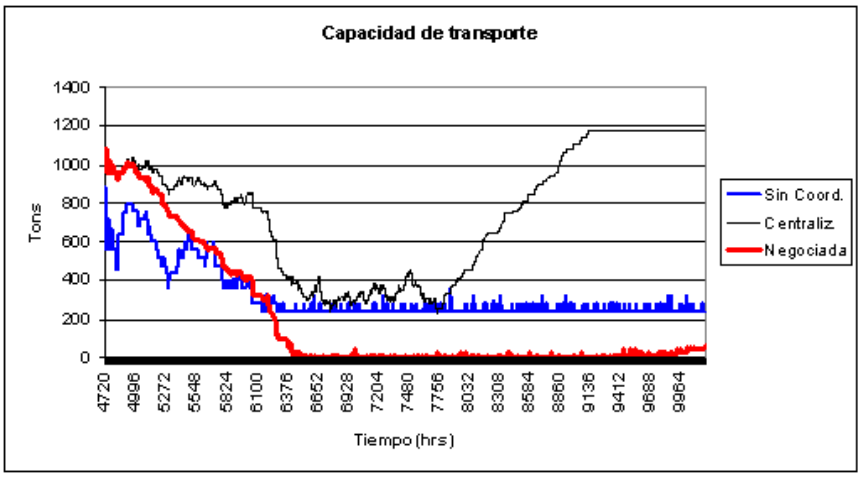


Figura C.6: Capacidad de transporte para los tres escenarios

### C.3.3. Tasas de cosecha diaria

La cosecha diaria de caña depende del escenario considerado. La figura C.7 contiene el comparativo de los tres escenarios. El escenario que corresponde a la ausencia de coordinación tiene un comportamiento muy oscilante al principio debido a que no todas las fincas están disponibles, fluctuando la tasa de cosecha entre 500 y 1800 ton/dia, luego se estabiliza cuando las fincas están en el potencial máximo, alrededor de 1950 ton/día. Las fluctuaciones se deben a que no todas las cosechadoras trabajarán las 24 horas, debido a que deben esperar que el transporte asignado lleve la caña a la central y regresen.

En el escenario de coordinación central, la cantidad cosechada inicialmente es baja debido a que apenas están madurando las fincas de caña. Posteriormente y cuando se utiliza en su potencial pleno las cosechadoras y el transporte, la cadencia de cosecha se estabiliza cerca de las 5000 ton/día, que es el valor que más se aproxima a la capacidad de proceso de la planta. Posteriormente esta cantidad disminuye debido a que algunas fincas pasan a inactividad o dejan de estar disponibles. Bajo el escenario de coordinación distribuida, se tiene que inicialmente la cantidad cosechada es baja, debido a que no todas las fincas están produciendo. Pero a medida que progresa la

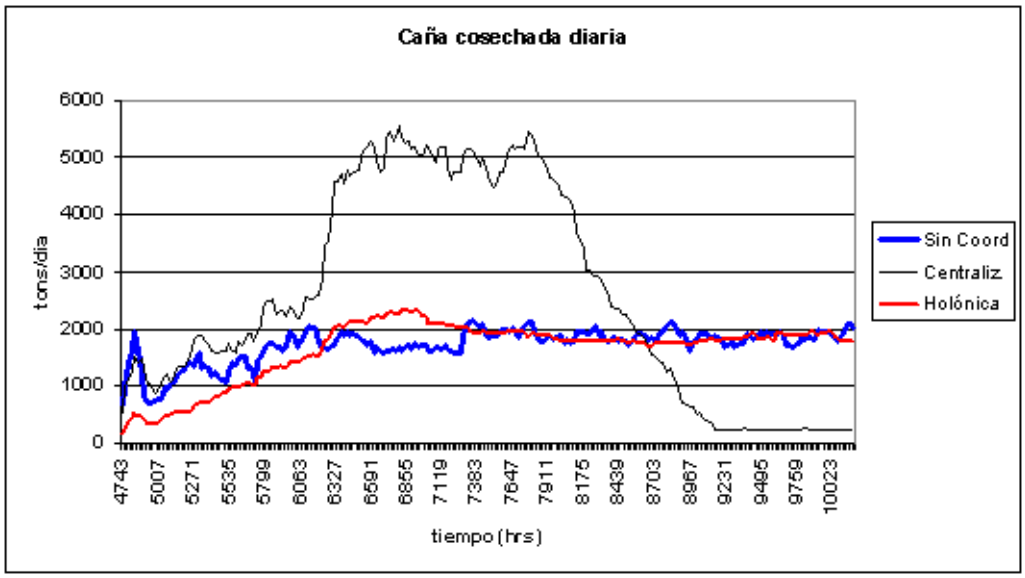


Figura C.7: Comparativo de caña cosechada por día

simulación esta cantidad se estabiliza en 2000 ton/dia hasta el final de simulación. Este escenario es el que presenta una menor variabilidad cuando se estabiliza el cosechado diario, ya que utiliza de manera más racional los recursos de cosecha y transporte.

### C.3.4. Costo monetario de operación

Otra de las maneras de evaluar el comportamiento de los diferentes enfoques para llevar a cabo la logística de aprovisionamiento para un Central Azucarero consiste en el análisis de costos. Para el modelo se consideraron diversos costos, como por ejemplo el costo de transportar, el costo de funcionamiento de cada una de las cosechadoras, costo de funcionamiento de la planta, costo de las granjas para utilizar los elementos de cosecha y transporte, entre otros. La tabla C.1 nos muestra como es el costo acumulado para utilizar recursos de transporte y cosecha, consolidado para todas las granjas.

Como puede observarse, en el escenario que corresponde a la coordinación holónica se observa el menor costo acumulado para las granjas, arrojando un total de 26059 dólares en promedio para cada granja, luego de ocho simulaciones.

Tabla C.1: Costo acumulado de utilización de recursos para todas las granjas

| Replicación | Sin coordinar | Centralizado | Holónico |
|---|---|---|---|
| 1 | 953360 | 4273689,5 | 812865 |
| 2 | 918160 | 4319119,5 | 417100 |
| 3 | 914360 | 2809720,5 | 863540 |
| 4 | 1070240 | 2754182 | 835380 |
| 5 | 977140 | 3100281,5 | 787375 |
| 6 | 949280 | 3311559 | 654680 |
| 7 | 1023840 | 1779151,5 | 736460 |
| 8 | 1000040 | 2433558,5 | 729615 |
| Promedio | 975802,5 | 3097657,75 | 729626,875 |
| Prom. por granja | 34850,08929 | 110630,6339 | 26058,10268 |

# Anexo D

# Ejemplo de aplicación del Modelo Multi-resolucional extendido a un problema industrial

Los Sistemas de Control Supervisorio (SCS) están diseñados para controlar a un alto nivel aquellos procesos en condiciones nominales, y en ocasiones no funcionan cuando ocurren fallas. Una propuesta de supervisión de un sistema de producción continua expuesto a fallas fue presentada por Parra, Colina y Chacón [86], de manera que no se detenga el funcionamiento de un sistema hidroneumático, a pesar de presentar fallas menores. Para lograr la especificación anteriormente mencionada se propuso un enfoque que combina los sistemas supervisores, métodos derivados de la Lógica Difusa para detectar eventos de falla, el modelo de razonamiento multiresolucional para obtener leyes de control, y los sistemas multi-agente para implementar los supervisores y detectores de eventos dentro de un entorno distribuido. En este anexo se detallará como es la aplicación del modelo multiresolucional para llevarlo a un esquema de supervisión inteligente. El sistema en el cual se basa esta aplicación es el sistema hidroneumático, que se describe en el anexo B.

## D.1. Modelo Numérico

Para describir el comportamiento físico del sistema se considerará como variables de estado el nivel de agua y la presión en el tanque cilíndrico acostado. Cada una de estas variables las podemos representar por medio de ecuaciones diferenciales ordinarias, las cuales se presentan a continuación:

El nivel del depósito lo podemos expresar con la siguiente ecuación:

$$\dot{h}(t) = \frac{\lambda(t) - \mu(t)}{r^2 \cos^{-1}(1 - \frac{h}{r}) - (r - h)\sqrt{2rh - h^2}}$$

Donde $\dot{h}(t)$ es la tasa de cambio del nivel de agua en el tanque y $h$ es la altura instantánea, en un instante determinado. El cambio de presión viene dado por la expresión:

$$P_1 = P_0(\frac{V_0}{V_1})$$

Donde $P_1$ es la presión en un instante $t$, $P_0$ es la presión en el instante $t = 0$, $V_1$ y $V_0$ son los volúmenes ocupados por el agua en los instantes $t$ y $t = 0$, respectivamente.

## D.2. Modelo Cualitativo Local

Este modelo describe los estados discretos en operación normal. Para lo cual se han definido nueve regiones de operación para las variables de nivel y presión en abscisas y ordenadas, como se pueden apreciar en la figura D.1. Estas regiones de operación se pueden proyectar a un autómata de estados finitos, como el que muestra la figura D.2.

Donde cada estado corresponde a una región de operación, y hay nueve estados. Las transiciones posibles son:

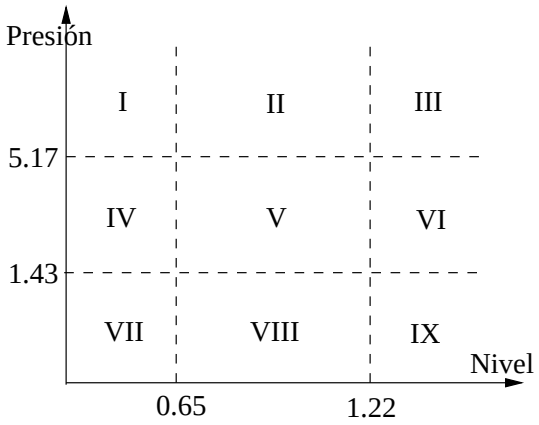


Figura D.1: Regiones de operación en condiciones normales

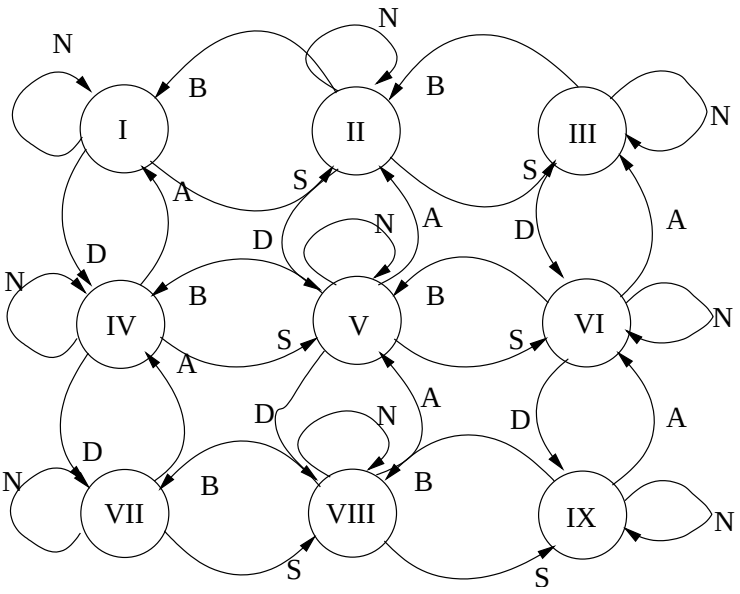


Figura D.2: Autómata que describe la operación normal del depósito

- N: Los cambios en los valores de las variables indica que no ocurrió evento alguno.
- S: Cambios en los valores de las variables indican que ocurrió un ascenso de nivel
- B: Los cambios en los valores de las variables indican que hubo un descenso de nivel
- A: La cualificación de las variables indica que subió la presión
- D: La evolución de las variables sugiere que la presión disminuyó, pasando de una región de mayor presión a una de menor presión.

Un cambio de región diferente al contemplado aquí indicará una posible ocurrencia de falla, como se mostrará más adelante.

### D.2.1. Determinación del estado de operación en que se encuentra el sistema de producción - función $Fa_{NC}$.

Para evitar cambios de estado innecesarios debido a la imprecisión de los dispositivos de medición o a efectos del oleaje dentro del depósito, se hace necesario un esquema basado en lógica difusa, que permita mapear de una forma más segura los valores numéricos resultado de las mediciones de variables hacia unas variables lingüísticas que representan a los estados del sistema (las regiones de operación). La figura 3.12, que se encuentra en el capítulo tres de este documento, muestra el esquema de difusificación para la variable presión, proyectada a tres valores: bajo, medio y alto.
La tabla D.1 resume la clasificación para deducir en que región de operación se encuentra el sistema, dados los valores cualitativos de nivel de agua y su presión.

Que expresado en forma de reglas, quedaría de la siguiente manera, por ejemplo:

Tabla D.1: Clasificación para determinar región de operación

| Presión / Nivel | Bajo | Normal | Alto |
|---|---|---|---|
| Bajo | VII | VIII | IX |
| Normal | IV | V | VI |
| Alto | I | II | III |

*Si presión es baja y nivel es bajo, rango de operación es VII*

*Si presión es normal y nivel es alto, rango de operación es II*

Y así sucesivamente, para las demás combinaciones de valores.

### D.2.2. Reglas para determinar la ocurrencia de eventos

Tienen la forma:

*Si rango de operaciones es V y la presión baja, hay cambio de región a VIII (Ocurrió Evento D)*

Exhaustivamente, para cada región de operación se pueden condensar estas reglas por medio de una tabla, como se muestra en la tabla D.2.

Tabla D.2: Reglas para detectar eventos

| Región | Cualidad de variable | Evento |
|---|---|---|
| I | Nivel bajo, presión alta | N (no evento) |
| I | Nivel medio, presión alta | S (sube nivel) |
| I | Nivel bajo, presión media | D (descenso presión) |
| I | Otros valores | F (posible fallo) |
| II | Nivel medio, presión alta | N |
| II | Nivel medio, presión media | D |
| II | Nivel bajo, presión alta | B (baja nivel) |
| II | Nivel alto, presión alta | S |
| II | Otros valores | F |
| ... | ... | ... |
| IX | Nivel alto, presión baja | N |
| IX | Nivel medio, presión baja | B |
| IX | Nivel alto, presión media | A (ascenso presión) |
| IX | Otros valores | F |

### D.2.3. Fallas que se pueden presentar.

Dado que el sistema hidroneumático se compone de dispositivos mecánicos, eléctricos y electrónicos de precisión, algunas fallas pueden presentarse, modificando su normal funcionamiento. Entre las más comunes tenemos:

- Falla en el compresor o existe un escape de aire (a pesar de estar encendido, la presión no aumenta, aún sin haber consumo de agua)
- Falla en la motobomba o en el ducto del depósito externo al principal, la bomba no enciende o no hay agua en el tanque externo (a pesar de estar encendida, el nivel sigue bajando).
- Falla en el sensor de nivel de agua o en el medidor de presión (lecturas perturbadas irregulares)

## D.3. Descripción del comportamiento global del sistema sujeto a fallos - Modelo Cualitativo Global

El sistema hidroneumático tiene un estado normal, donde operan todas las condiciones de operación que se han mencionado anteriormente. Cuando ocurre una falla, no es válido considerar las regiones de operación de la figura **??**, ya que cambiaron las condiciones del sistema. La dinámica global de este sistema sujeto a fallas se muestra en la figura 7, utilizando una máquina de estados finitos. El estado inicial es el que se etiqueta como "normal", donde es válido considerar las regiones de operación planteadas anteriormente, ciertos eventos detectados en ella permiten que se considere

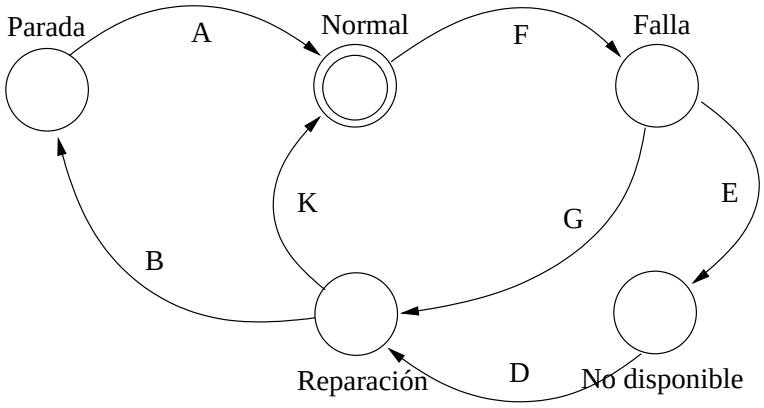


Figura D.3: Autómata para modelar el comportamiento global del sistema hidroneumático

el sistema global en estado normal, mientras que otros eventos sugieren que se ha presentado una condición de falla. Una vez ocurre el evento F (falla) el sistema entra al estado "Falla", y a partir de este estado pueden ocurrir otros dos eventos: la falla es muy grave, por lo que el sistema pasa a estado "no disponible", ocurriendo el evento E, o la falla puede repararse sin desmontar el sistema, indicando que ocurrió el evento G, que lleva al estado de reparación. Si este proceso de reparación es exitoso, el sistema entra a operación normal nuevamente, por medio del evento K. Ciertos procesos de reparación implican que el sistema se reinicie, por lo tanto se lleva a una condición de "parada", a partir del cual el sistema arranca (evento A) para volver a operación normal. Un ejemplo de arranque consiste en el reemplazo de una motobomba defectuosa, el depósito debe vaciarse para instalar la motobomba, y luego bombear agua y aire hasta que se coloquen en los valores de operación normal.

Aplicando los conceptos de control supervisorio utilizando teoría de lenguajes, hay que caracterizar el comportamiento del sistema a controlar definiendo para ello el lenguaje marcado del sistema. El lenguaje de este autómata es:

$$Lm(G) = (F(G + ED)K + BA))^*$$

Los eventos controlables son $K, A, D$ y $G$, mientras que los eventos $F, B$ y $E$ son no controlables. Cuando el sistema está en modo de operación normal, opera la máquina de estados finitos de la figura D.2.

## D.4. Cambio en la configuración del sistema de producción (Modelo R):

La configuración del sistema de producción la podemos establecer sobre los valores discretos de dos variables que corresponden a la motobomba y al compresor. Estos valores corresponden a 0 (apagado) y 1 (encendido). Esta configuración debe variar de acuerdo a la región de operación en que se encuentra el sistema, de acuerdo al último evento detectado, y el estado en que se encuentra la motobomba y el compresor. En general, opera en forma de reglas de tipo condición - acción, de las cuales se muestra un ejemplo.

*Si variable no se ha manipulado y región es 5, entonces no manipular*
*Si la variable no se ha manipulado, y la región es otra, manipular de acuerdo a la siguiente tabla:*

Tabla D.3: Modelo de Razonamiento para la configuración del sistema hidroneumático

| Región | B=Off | C=Off | B=On | C=On |
|---|---|---|---|---|
| I | - | - | - | Apagar C |
| II | - | - | - | Apagar C |
| III | - | - | Apagar B | Apagar C |
| IV | - | - | - | - |
| V | - | - | - | - |
| VI | - | - | - | - |
| VII | Encender B | - | - | - |
| VIII | - | - | - | - |
| IX | - | Encender C | - | - |

Donde "-" indica no manipular o no modificar la configuración del sistema, Apagar B

indica apagar la motobomba, Encender C indica que se debe encender el compresor, y así para los demás dispositivos y acciones.

Una vez se tiene este modelo racional, se derivan consignas para los controladores, según sea necesario encender o apagar tanto la bomba como el compresor, así como comandos de alto nivel para indicar que se ha producido una falla y que se requiere una etapa de reparación o inclusive una intervención humana para reinicializar el sistema.