Universidad Central “Marta Abreu “de Las Villas

Facultad de Ingeniería Eléctrica

Departamento de Automática y Sistemas Computacionales

Trabajo de Diploma

“Modelado de Sistemas de Automatización con el uso de *“UML”* y Redes de Petri: estado del arte”

Autor: Reinier Correa Pérez

Tutor: MsC. Ing. Robby Gustabello Cogle.

Santa Clara

2011

“Año 53 de la Revolución”

Universidad Central “Marta Abreu “de Las Villas

Facultad de Ingeniería Eléctrica

Departamento de Automática y Sistemas Computacionales

Trabajo de Diploma

“Modelado de Sistemas de Automatización con el uso de *“UML”* y Redes de Petri”

Autor: Reinier Correa Pérez

E-mail: rcorrea@uclv.edu.cu

Tutor: MsC. Ing. Robby Gustabello Cogle.

E-mail: robby@uclv.edu.cu

Santa Clara

2011

“Año 53 de la Revolución”

Hago constar que el presente trabajo de diploma fue realizado en la Universidad Central “Marta Abreu” de Las Villas como parte de la culminación de estudios de la especialidad de Ingeniería en Automática, autorizando a que el mismo sea utilizado por la Institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos, ni publicados sin autorización de la Universidad.

_____

Firma del Autor

Los abajo firmantes certificamos que el presente trabajo ha sido realizado según acuerdo de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del Autor Firma del Jefe de Departamento

_____

Firma del Responsable de Información Científico- Técnica

Pensamiento

¨En los momentos de crisis sólo la imaginación es más importante que el conocimiento¨

Albert Einstein

Dedicatoria

A mi abuela, mi hermana, mi padre, mis tíos, mi novia y mis amigos, por estar a mi lado durante este tiempo y brindarme su apoyo incondicional.

A mi madre, que a pesar de no estar a mi lado, me dio su apoyo siempre.

Agradecimientos

Especialmente quiero agradecer a mi tutor por su paciencia, tiempo y dedicación.

A todos mis amigos y a las personas que de una forma u otra hicieron posible la realización de este trabajo.

RESUMEN

Debido a la creciente complejidad que están experimentando los Sistemas de Automatización en nuestros tiempos, surge la necesidad de desarrollar métodos y herramientas más eficaces para lograr el diseño eficiente de estos. Es por eso que se han buscado vías para combinar aspectos del diseño Orientado a Objetos e Ingeniería de ¨*software*¨ con métodos formales de modelado y análisis. Una posibilidad de combinar estos aspectos la constituye integrar el Lenguaje Unificado de Modelado con las Redes de Petri.

A lo largo del trabajo se hace un recorrido por los Sistemas Automatizados en el ámbito industrial actual, resaltando sus principales características. De forma individual se exponen las particularidades para el diseño de cada herramienta y se presentan algunos ejemplos. Por último se analizan algunas metodologías a seguir para lograr una integración eficiente de estas herramientas en el modelado de Sistemas Automatizados. También se presentan algunos ejemplos reales y se determinan las principales ventajas y desventajas de esta fusión.

TABLA DE CONTENIDO

INTRODUCCIÓN

La automatización está estrechamente vinculada a casi todas las formas o alternativas que el hombre ha creado y crea para enfrentarse a las difíciles metas que se le imponen para lograr el desarrollo. Su alcance ha ido más allá de la simple mecanización de los procesos ya que ha brindado, a operadores humanos, mecanismos para asistirlos en los esfuerzos físicos del trabajo; además redujo ampliamente la necesidad sensorial y mental del hombre.

Es prácticamente imposible encontrar una industria o proceso que no presente cierto grado de automatización. Desde las tareas más sencillas tales como la regulación de la temperatura de un horno o el mando secuencial de una máquina herramienta, hasta las más complicadas como la dirección mediante computadoras personales y controladores lógicos programables de un planta electroquímica; que podría ser una central generadora de energía eléctrica.

Un aspecto importante de los Sistemas Automatizados modernos es su carácter distribuido o descentralizado, lo cual permite usar dispositivos digitales más simples y baratos como ¨*PC´s*¨[1] y ¨*PLC´s*¨[2]; sin embargo se requieren esfuerzos adicionales para manejar la coordinación entre ellos.

Recientemente se han comenzado a integrar los Sistemas de Automatización industrial con los sistemas empresariales, con el fin de elevar el nivel de flexibilidad, eficiencia y seguridad de todas las funciones. Este desarrollo está siendo acelerado por la necesidad de mantener una alta competitividad de las empresas frente al comercio electrónico globalizado. Sin embargo, el principal problema radica en que el diseño e implementación de este tipo de sistemas

[1] PC: ¨*Personal Computer*¨

[2] PLC: ¨*Programmable Logic Controler*¨

automatizados son tareas complejas que requieren herramientas y metodologías que formalicen su estructura y comportamiento.

Estas herramientas y metodologías deben ser capaces de describir el comportamiento de todos los niveles que componen el sistema, así como tratar de una manera integrada la modelación, análisis, validación e implementación. Para esto se deben usar formalismos de modelado que sean capaces de captar características como: concurrencia, paralelismo, operaciones asincrónicas, conflictos y compartimiento de recursos, las cuales son inherentes a la mayoría de los sistemas automatizados actuales. Es por eso que se propone utilizar de forma conjunta, como herramientas de modelado y análisis, al Lenguaje Unificado de Modelado (¨*Unified Modelling Language,* ***UML***¨) y a las Redes de Petri (RP).

Combinando las extraordinarias capacidades de diseño y especificación de ¨*UML*¨ y el poder de análisis de las RP en Sistemas de Eventos Discretos, se podría obtener una herramienta de modelado muy eficiente para sistemas de automatización complejos.

Por tanto el **objetivo principal** de esta investigación consiste en: determinar las potencialidades de *"UML"* y las Redes de Petri en el diseño de Sistemas de Automatización.

Para alcanzar el objetivo principal se plantean como **objetivos específicos**:

- Realizar una caracterización de los sistemas automatizados modernos en la industria actual
- Definir las particularidades para el diseño de ¨*UML*¨ y RP como herramientas independientes
- Analizar las especificidades y características que podría presentar el uso en conjunto de ambas para la automatización.

El Trabajo de Diploma se ha estructurado principalmente en: Resumen, Introducción, tres capítulos, Conclusiones y Recomendaciones.

El **Capítulo 1** se dedica a realizar una caracterización de los sistemas automatizados en el ámbito industrial actual.

En el **Capítulo 2** se presentan las principales particularidades de ¨*UML*¨ y las Redes de Petri por separado, y a las posibles aplicaciones de cada cual para la automatización,

Por último, en el **Capítulo 3** se presentan varias metodologías para integrar ¨*UML*¨ y las Redes de Petri. También se exponen algunos ejemplos de sistemas modelados y se determinan algunas de las potencialidades que presenta el uso de esta integración.

En las conclusiones finales puede apreciarse el cumplimiento de los objetivos trazados al inicio de esta investigación.

Se brindan, al final del informe, algunas recomendaciones de aspectos que pueden ser mejorados para continuar el desarrollo de esta temática.

Capítulo 1- Caracterización general de los Sistemas Automatizados

1.1- Conceptualización de Sistema Automatizado

El concepto de Automatización ha sido manejado por una gran variedad de científicos, ingenieros y técnicos de los diferentes tipos de industrias, llegando a ser incluso de cierto dominio por parte de aquellas personas no relacionadas directamente con el sector tecnológico, lo cual demuestra que ha tenido una amplia difusión, no solo por su aplicación en los procesos de fabricación, sino por estar prácticamente presente en el quehacer cotidiano. A continuación se presentan varios conceptos brindados por diferentes autores que han investigado sobre la materia, que de forma general abarcan todas las posibles vertientes que podría tomar un sistema automatizado en el ámbito industrial.

Según (Herrera and Rodriguez, 1990) un sistema automatizado **implica la supresión total o parcial del factor humano en la ejecución de las tareas, o sea la separación física del obrero del proceso de producción y la sustitución por máquinas de los procesos repetitivos**, exigiendo con ello que las capacidades intelectuales de los obreros se desarrollen, así como su preparación técnica y científica. Consiste en confiar a órganos tecnológicos todas o parte de las funciones intelectuales que realizaría un ser humano frente a un proceso productivo.

Otro punto de vista es el brindado en (Moreno, 1999) en el cual se plantea que **un sistema automatizado no solamente está relacionado con el proceso productivo o industrial en sí, sino también con la distribución de los productos fabricados y con la prestación de servicios. Forma parte integrante de la concepción y de la gestión de los grandes complejos industriales, administrativos y comerciales.**

De una forma más abarcadora, y según se plantea en (Castellanos, 2008), un sistema de automatización, en el contexto histórico actual, **incluye el conjunto de elementos (equipamiento, sistema de información y procedimientos) interrelacionados funcionalmente entre sí que conforman una estructura jerárquicamente expandida cuya función es garantizar el desempeño independiente del proceso a través de operaciones de control y supervisión total del sistema, bajo las técnicas más modernas y cumpliendo los requisitos establecidos de acuerdo al tipo de planta.**

En suma, analizando cada uno de los conceptos dados, se puede concluir que consiste en la incorporación a un proceso industrial, o conjunto de máquinas, de una serie de dispositivos tecnológicos con una lógica funcional determinada, que aseguren su control y buen comportamiento. Por lo que particularmente constituye uno de los factores de aumento de la productividad y de mejora de la calidad. En cuanto a su aspecto tecnológico, puede decirse que la automatización siempre ha sido un campo propicio para la aplicación de los más recientes avances científico - técnicos.

Cabe destacar que el desarrollo alcanzado en nuestros días en la esfera de la Automática no ha sido un fenómeno aislado, sino que siempre estuvo en constante interacción con los cambios cualitativos y cuantitativos que se produjeron en la industria tecnológica. A medida que se fue desarrollando la tecnología, se desarrollaba a la par la automatización, y viceversa.

1.2- Breve reseña histórica sobre los sistemas automatizados

Muchos artículos hacen referencia a la aparición de los primeros aparatos automáticos desde incluso antes de nuestra era. Existen innumerables ejemplos de mecanismos y dispositivos que fueron inventados desde aquella época con el objetivo de imitar las funciones que realizaban los seres humanos. Sin embargo con aplicaciones prácticas en la industria podemos hablar del surgimiento de la era moderna de la automatización en 1775 con la aparición de la máquina de

vapor de simple efecto creada por James Watt. Unos años más tarde, en 1784 se crea la máquina de doble efecto, la cual estaba provista de dos automatismos: el distribuidor de vapor y el regulador de bolas; que mantenían constante la velocidad del árbol de salida a pesar de las fluctuaciones de la carga (Moreno, 1999).

A finales del siglo XVIII y principios del XIX se desarrollaron algunas invenciones mecánicas ingeniosas, utilizadas fundamentalmente en la industria textil entre las que se destacan la hiladora giratoria de Hargreaves (1770), la hiladora mecánica de Cromptom (1779) y el telar mecánico de Cartwraight (1785), no obstante el invento más importante y que revolucionó este tipo de industria en aquel entonces fue el de Joseph Marie Jacquard que en 1801 recibió la patente por inventar un telar automático utilizando una cinta de papel perforada como un programa para las acciones de la máquina (Antonio Barrientos et al., 1997).

La automatización existió por muchos años en una escala pequeña, utilizando mecanismos simples para automatizar tareas sencillas de manufactura. Sin embargo el concepto solamente llegó a ser realmente práctico con la adición (y evolución) de las computadoras digitales, cuya flexibilidad permitió manejar cualquier clase de tarea. Por ejemplo en 1950 surge por primera vez la concepción de un sistema de control por computadora.

Las computadoras digitales con la combinación requerida de velocidad, poder de cómputo, precio y tamaño comienzan a aparecer en la década de 1960. Antes de ese tiempo, las computadoras industriales eran exclusivamente analógicas o híbridas (con algunos rasgos de digitalización) aunque, fuera de las industrias, algunas se aplicaban ya al control y monitoreo de vuelos (Castellanos, 2008).

Desde aquel entonces las computadoras digitales tomaron el control de la mayoría de las tareas simples, repetitivas, semiespecializadas y especializadas (con algunas excepciones notables en la producción e inspección de alimentos y fármacos).

Existen muchos ejemplos que avalan la anterior afirmación. Se pueden citar algunos como: el montaje del primer sistema para monitoreo en una planta generadora de potencia en el año 1958, cuyos elementos indicadores e interruptores se colocaron en una sala de control fuera del proceso. Un año más tarde se lleva a cabo la puesta en marcha del primer sistema en lazo cerrado en una refinería en Texas, el cual contaba con 40 lazos supervisores. En el año 1962 se realiza el primer Control Digital Directo en una planta química en el Reino Unido. A partir del año 1963 comienzan a aparecer en el mercado los Sistemas Operativos para tiempo real con superlenguaje los cuales, junto con el desarrollo en el área de los microprocesadores y microcomputadoras, condicionaron la aparición de los Sistemas de Control Distribuido con sus diferentes variantes y estructuras.

A partir de la década del ochenta, y condicionado por el constante avance en la esfera de la microelectrónica, los Sistemas Automatizados que utilizaban elementos de cómputo digitales se encontraban ya presentes en prácticamente todo tipo de industrias (automotriz, textil, electroenergética etc.), mayormente en los países desarrollados, trayendo consigo un elevado nivel de producción y calidad.

Actualmente estamos viviendo en la era de la digitalización, y los sistemas automatizados, luego de pasar por varias etapas en su proceso evolutivo, son considerados como los centinelas del correcto funcionamiento y control de la planta. Se han creado modernos centros de supervisión y control totalmente gobernados por ordenadores y con una alta integración, que son los encargados del monitoreo, planificación, calidad y eficiencia de la producción. Además con el empleo de instrumentación de campo inteligente, robots y sistemas digitales para la comunicación, se ha alcanzado un nivel de Automatización nunca antes soñado por el hombre. (Fig.1.1)

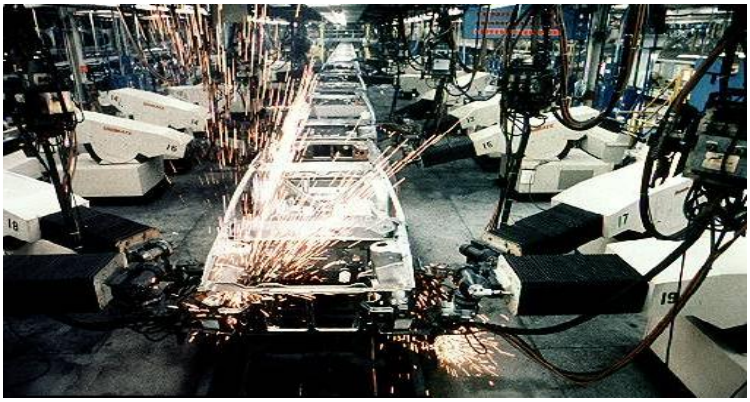


Figura 1.1- Línea de producción totalmente automatizada

1.3-Principales características de los Sistemas Automatizados

1.3.1-Partes estructurales

Muchos autores plantean que en cuanto a la estructura, un sistema de automatización industrial puede clasificarse en dos partes claramente diferenciadas: (Fig.1.2) (Castellanos, 2008, Moreno, 1999, Sarasola, 2003)

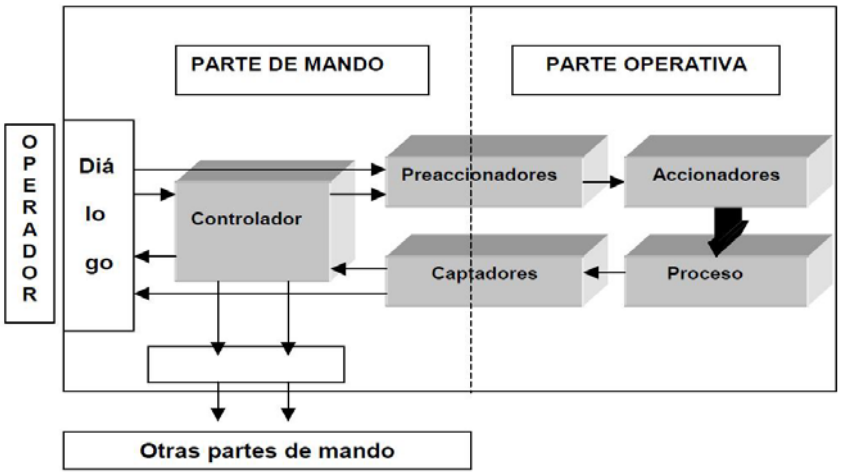


Figura 1.2- Esquema funcional de un Sistema Automatizado

- *Parte Operativa:* Es la parte que actúa directamente sobre el proceso tecnológico o planta, los dispositivos que hacen que la maquinaria se mueva y realice la operación deseada. Está compuesta principalmente por los medios técnicos de automatización como: elementos acondicionadores de señales, material de campo como son los

transductores y los captadores de información, los preaccionadores (relés, contactores magnéticos etc.) y accionadores o actuadores (válvulas, motores, órganos de desplazamiento lineal, cilindros, compresores, herramientas de corte etc.).

- *Parte de Mando:* Es la que emite las órdenes hacia la Parte Operativa y recibe las señales de retorno para coordinar sus acciones. Dentro de esta parte se encuentran las interfaces con los medios de cómputo (paneles gráficos, pantallas táctiles, monitores etc.), así como todo el aseguramiento matemático e informativo (métodos, algoritmos, programas, normas, bases de datos, ajustes de los controladores, toda la información útil sobre la planta o proceso). En el centro de la Parte de Mando está el "controlador" que coordina la información que a él converge. En los sistemas modernos suele ser un medio de cómputo (tecnología programada) como un ¨*PLC*¨ o una computadora digital; aunque hace unos años atrás se utilizaban relés electromagnéticos, tarjetas electrónicas y módulos lógicos neumáticos (tecnología cableada). Este elemento, en un sistema de fabricación automatizado debe ser capaz de comunicarse con todos los constituyentes del mismo

Como elemento integrante de ambas partes funcionales se encuentra sin duda alguna el factor humano, que tiene una importancia relevante dentro de la conformación del sistema, no solo por su papel protagónico en la creación y conformación del mismo, sino por la necesidad de su presencia a la hora de tomar las decisiones correctas en caso de ocurrencia de anomalías, además de ser el que proporciona el aseguramiento organizativo (conjunto de instrucciones y reglamentos para el personal de operación), mantenimiento y optimización de procesos.

Conjuntamente, todo sistema de automatización moderno debe presentar un esquema de comunicación que garantice el flujo fiable de datos en tiempo real, tanto vertical como horizontal entre todos los elementos que lo conforman, así

como presentar una alta capacidad de adaptación a las características del proceso para futuras aplicaciones (automatización flexible). Esto lleva implícito la incorporación de una gran cantidad de funciones implementadas que abarcan la explotación del sistema en las esferas de: control, monitoreo, ingeniería (simulación y parametrización), diagnóstico y ayuda, administración, entre otras (Castellanos, 2008). Los más modernos incluyen funciones como: administración de la producción, documentación y medios, seguridad y control crítico avanzado, instrumentación inteligente y centro de control de motores (O'Brian, 2007).

Los medios de cómputo en las instalaciones industriales se encargan de realizar los cálculos de balances de energía o de materia, de la vigilancia de las magnitudes que puedan llegar a adquirir valores peligrosos, del comportamiento secuencial del arranque y de la detención del proceso, así como llevar a cabo complejos cálculos de auto-adaptación y auto-optimización (Moreno, 1999). El empleo de estos medios y además de las modernas interfaces que traen consigo, ha permitido un aumento en la velocidad de realización de los cálculos, ha posibilitado además disponer de mayor cantidad de memoria para el almacenamiento y procesamiento de los datos, así como una mayor calidad en la representación de la información y visualización de parámetros. Todo esto garantiza que se cumplan de forma eficiente los objetivos y funciones para los cuales se diseña el sistema y además una mejor operación y explotación general del mismo.

1.3.2-Objetivos y funciones

El diseño, montaje y puesta en marcha de los sistemas automatizados persigue varios objetivos importantes relacionados con la obtención de mejoras de forma general en el proceso productivo. Entre ellos se encuentran: integrar varios aspectos de las operaciones de manufactura para mejorar la calidad y uniformidad del producto, minimizar el esfuerzo y los tiempos de producción, mejorar la productividad reduciendo los costos de manufactura mediante un mejor control de la producción, reducir la intervención y posibilidad de error del

hombre, reducir el daño en las piezas que resultaría del manejo manual, aumentar la seguridad para el personal y ahorrar área en la planta (Coca, 2007). La planta o proceso, ante cualquier condición operativa y manteniéndose siempre bajo los requisitos de comportamiento deseados, debe presentar siempre una alta disponibilidad y elevados márgenes de seguridad, confiabilidad y eficiencia; no solo para el personal, sino también para el equipamiento tecnológico.

Sin embargo, la principal razón de automatizar es el incremento de la productividad. Ello se logra racionalizando las materia primas e insumos, reduciendo los costos operativos, reduciendo el consumo energético, incrementando la seguridad de los procesos, optimizando el recurso humano de la empresa y mejorando el diagnostico, supervisión y control de la calidad de la producción.(Velásquez, 2004)

Entre las principales funciones de un Sistema Automatizado encontramos las siguientes: (Castellanos, 2008)

1. **Funciones de Dirección** : Las funciones de dirección son imprescindibles y de vital importancia para lograr el correcto funcionamiento del sistema, entre las más importantes se encuentran: establecimiento de los regímenes de operación del proceso, formación y transmisión de señales hacia los dispositivos de mando, elaboración de secuencias de órdenes para el arranque y parada, atención a condiciones anormales, establecimiento de prioridades funcionales entre los elementos del sistema y con otros sistemas a diferentes niveles.
2. **Funciones de Procesamiento y Control de la Planta**: Comprende las acciones que ejecuta el sistema automático desde el punto de vista del control propiamente dicho como: adquisición y validación de los datos, linealización y calibración de las mediciones, manipulación aritmética y lógica de los datos, corrección de resultados, ajuste automático y conversión de escalas y a unidades de ingeniería.

3. **Funciones de Comunicación**: Entre las más importantes encontramos: comunicación entre los elementos de la planta, comunicación entre el proceso y el hombre y la comunicación entre diferentes subsistemas en los posibles modos de operación que tenga la planta.
4. **Funciones informativas – computacionales**: Son aquellas orientadas a la presentación de la información y reportes al operador del proceso, entre ellas están: indicación de las variables y parámetros del proceso, detección de estados y bloqueos, reconocimiento y tratamiento de las condiciones de alarma, diagnóstico y pronóstico del proceso. Estas funciones son de gran utilidad para la realización en el momento adecuado de los mantenimientos, así como para la preparación de la información e intercambio con otros niveles jerárquicos de procesamiento.

En esta última función se hace referencia a los niveles jerárquicos de procesamiento, que son en los que se organiza un sistema automatizado atendiendo a las características y funciones de los elementos que lo integran. Estos niveles son: (Castellanos, 2008, Vega, 2007) (Fig.1.3)

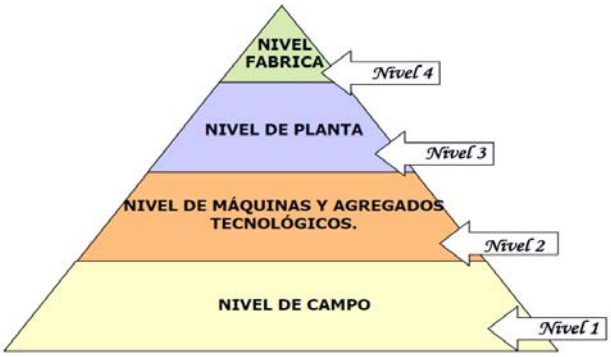


Figura 1.3- Niveles jerárquicos de procesamiento

1. Nivel de Campo: Está compuesto por sensores, transductores, motores, válvulas, actuadores, elementos de control final y pequeños ¨*PLC´s*¨. La función básica de este nivel es el control del equipamiento mediante lazos simples.

2. Nivel de Célula o de Máquinas: En este nivel se encuentran las líneas de producción, ensamblaje de máquinas, medianos ¨*PLC´s*¨, entre otros. Como función principal tiene el mando centralizado de máquinas y procesos.
3. Nivel de Supervisión o de Planta: Compuesto por grandes controladores, módulos de E/S, redes de *PLC´s* y *PC´s*, programas de supervisión y control. Su función principal es el control y gestión de la producción.
4. Nivel de Coordinación o Fábrica: Está compuesto por grandes controladores y *PC´s* operando en red, y en función del control de la fábrica o empresa en su conjunto. Su función principal es establecer los planes y estrategias de producción y mercado.

Cabe destacar que el cumplimiento eficiente o no de las anteriores funciones y objetivos está muchas veces condicionado por el grado de uso de los medios técnicos de automatización digital, y de la conformación del sistema digital adecuado según sean los requisitos de control de la planta.

1.3.3-Clasificación de los Sistemas Automatizados digitales

El empleo de los microprocesadores y microcomputadoras en la adquisición de datos y automatización, trajo como consecuencia que se fueran concibiendo de forma paulatina los sistemas digitales descentralizados. Estos sistemas están compuestos en su mayoría por módulos inteligentes para la adquisición, procesamiento de datos y actuación sobre el proceso.

En (Castellanos, 2008) aparecen las principales categorías en las que se agrupan los Sistemas Digitales Automatizados.

- Sistemas de Control Distribuido **(SCD)**.
- Sistemas ¨***SCADA***¨ (¨*Supervisory Control and Data Acquisition*¨).
- Sistemas a base ¨***PLC´s***¨ y ¨***PC´s***¨**.**

Se considera necesario mencionar otras clasificaciones que aunque en ocasiones podrían encontrarse incluidas en las tres primeras, en gran variedad de procesos sobresalen por sus propias características, estas son:

- Redes Neuronales Artificiales (¨*Artificial Neural Network,* ***ANN*** ¨)
- Interfaces Hombre-Máquina (¨*Human-Machine Interface,* ***HMI*** ¨)

1.3.3.1- Sistemas de Control Distribuidos

Un sistema de control distribuido (Fig.1.4) es el conjunto de elementos de hardware y paquetes de software que de forma conjunta permiten lograr toda la funcionalidad requerida para realizar el control y la adquisición de datos de una planta o proceso dado.

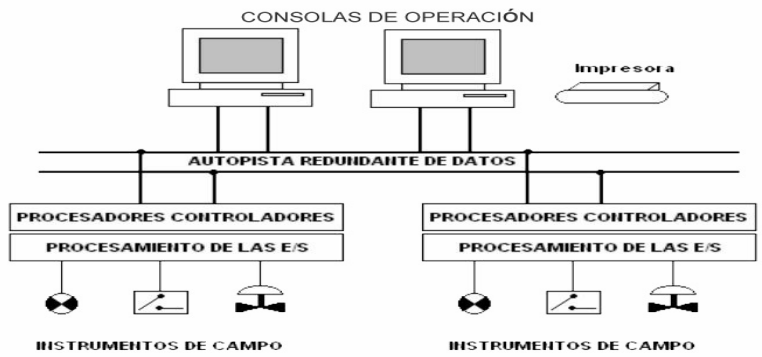


Figura 1.4- Esquema de un SCD.

Surgieron producto del propio desarrollo alcanzado por los microprocesadores en la mitad de la década de los 70, donde aparecen los primeros SCD basados en microprocesadores que fueron concebidos para desplazar los paneles de instrumentación electrónicos convencionales por paneles conformados por sistemas digitales, donde existía una mejor compactación y representación de la información, empleo de vídeo terminales, funciones compartidas por controladores y elementos que permitían de una manera más fácil realizar las funciones de regulación, estrategias de control, secuencias control, temporización y conteo (Castellanos, 2008).

En este tipo de sistemas los controladores no se encuentran centralizados en un nodo, sino que tienden a estar dispersos de acuerdo a la geografía de la región y las necesidades de monitoreo del sistema, además cada controlador está conectado a los otros a través de una red de comunicaciones. Este tipo de sistema es típicamente usado en procesos de fabricación, especialmente cuando la acción o producción es de régimen continuo, por ejemplo en los procesos de refinamiento de petróleo o en el control de una sección de una estación generadora de energía eléctrica (Xusheng Yang et al., 2006).

1.3.3.2- Sistemas "SCADA"

Un **SCADA** (Fig.1.5) consiste en un software de aplicación diseñado especialmente para ejecutarse sobre ordenadores destinados al control de la producción. Proporciona comunicación con los dispositivos de campo (controladores digitales autónomos, autómatas programables, instrumentación inteligente, etc.) y controla el proceso de forma automática desde la pantalla del ordenador (Castellanos, 2008).

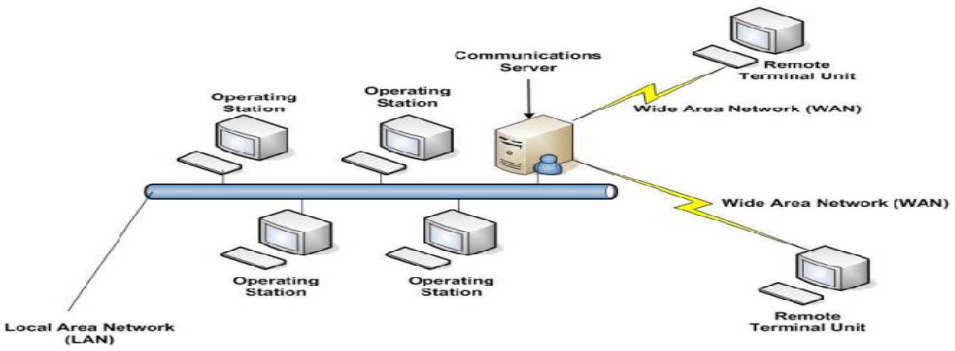


Figura 1.5- Esquema de un sistema "*SCADA*"

El mismo permite realizar a distancia operaciones de control, supervisión y registro de datos del proceso industrial. De esta manera, provee de toda la información que se genera en el proceso productivo a diversos usuarios, tanto desde el propio nivel de campo como de otros niveles superiores que pueden llegar hasta nivel de empresa.

Estos sistemas mejoran la eficacia del proceso de monitoreo y control proporcionando la información oportuna para poder tomar decisiones operacionales correctas. Ya que recibe información del proceso de primera mano (alarmas, históricos, paradas), permite la integración con otras herramientas como lo son las bases de datos y estadísticas del proceso. Los "*SCADA*" pueden trabajar juntos para controlar una industria completa, e incluso una región industrial entera, mediante la interconexión de las redes de comunicación de cada planta o sector productivo (Xusheng Yang et al., 2006) permitiendo automatizar procesos hasta niveles insospechados por el propio cliente.

De igual manera los sistemas *SCADA* permiten generar planes de mantenimiento y eficaces procedimientos de actuación para los operadores de la planta, facilitando el trabajo de todo el personal.

1.3.3.3- Sistemas a base de "PLC's" y "PC's"

En general, un sistema a base de "***PLC´s***" y "***PC"*** está formado por dispositivos autónomos inteligentes que cooperan con objetivos concretos. Se destaca la enorme importancia que juega el sistema de comunicación como soporte de las estrategias de automatización. En estas redes de comunicación modernas está fuertemente desarrollado el empleo de los buses de campo y los sensores y actuadores inteligentes, lo cual ha permitido que las redes con "*PLC´s*" y computadoras personales trabajen sobre los modelos de capas y soporten la utilización de estas técnicas en módulos especiales de acoplamiento cuyo hardware está muy a tono con el desarrollo científico-técnico actual en la rama.

Los "*PLC´s*", en la mayoría de los casos, están diseñados específicamente para ser empleados en ambientes industriales exigentes y han sido continuamente desarrollados de forma que sus sistemas operativos en tiempo real constituyan su mayor virtud y ventaja. Por otra parte las redes de autómatas descargan la

información del proceso en "*PC´s*" destinadas para ello (calidad industrial), en las cuales se procesan y visualizan los datos (Castellanos, 2008).

1.3.3.4- Redes Neuronales Artificiales

Una "***ANN***" es un modelo matemático o computacional, un paradigma de aprendizaje y procesamiento automático con un carácter distribuido, inspirado en la forma en que funciona el sistema nervioso de los animales. Se trata de un sistema de interconexión de neuronas en una red que colabora para producir un estímulo de salida. Una de las misiones en una red neuronal consiste en simular las propiedades observadas en los sistemas neuronales biológicos a través de modelos matemáticos recreados mediante mecanismos artificiales (como un circuito integrado, un ordenador o un conjunto de válvulas). El objetivo es conseguir que las máquinas den respuestas similares a las que es capaz de dar el cerebro, que se caracterizan por su generalización y su robustez. Las características de las *ANN* las hacen bastante apropiadas para aplicaciones en las que no se dispone a priori de un modelo identificable que pueda ser programado, pero se dispone de un conjunto básico de ejemplos de entrada (previamente clasificados o no). Asimismo, son altamente robustas tanto al ruido como a la disfunción de elementos concretos y son fácilmente paralelizables. Esto incluye problemas de clasificación y reconocimiento de patrones de voz y ruidos, reconocimiento óptico de caracteres e imágenes (sistemas de visión inteligentes), señales y control de robots. Asimismo se han utilizado para encontrar patrones de fraude económico, hacer predicciones de mercado, entre otras. (Luebbers, 2004)

1.3.3.5- Interfaces Hombre -Máquina

Las "***HMI***", comúnmente referidas como interfaz de usuario u operador, constituyen la principal vía para introducir la acción humana sobre el proceso. El usuario proporcionará la entrada y el sistema a su vez, dará una salida que coincida con el intento de entrada. Aunque las "*HMI*" forman parte constituyente

de otros sistemas digitales como los "*SCADA´s*", muchas veces nos encontramos secciones de planta controladas directamente por pantallas y paneles que no se encuentran asociadas al sistema principal. El "*software*" "*HMI*" debe ser estrechamente integrado a una plataforma de "*hardware*" que soporte el ambiente industrial en el cual se instalará. Este "*hardware*" generalmente funciona en un ambiente de sub-estación, lo más común es que sea en "*PC´s*" personales comerciales, computadoras industriales, sub-estaciones de computadoras y servidores. El sistema "*HMI*" además cuenta con otros requerimientos de "*hardware*" como: múltiples puertos serie, puertos de red, debe tener al menos un puerto IRIG-B para la sincronización de tiempo, armarios para paneles, mouse, teclados etc. (IEEE, 2007)

Todos los sistemas digitales anteriores necesitan pasar por una etapa de diseño, en la que no solo se conforma el sistema en sí, sino que también se analiza mediante el uso de herramientas de modelado y planificación, el impacto que podría tener sobre el proceso.

1.4-Herramientas empleadas en la automatización.

Con el uso de la computadora en la industria tecnológica desde mediados del siglo XX, se han creado y desarrollado una gran cantidad de herramientas de software que han ayudado a los ingenieros y especialistas en la rama de la automatización en el modelado, simulación y diseño de sistemas.

Para realizar la automatización de una planta es necesario utilizar herramientas de cómputo que permitan planear todas las actividades en que se divide el proyecto, asignándole los recursos necesarios (humanos, técnicos, materiales y económicos). Estas herramientas son las llamadas "***ERP"*** (¨*Enterprise Resource Planning*¨). Estos sistemas tienen la característica de que están interrelacionados entre sí, es decir, son un sistema integral de información que abarca todas las áreas de una organización, desde la gestión financiera hasta la producción de insumos (Gómez, 2010). Los "*ERP*" están funcionando

ampliamente en todo tipo de empresas modernas. Todos los departamentos funcionales que están involucrados en la operación o producción están integrados en un solo sistema. Además, abarca áreas como: almacenamiento, logística e información tecnológica, contabilidad, recursos humanos y herramientas de mercadotecnia y administración estratégica. Algunas de las herramientas ¨*ERP*¨ más usadas son el *Adempiere*, el *Compiere* y el *FacturaLUX*, todos basados en software libre. Vale destacar, que a pesar de sus potencialidades, las ¨*ERP*¨ no son herramientas ideales para modelar el funcionamiento de Sistemas Automatizados.

Otro tipo de herramientas usadas son los **Diagramas de Flujo de Datos** y **Diagramas de Bloques Funcionales**, empleados en varios programas de diseño. Permiten mostrar gráficamente y de manera general el funcionamiento de cada componente del sistema, el flujo de información y los procesos necesarios para su desarrollo. Además cada bloque puede contener información relacionada con el comportamiento dinámico de las partes y no incluir información de la construcción física de las mismas. Son herramientas capaces de describir, analizar y manejar el movimiento de los datos a través de un sistema.

Una de las herramientas que más se usa actualmente para modelar es el Lenguaje Unificado de Modelado. El ¨***UML***¨ cuenta con un conjunto amplio de herramientas que nos permite modelar el mundo siguiendo un paradigma de Orientación a Objetos. Sirve para el modelado completo de sistemas complejos, tanto de ¨*software*¨ como para la arquitectura de ¨*hardware*¨ donde se ejecuten, aporta mayor rigor en la especificación y permite realizar la verificación y validación del modelo obtenido. Para el diseño en "*UML*" se utilizan software como *ArgoUML*, *Umbrello*, *BOUML* (genera código C++, Java e IDL), *StarUML* (desarrollada en Delphi, bastante estable y utilizada) y *Poseidon for UML*.

Otra muy empleada es el **GRAFCET** (Gráfico Funcional de Control de Etapas y Transiciones), el cual surgió debido a la necesidad de emplear una metodología

clara para la descripción y diseño de automatismos que fueran independientes de la tecnología a utilizar (David and Alla, 1992). Se emplea para la descripción, control y modelado de procesos secuenciales de una forma gráfica, además de ofrecer una metodología de programación estructurada de forma descendente, que permite el desarrollo conceptual de lo general a lo particular. No fue concebido como un lenguaje de programación de autómatas, sino un tipo de Grafo para elaborar el modelo pensando en la ejecución directa del automatismo. En la actualidad no tiene una amplia difusión como lenguaje de programación, pero se ha universalizado como herramienta de modelado que permite el paso directo a esta (Moreno, 1999).

Es necesario destacar que los trabajos de investigación que han precedido algunos métodos de modelado y diseño como el GRAFCET fueron los de P. Giraud que introdujo los conceptos de receptividad y etapa, y los del alemán Karl Adam Petri que dio origen a las llamadas **Redes de Petri**. Estas redes han sido usadas para la modelación y la evaluación de todo tipo de Sistemas de Eventos Discretos, sistemas de fabricación flexibles, arquitecturas multiprocesador, sistemas de comunicación, y también para la escritura de programas concurrentes, eficientes y confiables. La facilidad del modelado conceptual, así como la forma natural de notación gráfica de las RP, permite que este formalismo sea elegido para el desarrollo de muchas aplicaciones. Aplicada de forma metodológica, podría garantizar el objetivo de conseguir una representación fiable de la planta y de este modo minimizar los fallos que ponen en peligro el buen funcionamiento del sistema, reducir los tiempos de desarrollo, aumentar la productividad y la competitividad.

Tanto ¨***UML***¨ como las **Redes de Petri,** constituyen actualmente las de mayor aplicación en la descripción de procesos, ya que facilitan una modelación sencilla e intuitiva, siendo generalmente sus modelos más compactos y poderosos.

Conclusiones del Capítulo

En este capítulo se ha realizado una caracterización de los Sistemas Automatizados en el ámbito Industrial y se ha arribado a las siguientes conclusiones:

1. El empleo de los Sistemas Automatizados está condicionado por las necesidades industriales de aumentar la calidad y cantidad de productos, así como garantizar la seguridad y buena operatividad tanto del personal de la planta como de los elementos tecnológicos que la conforman.
2. Para el correcto desempeño de las funciones y objetivos de un Sistema Automatizado moderno es necesario el uso de medios técnicos digitales, y además determinar cuál de los Sistemas Digitales (***SCD***, ¨***SCADA***¨, ¨***PLC***¨ y ¨***PC***¨**,** entre otros) es el más adecuado para la planta.
3. Debido a la gran cantidad de recursos materiales e intelectuales que requiere la implementación de un Sistema Automatizado, su modelación constituye un paso de vital importancia para lograrlo.

Capítulo 2- Particularidades de las Redes de Petri y UML en el diseño de Sistemas Automatizados.

En este capítulo se presentan las principales características, propiedades y particularidades para el diseño de las Redes de Petri y ¨*UML*¨. También se exponen algunos ejemplos reales que ilustran las ventajas y beneficios que brindan estas herramientas en la automatización industrial.

2.1- Generalización sobre las Redes de Petri.

Las Redes de Petri fueron creadas por el alemán Karl Adam Petri en su tesis de doctorado a principios de los años 60. Hasta la actualidad, este término ha servido para designar a un amplio número de modelos de sistemas, técnicas de análisis, procedimientos, representaciones gráficas y patrones descriptivos, los cuales están basados en premisas específicas sobre el mundo del procesamiento de la información. (Israel Benítez et al., 2002, Reisig and Rozenberg, 2005)

Constituyen una potente herramienta de modelado, descripción y análisis, tanto gráfico como matemático aplicable a una gran diversidad de sistemas (Murata, 1989).

Como herramienta gráfica, las RP sirven de soporte para la comunicación visual, al igual que los diagramas de flujo de datos y diagramas de bloques funcionales abordados en el capítulo anterior, además, los ¨*tokens*¨ o marcas que se utilizan en estas redes sirven para simular las actividades dinámicas y concurrentes.

Como herramienta matemática, permiten modelar y analizar ecuaciones de estados, ecuaciones algebraicas y otros modelos matemáticos (Murata, 1989). También facilita el análisis y la verificación de un gran número de propiedades presentes en los sistemas. El análisis de estas propiedades permite la verificación de requisitos funcionales de los modelos, posibilitando la detección

de posibles errores y por tanto la corrección de estos antes de pasar a la etapa de implementación.

Como se plantea en (Murata, 1989) la utilización de esta herramienta de modelado se hace efectiva en áreas como:

- Fabricación Flexible y Sistemas de Control Industrial.
- Sistemas de eventos discretos.
- Modelado y análisis de sistemas de ¨*hardware*¨, ¨*software*¨ y bases de datos distribuidos.
- Sistemas de tolerancia a fallos.
- Redes Neuronales Artificiales.
- Sistemas de memoria multiprocesador y flujo computarizado de datos.
- Programas concurrentes y paralelos.

La definición clásica de las Redes de Petri ordinarias, está formada de cuatro componentes básicos que forman su estructura: Un conjunto de lugares o estados *P*, un conjunto de transiciones *T*, la función de entrada *I*, y la función de salida *O*. Las funciones de entrada y salida relacionan las transiciones y los lugares. La definición según (David and Alla, 1992, Rozenberg and Engelfriet, 2005) queda como: *PN*= (P, T, I, O) donde:

P= *{p1, p2,..., pn}* es un conjunto finito de lugares.

T= *{t1, t2,..., tm}* es un conjunto finito de transiciones.

I= $P \Rightarrow T$: es la función de entrada, un mapeo desde los lugares de entrada hacia el conjunto de transiciones.

O= $T \Rightarrow P$: es la función de salida, un mapeo desde las transiciones hacia el conjunto de lugares de salida.

No obstante, otra definición muy utilizada por varios autores en el ámbito científico, como (Murata, 1989), define matemáticamente una Red de Petri como una quíntupla RP= (**P**, **T**, **F**, **W**, $\mathbf{M_o}$) donde:

P= $\{p_1, p_2, \ldots, p_m\}$: conjunto finito de lugares o estados (representados por circunferencias).

T= $\{t_1, t_2, t_3, \ldots, t_n\}$: conjunto finito de transiciones (representados generalmente por barras o rectángulos).

F: conjunto de arcos que unen estados con transiciones y viceversa (relación de flujo).

W: $F \rightarrow \{1, 2, 3, \ldots\}$: función de peso.

$\mathbf{M_o}$: $P \rightarrow \{1, 2, 3, \ldots\}$: marcación inicial.

$(P \cap T) = \emptyset$

$(P \cup T) \neq \emptyset$

En esta definición cuando no se especifica una marcación inicial la estructura de la Red de Petri es una cuádrupla RP= (**P**, **T**, **F**, **W**).

Para simular el comportamiento dinámico del sistema, las marcas se desplazan por la red de acuerdo a las reglas de disparo de las transiciones (Fig.2.1): (Salmon, 2009)

1. Una transición *t* está habilitada si cada estado de entrada *p* a la transición *t* está marcado con al menos w (p, t) *tokens*, donde w (p, t) es el peso del arco desde *p* a *t.*
2. Una transición habilitada puede o no dispararse (si los eventos de que depende tienen o no lugar en ese momento).
3. Una transición habilitada *t* elimina *w (p; t)* tokens de cada lugar de entrada

p de *t*, y adiciona *w (t; p)* tokens a cada lugar de salida *p* de *t,* donde *w (t; p)* es el peso del arco de *t* a *p*.

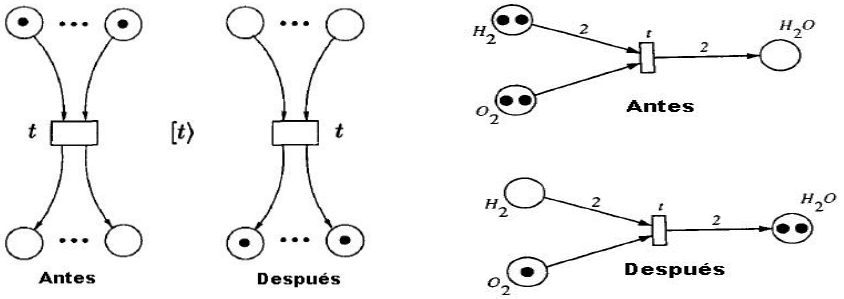


Figura 2.1-Disparo de Transiciones.

En el análisis y diseño de modelos basados en RP se tienen en cuenta un conjunto de propiedades (Murata, 1989) , pudiendo clasificarse estas como:

- Propiedades dependientes del marcaje, o **propiedades funcionales**: Son las propiedades que dependen del marcaje inicial y reflejan el comportamiento dinámico del sistema.
- **Propiedades estructurales**: Son independientes del marcaje inicial de la red.

Según (Israel Benítez et al., 2002), entre las **propiedades funcionales** más importantes se encuentran:

1. Alcanzabilidad: Verifica si todos los estados del modelo son alcanzables. A partir de un estado inicial, muchas veces se requiere que la red avance automáticamente hasta un estado dado, si no tiene esta capacidad, no podrá ejecutar el comportamiento deseado.
2. Vivacidad: Evita bloqueos y lazos infinitos para tener un programa eficiente.
3. Limitación: Garantiza que los lugares no excedan su capacidad permitida.
4. Reversibilidad: La subred siempre puede retornar al estado inicial.
5. Persistencia: Verifica paralelismos y sincronismos, para evitar conflictos.

Entre las **propiedades estructurales** que más se analizan se encuentran la **Vivacidad Estructural** y **Limitación Estructural** (Israel Benítez et al., 2002)**.** La primera asegura que todas las transiciones puedan ser disparadas a partir de un marcaje inicial, la segunda asegura que el número total de ¨*tokens*¨ en la red sea finito y que no haya acumulación en ningún nodo. Otras propiedades son la capacidad de **Repetitividad**, la **Controlabilidad** y las **Invariantes S** y **T**, siendo estas últimas unas de las más importantes. En el caso de las Invariantes **S** (Rozenberg and Engelfriet, 2005) descubren la estructura mínima de los circuitos que pueden compartir marcas independientemente de la evolución dinámica que experimente la red. Por su parte, las invariantes **T** definen el número de veces que cada transición debe dispararse para ir desde una marca $\mathbf{M_o}$ hasta esa misma marca inicial completando un ciclo entero.

2.2- Extensiones de las Redes de Petri.

Las Redes de Petri que describen el comportamiento del mundo real tienden a ser complejas y grandes. Las RP clásicas no permiten la modelación de datos, no es posible establecer condiciones temporales, así como tampoco especificar una política de selección entre transiciones diferentes que estén habilitadas para dispararse, entre otras características de los sistemas actuales. Para solucionar este problema, han surgido varias extensiones, las cuales se pueden agrupar en tres categorías: extensiones con tiempo, con colores y con jerarquía: (Salmon, 2009)

Extensiones con tiempo: Las ¨***TPN****s*¨ (¨*Timed Petri Nets*¨), según (Du et al., 2007), pueden definirse como: ***TPN*** **=** (**P**, **T**, **F**, $\mathbf{M_o}$, **SI**), donde (**P**, **T**, **F**, $\mathbf{M_o}$) es una **RP**, y **SI** una función de intervalo de disparo estático.

Estas extensiones permiten una fácil representación de las características de los sistemas en tiempo real (sincronismo, paralelismo, etc.). El tiempo puede estar asociado con los ¨*tokens*¨, lugares y/o transiciones. Las RP con tiempo pueden dividirse a su vez en dos clases: RP de tiempo determinístico (regulares) y de

tiempo estocástico (estocásticas). Las más utilizadas son las Redes de Petri Estocásticas Generalizadas (¨***GSPN***-*Generalize Stochastic Petri Nets*¨). El modelo ¨*GSPN*¨ es una extensión de las Redes de Petri Ordinarias, donde están definidas transiciones temporizadas con retardos exponenciales negativos y además transiciones inmediatas.

Extensiones con color: Las ¨***CPN****s*¨ ("*Colored Petri Nets"*) son una extensión de las RP que tienen incorporado un lenguaje de modelación. Se consideran ideales para sistemas en los cuales la comunicación, sincronización y el uso compartido de recursos son importantes, además provee modelos compactos para sistemas grandes con un elevado nivel de abstracción y una capacidad perfeccionada de representación gráfica. Estas redes sacan provecho de la simetría que existe en los modelos de RP agrupando los estados idénticos y transiciones, y definiendo funciones de matrices apropiadas para representar los nodos, arcos, colores, marcas, e invariantes. Pueden considerar además referencias de tiempo. (Peng and Zhou, 2004)

Extensiones con Jerarquía: Estas redes proporcionan una jerarquía de subredes. Una subred es un conjunto de lugares, transiciones, arcos e incluso de mismas subredes. De esta forma la construcción de grandes sistemas, se basa en un mecanismo de estructuración de dos o más procesos, representados por subredes, tal que, en un nivel se da una descripción simple de los procesos, y en otro nivel se da una descripción más detallada de su comportamiento (Salmon, 2009). Las extensiones con jerarquía aparecieron como extensiones a las Redes de Petri coloreadas, aunque se han desarrollado tipos de Redes de Petri jerárquicas que no son coloreadas, como por ejemplo las redes extendidas ¨***GHENeSys***¨ (¨*General Hierarchical Enhanced Net System*¨), las cuales se definen como una séxtupla G= (**L**, **A**, **F**, **K**, **M**, **π**) donde: (Israel Benítez et al., 2002).

L: Son llamados lugares y está compuesto por la unión de los conjuntos B y P (¨*boxes*¨ y ¨p*seudoboxes*¨, respectivamente).

A: Son las actividades, a cuyos elementos se enlazan por arcos habilitadores e inhibidores el conjunto P.

F: Es la relación de flujo y sus elementos son llamados arcos.

K: Es la función de capacidad que indica la capacidad máxima permitida en cada lugar.

M: Es el marcaje inicial de la red respetando las capacidades de cada lugar.

Π: Es una función que identifica los elementos macros de la red, siendo igual a 0 para los elementos simples e igual a 1 para los elementos macros.

Actualmente, las redes ***GHENeSys*** son ampliamente usadas en la automatización, resultando ser una propuesta adecuada al concepto de modularidad de sistemas complejos, presentada en un contexto más cercano a los sistemas de control distribuidos (Salmon, 2009).

Para realizar un diseño eficiente y correcto es necesario, además de conocer las definiciones y propiedades, seguir una metodología adecuada que asegure el cumplimiento de los objetivos.

2.3- Metodología para el diseño

Las ventajas de los métodos formales en el proceso de diseño como por ejemplo, la facilidad para el análisis, la reutilización, la repetitividad, la facilidad para proceder al análisis de sistemas de gran tamaño, la precisión del proceso de modelado y la facilidad de documentación, justifica la obtención de nuevos métodos que tornen más rápido y seguro este proceso, pero sin descuidar el contenido formal. En este campo las Redes de Petri y sus extensiones han sido la representación o formalismo que más ha avanzado por su dualidad matemático-gráfica, su aplicabilidad y expresividad (Israel Benítez et al., 2002).

Infinidad de sistemas de automatización han sido modelados y/o analizados con esta herramienta. De forma general, independientemente del tipo de red que se use y de las características físicas y funcionales del sistema, se han propuesto metodologías o pasos a seguir por parte de diferentes autores para lograr un correcto diseño y una fácil interpretación del modelo.

Entre los principales pasos a seguir se encuentran: (Israel Benítez et al., 2002, Salmon, 2009, Sarasola, 2008)

1. Estudiar el sistema a controlar y los requerimientos funcionales del usuario.
2. Identificar los posibles subsistemas que forman parte del sistema principal, sin coincidir necesariamente con su futura implementación, con la idea de reducir la complejidad a la hora de realizar el análisis.
3. Analizar y modelar cada uno de los subsistemas con los modelos básicos presentados, de forma que en su estructura presenten las características de limitación, vivacidad y reversibilidad. Para realizar el análisis se comenzará con una descripción detallada del funcionamiento del subsistema, seguido con una identificación de los elementos necesarios para llevar a cabo la tarea de automatización (sensores, actuadores, etc.). Una vez completada la identificación de los elementos, se procederá determinar las condiciones iniciales en las que se ha de encontrar el subsistema para que empiece a trabajar, así como las condiciones externas proporcionadas por otros subsistemas para iniciar la secuencia. Las condiciones externas deben tener en cuenta el estado de otros subsistemas, de forma que no se produzcan movimientos prohibidos que pongan en peligro la integridad de diferentes elementos.
4. Modelar el sistema completo, introduciendo el formato estándar de los subsistemas e interrelacionándolos, de forma que se obtenga el funcionamiento deseado para el sistema principal.

5. Aumentar la red de Petri obtenida introduciendo estructuras de recuperación de errores.
6. Refinar el modelo para detallar la estructura de cada subred. Aquí, inicialmente, se van integrando los modelos de los subsistemas según la secuencia de operación, quedando conformado el modelo de planta sin control. Esto garantiza el desarrollo de nuevas sub-redes integradas que representan las secuencias deseadas de funcionamiento de la planta, los cuales constituyen elementos para la síntesis del controlador.
7. Determinar las propiedades funcionales y estructurales.
8. Aplicar métodos de reducción simples para revisar la estructura de las sub-redes y alcanzar la mayor independencia entre selección y concurrencia de las ramas del modelo, creando modelos bien formados.
9. Simular el funcionamiento del sistema.
10. Traducir el modelo para un programa IEC-1131 compatible.

En dependencia de la Red de Petri que se use y las exigencias de la planta se pueden omitir y añadir pasos, así como cambiar el orden de estos.

2.4- Ejemplos de sistemas modelados usando Redes de Petri

Un ejemplo sencillo usando las RP lo constituye realizar la automatización de un montacargas que se desplaza entre dos pisos, expuesto en (Berea, 2006) (Fig.2.2).

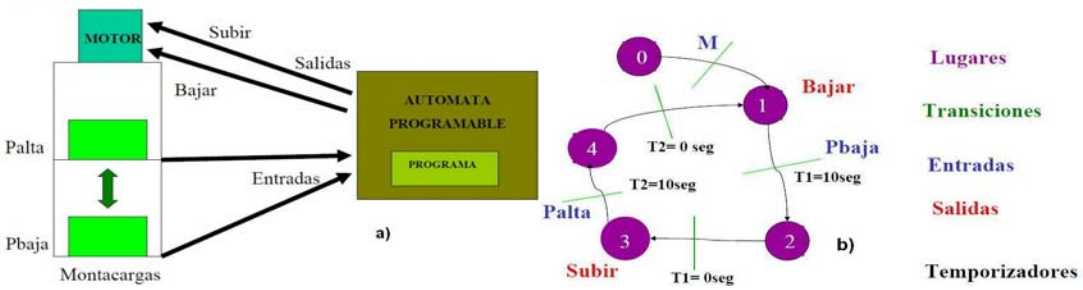


Figura 2.2- a)-descripción informal del proceso, b)-modelo en RP del montacargas.

A partir del estudio, se desea que el sistema arranque al pulsar el contacto M y se posicione en la planta baja, manteniéndose 10 segundos en cada planta. Las señales de entrada identificadas son: contacto M, sensor de presencia en planta baja (Pbaja), sensor de presencia en planta alta (Palta). Las señales de salida son subir y bajar. Siguiendo la metodología propuesta anteriormente, se salta del paso 1 al 4, ya que no es necesario modelar subredes. Luego de obtenido el modelo completo se verifican las propiedades para encontrar errores. Posteriormente se simula y se traduce a lenguaje de ¨*PLC*¨.

El anterior ejemplo tiene un carácter educativo y sirve para ilustrar de forma general el proceso de diseño en RP.

Existen muchos otros ejemplos reales que se pueden citar sobre automatización y análisis de sistemas a través del modelado en Redes de Petri

En nuestro país, aplicaciones de estas redes están siendo desarrolladas por la **UO** (Universidad de Oriente) en colaboración con la **USP** (Universidad de Sao Paulo) y la Universidad de Manaus. Un ejemplo lo constituye el modelado en Redes de Petri ¨***GHENeSys***¨ del sistema de automatización de la sección de molinos de la fábrica de cemento de Santiago de Cuba. Implementado por la Empresa de Automatización Integral **CEDAI** en unión con la UO, tuvo como objetivo la reducción de los tiempos de paradas por averías y el consumo energético, el incremento de la productividad y la mejora de las condiciones de trabajo en la planta. Entre los modelos que se obtuvieron se encuentran: (Fig.2.3) (Israel Benítez et al., 2010b)

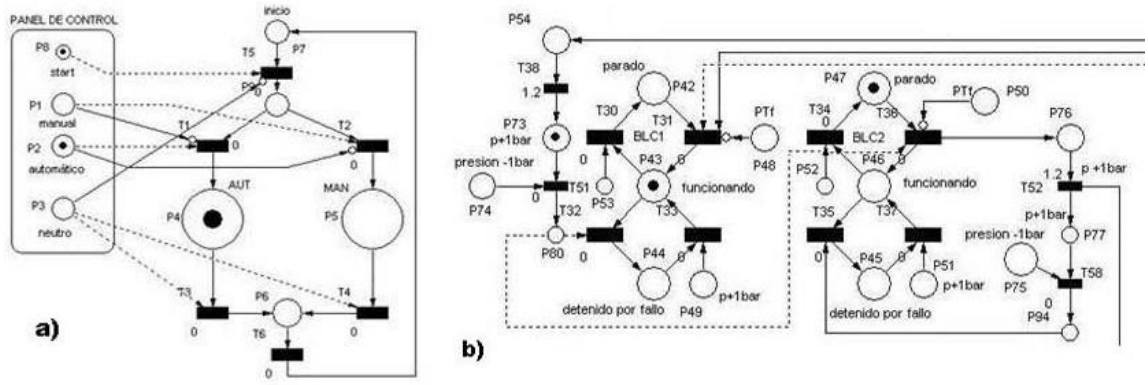


Figura 2.3- a) Selección de los modos de Funcionamiento, b) Bombas de lubricación

En el modelo a) se presentan los modos de funcionamiento (manual, automático, neutro), mientras que en el b) se muestra parcialmente el funcionamiento automático de las bombas de lubricación de collar y de los demás componentes del molino (bombas de aceite, ventiladores, electroválvulas etc.). Luego se analizaron las propiedades funcionales y estructurales para evitar malformaciones en el modelo.

Otro ejemplo es el modelado y programación del sistema supervisor de la planta de Hormigón de Santiago de Cuba ¨Santiago 1¨, en el cual se utilizó el programa ¨*Visual Object Net*¨ para la creación de los modelos y el ¨*CoDeSys*¨ para la traducción a lenguajes de ¨*PLC´s*¨ según la norma IEC-1131.

Con este proyecto se logró elevar el grado de automatización de la planta el cual era muy bajo y además, incrementar la eficiencia y productividad al mejorar la coordinación de actividades integradas. Los modelos obtenidos incluyen control local y remoto, regímenes de operación manual y automático, arranque y parada de la instalación y parada de emergencia con señalización a fallos.

De forma similar, en una gran variedad de artículos científico-técnicos se hace referencia al uso de las Redes de Petri en aplicaciones relacionadas con la Automatización.

Un ejemplo se trata en (Ghomri and Alla, 2007), utilizando Redes de Petri para la creación de modelos sencillos que sirven para realizar el llenado de tanques industriales. También se han usado las RP para modelar sistemas de reconocimiento de patrones y visualización industrial, capaces de especificar de forma sencilla tareas de alto nivel que tienen que realizar robots móviles autónomos, como el desarrollado en (Joaquin L. Fernández et al., 2007).

En (Sarasola, 2008) se describe el proceso de modelado y simulación del complejo sistema de control de una Célula de Fabricación Flexible, la cual está integrada por cuatro estaciones de trabajo y un sistema de transporte encargado de trasladar el material entre ellas, obteniendo modelos como el de la (Fig.2.4), que aunque pueda parecer muy extenso, se puede traducir fácilmente a lenguajes de ¨*PLC's*¨.

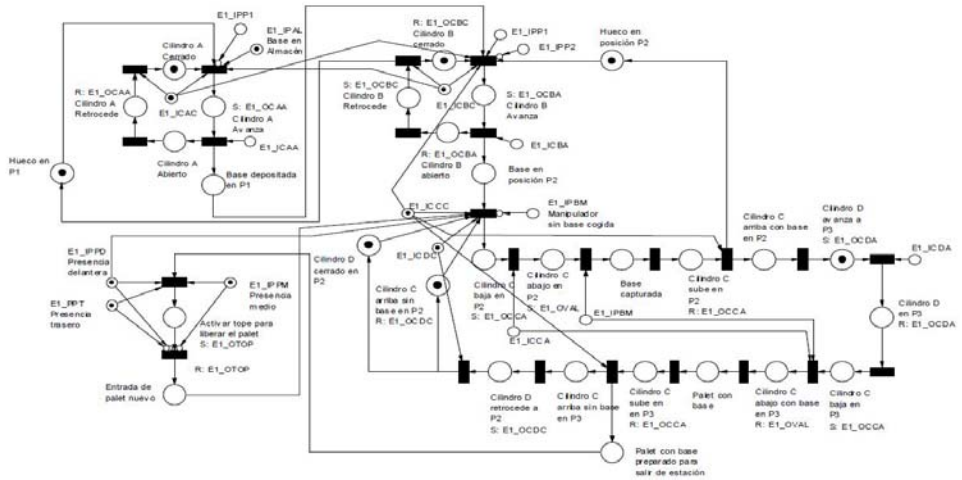


Figura 2.4- Modelo del Sistema de Control de una Célula de Fabricación Flexible.

2.5-El Lenguaje Unificado de Modelado

El Lenguaje Unificado de Modelado es la creación de Grady Booch, James Rumbaugh e Ivar Jacobson. Estos caballeros, apodados "Los tres amigos", trabajaban en empresas distintas durante la década de los años ochenta y principios de los noventa del siglo XX, y cada uno diseñó su propia metodología para el análisis y diseño orientado a objetos. Sus metodologías predominaron

sobre las de sus competidores, y a mediados de los años noventa se unieron a la compañía ¨*Rational*¨ de Booch, por lo que empezaron a intercambiar ideas entre sí y decidieron desarrollar su trabajo en conjunto.

¨*UML*¨ es el lenguaje de modelado orientado a objetos más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar cualquier tipo de sistema. Ofrece un estándar de descripción de modelos, que incluyen aspectos conceptuales tales como procesos de negocios y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de ¨*software*¨ reutilizables (Booch et al., 1998). Es ideal incluso para modelar sistemas empotrados de tiempo real muy exigentes y cubrir la documentación de la arquitectura del sistema y todos sus detalles. ¨*UML*¨ también proporciona un lenguaje para expresar requisitos y pruebas, proporcionando un lenguaje para modelar las actividades de planificación de proyectos y gestión de versiones.

Otra característica de ¨*UML*¨ (Orallo, 2002) es que sus modelos son independientes del lenguaje de implementación, de tal forma que los diseños realizados se puedan implementar en cualquier lenguaje que soporte las posibilidades de *UML* (principalmente lenguajes orientados a objetos).

¨*UML*¨ aporta las siguientes ventajas a la hora del diseño:

1. Mayor rigor en la especificación.
2. Permite realizar una verificación y validación del modelo realizado.
3. Se pueden automatizar determinados procesos y permite generar código a partir de los modelos y a la inversa (a partir del código fuente generar los modelos).

Aunque está pensado para modelar sistemas complejos con gran cantidad de ¨*software*¨, el lenguaje es lo suficientemente expresivo como para modelar sistemas que no son informáticos, como flujos de trabajo (*workflow)* en una

empresa, diseño de la estructura de una organización, diseño de ¨*hardware*¨, y por supuesto para Sistemas Automatizados.

Como se mencionó anteriormente, los principales objetivos de ¨*UML*¨ básicamente se pueden centrar en (Booch et al., 1998, Orallo, 2002): visualizar, especificar, construir y documentar un sistema.

- Visualizar: ¨*UML*¨ permite expresar de una forma gráfica y generalizada un sistema, de forma que otro lo pueda entender. Detrás de cada símbolo en la notación hay una semántica bien definida. De esta manera, un desarrollador puede escribir un modelo en ¨*UML*¨, y otro desarrollador, o incluso otra herramienta, puede interpretar ese modelo sin ambigüedad
- Especificar: ¨*UML*¨ permite especificar cuáles son las características de un sistema antes de su construcción. Especificar significa construir modelos precisos, no ambiguos y completos. En particular, cubre la especificación de todas las decisiones de análisis, diseño e implementación que deben realizarse al desarrollar y desplegar un sistema.
- Construir: A partir de los modelos especificados se pueden construir los sistemas diseñados. Sus modelos pueden conectarse de forma directa a una gran variedad de lenguajes de programación, lo que significa que es posible establecer correspondencias desde un modelo ¨*UML*¨ a un lenguaje de programación como ¨*Java*¨, C++, ¨*Visual Basic*¨, o incluso a tablas en una base de datos relacional o al almacenamiento persistente en una base de datos orientada a objetos.
- Documentar: Los propios elementos gráficos sirven como documentación del sistema desarrollado, que pueden servir para su futura revisión, además, proporciona un lenguaje que permite expresar requisitos y pruebas.

Para comprender ¨*UML*¨, se necesita adquirir un modelo conceptual del lenguaje, y esto requiere conocer tres elementos principales: los bloques básicos

de construcción, las reglas que dictan cómo se pueden combinar estos bloques básicos y algunos mecanismos comunes que se aplican en el modelado con esta herramienta.

2.5.1-Bloques de construcción y diagramas.

El vocabulario de ¨*UML*¨ está compuesto por tres clases de bloques de construcción: **Elementos**, **Relaciones** y **Diagramas**. Los **Elementos** son abstracciones de cosas reales o ficticias (objetos, acciones, etc.). Existen cuatro tipos de elementos principales, estos son: elementos estructurales, de comportamiento, de agrupación y de anotación. (Booch et al., 1998, Mora, 2002)

Los *elementos estructurales* son los nombres de los modelos. En su mayoría son las partes estáticas de un modelo, y representan cosas que son conceptuales o materiales. En total, hay siete tipos de elementos estructurales: clases, interfaces, colaboraciones, casos de uso, clases activas, componentes y nodos. Los elementos de comportamiento son las partes dinámicas de los modelos ¨*UML*¨. Estos son los verbos de un modelo, y representan comportamiento en el tiempo y el espacio. Los elementos de agrupación son las partes organizativas de los modelos, las cajas en las que puede descomponerse. En total hay un elemento de agrupación principal: los paquetes. Por último, los elementos de anotación son las partes explicativas de los modelos. Son comentarios que se pueden aplicar para describir, clarificar y hacer observaciones sobre cualquier elemento del modelo. Hay un tipo principal de elemento de anotación llamado **nota**.

Hay cuatro tipos de **Relaciones** en ¨*UML*¨ (Fig.2.5): Las de ¨dependencia¨ constituyen una relación semántica entre dos elementos, en la cual un cambio a un elemento puede afectar a la semántica del otro. Las relaciones de ¨asociación¨ conforman una relación estructural que describe un conjunto de enlaces que son conexiones entre objetos. Asimismo están las de ¨generalización¨, que son relaciones de especialización-generalización, donde

los objetos del elemento especializado (hijo) pueden sustituir a los objetos del elemento general (padre). Por último están las de ¨realización¨, que establece una semántica entre clasificadores, en donde un clasificador especifica un contrato que otro clasificador garantiza que cumplirá.

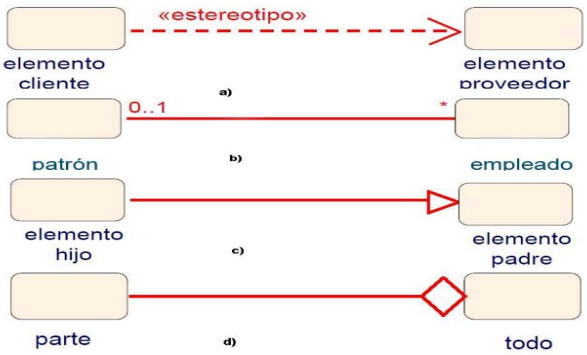


Figura 2.5- Relaciones en UML. a)-dependencia, b)-asociación, c)-generalización, d)-realización.

También existen variaciones de estos cuatro tipos, tales como el refinamiento, la traza, la inclusión y la extensión (para las dependencias).

Un **Diagrama** constituye la representación gráfica de un conjunto de elementos con sus relaciones. En concreto, ofrece una vista del sistema a modelar. Para poder representar correctamente un sistema, ¨*UML*¨ ofrece una amplia variedad de diagramas para visualizar el sistema desde varias perspectivas, incluyendo los siguientes: diagramas de casos de uso, de clases, de objetos, de secuencia, colaboración, estados, actividades, componentes, y despliegue (Booch et al., 1998, Mora, 2002, Orallo, 2002). Los diagramas que más se utilizan y los más interesantes (Orallo, 2002) son los de casos de uso, clases, secuencia y actividades (Fig.2.6).

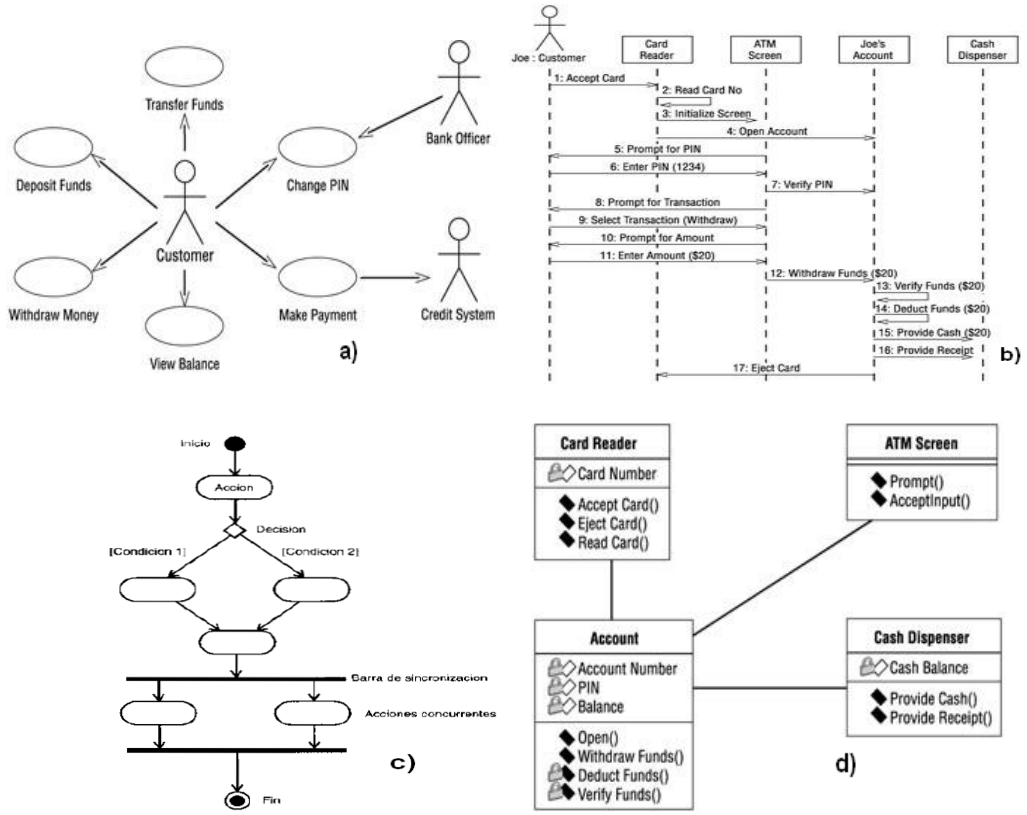


Figura 2.6- Diagramas ¨*UML*¨, a)-casos de uso, b)-secuencia, c)-actividades, d)-clases.

- Diagramas de casos de uso: representan gráficamente los casos de uso que tiene un sistema. Se define un caso de uso como cada interacción supuesta con el sistema a desarrollar, donde se representan los requisitos funcionales. En este tipo de diagrama se encuentran los actores que constituyen una idealización de una persona externa, proceso, subsistema o clase que interactúa con el sistema.

- Diagramas de clases: muestran un conjunto de clases, interfaces y sus relaciones. Éste es el diagrama más común a la hora de describir el diseño de los sistemas orientados a objetos.
- Diagramas de secuencia: muestran la interacción entre los objetos que componen un sistema de forma temporal.
- Diagramas de actividades: muestran el flujo lógico a través de un caso de uso.
- Diagramas de colaboración: muestran los objetos que son necesitados

para implementar la funcionalidad de un caso de uso, e incluyen comunicación entre objetos.

- Diagramas de estado: son usados para modelar el funcionamiento dinámico y frecuentemente son usados también en sistemas de tiempo real.
- Diagramas de componentes: muestran los componentes que serán creados por el sistema y las relaciones entre ellos.
- Diagramas de despliegue: se usan para mostrar la estructura de la red y donde será desplegado el sistema dentro de la misma.

2.5.2-Proceso de Desarrollo.

Aunque ¨*UML*¨ es bastante independiente del proceso de desarrollo que se siga, los mismos creadores han propuesto su propia metodología de desarrollo, denominada el **Proceso Unificado de Desarrollo**. (Orallo, 2002, Grady Booch et al., 2003)

El Proceso Unificado está basado en componentes, lo cual quiere decir que el sistema en construcción como tal está formado por componentes interconectados a través de interfaces bien definidas. Además, el Proceso Unificado utiliza el ¨*UML*¨ para expresar gráficamente todos los esquemas representativos de un sistema. Pero realmente, los aspectos que definen este Proceso Unificado son tres: Es dirigido por casos de uso, centrado en la arquitectura y además iterativo e incremental.

- **Dirigido por casos de uso**: Basándose en los casos de uso, los desarrolladores crean una serie de modelos de diseño e implementación que los llevan a cabo. Además, estos modelos se validan para que sean conformes a los casos de uso. Finalmente, los casos de uso también sirven para realizar las pruebas sobre los componentes desarrollados.
- **Centrado en la arquitectura**: Cada uno de los aspectos de la arquitectura de un sistema está representado por un gráfico con su

notación correspondiente. El concepto de arquitectura de ¨*software*¨ incluye los aspectos estáticos y dinámicos más significativos que componen un sistema.

- **Iterativo e incremental**: Todo sistema complejo supone un gran esfuerzo que puede durar desde varios meses hasta años. Por lo tanto, lo más práctico es dividir un proyecto en varias fases. Actualmente se suele hablar de ciclos de vida en los que se realizan varios recorridos por todas las fases. Cada recorrido por las fases se denomina iteración en el proyecto en la que se realizan varios tipos de trabajo (denominados flujos). Además, cada iteración parte de la anterior incrementado o revisando la funcionalidad implementada.

En los últimos años se ha venido utilizando ¨*UML*¨ para realizar el modelado formal y la verificación de sistemas que no son de ¨*software*¨ específicamente, como por ejemplo de Sistemas Automatizados.

2.5.3- Ejemplos de sistemas modelados usando UML.

Muchas empresas de desarrollo de ¨*software*¨ a nivel mundial emplean ¨*UML*¨ para la creación de sus productos. De forma similar, existen alrededor del mundo cerca de una docena de grupos de investigación especializados que trabajan en la dirección de la ingeniería de sistemas orientados a objetos en la automatización industrial, usando ¨*UML*¨, bloques funcionales y ¨*Java*¨ como ideas básicas.

El primer ejemplo que se muestra, ilustra de una manera sencilla la creación de un diagrama de casos de uso a partir de la descripción informal del problema.

Se trata de un cliente que llega a una cabina de venta automatizada para comprar algunos artículos, e interactúa con un sistema ¨*POS*¨ (¨*Point of Sale System*¨) (Larman, 2004). Este es un sistema de aplicación computarizado, utilizado para registrar procesos de venta y manejar pagos, típicamente usado

en negocios de venta al por menor. La cabina usa el sistema ¨*POS*¨ para registrar cada artículo adquirido y presentar información detallada de estos al usuario, así como el total de pago. El cliente introduce la información de pago, la cual el sistema valida y registra. El sistema actualiza el inventario y le entrega un comprobante al cliente. Para hacer el diagrama es necesario tener en cuenta todos los actores y acciones que podrían participar en el proceso (Fig.2.7).

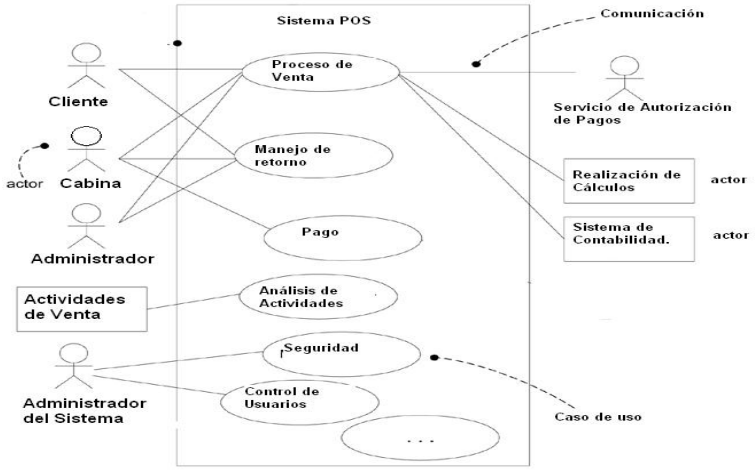


Figura 2.7- Diagrama de casos de uso del sistema ¨*POS*¨.

El carácter universal de ¨*UML*¨ es tan amplio que permite ser usado para modelar aplicaciones tan específicas como lo es un **Sistema de Control Distribuido** usado para el control en tiempo real de procesos tecnológicos, como el desarrollado en (Kotzian and Srovnal, 2004). En este sistema se hace necesario que los requerimientos de fiabilidad y seguridad general para todo el sistema sean elevados. En este caso se hace un énfasis especial en el modelo de la comunicación entre los dispositivos mediante buses de campo industriales. Luego de obtenido, sirve como base para la construcción del modelo general del sistema y para su posterior optimización y validación. El modelo general que se obtiene (Anexo 1) incluye el del bus (Anexo 2), modelos de interfaces, funcionamiento de varias aplicaciones y generador de eventos, además cuenta con una base de tiempo. Los principales diagramas usados son los de casos de uso, secuencia, actividades y estado.

¨*UML*¨ se emplea también de forma satisfactoria para realizar la **verificación formal** de Sistemas de Automatización Industrial. Según se plantea en (Vyatkin et al., 2005) la verificación formal es una importante vía para la validación del ¨*software*¨ intensivo de sistemas de automatización flexibles y sistemas de control distribuidos, sin embargo requiere un modelo preciso del sistema a controlar. Las técnicas de modelado y verificación orientadas a objetos están basadas en subconjuntos extendidos de ¨*UML*¨ y adoptan varios tipos de sus diagramas. Los diagramas de clases son usados para representar el modelo estructural y para describir de forma jerárquica la arquitectura del sistema mediante vínculos de composición entre los objetos. Los diagramas de colaboración estereotipados como ¨Diagramas de Flujo de Datos Mecatrónicos¨, especifican la interacción entre los objetos y los de estado el funcionamiento de estos. La verificación formal con ¨*UML*¨ permite una descripción modular, reutilizable e independiente de la plataforma de implementación del modelo del sistema de control.

El uso de ¨*UML*¨ en el control y la automatización es también examinado en (Thramboulidis and Tranoris, 2003), los cuales integran ¨*UML*¨ con el concepto de bloques funcionales (¨*Function Block,* ***FB*** ¨). A través del estándar IEC-61499 se definen los conceptos básicos y la metodología a seguir por los diseñadores de sistemas de automatización, para producir un software modular e inter-operable que trabaje adecuadamente en aplicaciones industriales que presenten Sistemas de Control Distribuido. En este caso, luego de crear el modelo del sistema en ¨*UML*¨, este se puede traducir a diagramas de ¨*FB*¨ a través de reglas de transformación previamente definidas. Posteriormente el diseño en ¨*FB*¨, que puede ser más comprensible para ingenieros en control, es transformado en modelos ejecutables, o traducido a lenguajes de programación como C++, o lenguaje de autómatas.

En (O.R.Natale et al., 2002) se hace referencia al uso de ¨*UML*¨ en el modelado e implementación de bases de datos relacionales para la rastreabilidad, en

industrias de procesamiento por lotes; de igual manera en (T. Tsai and R. Sato, 2004) se habla de ¨*UML*¨ para realizar la especificación y validación de políticas de planificación en sistemas de producción ligeros.

Conclusiones del Capítulo

1. Las herramientas de modelado ¨*UML*¨ y RP constituyen armas poderosas para el diseño y análisis de sistemas de automatización, además, sus modelos son fácilmente comprensibles y se pueden traducir directamente a lenguajes de programación.

Capítulo 3- Integración de "*UML*" y Redes de Petri para la modelación de Sistemas Automatizados.

En este capítulo se exponen las diferentes formas en que se podrían integrar " "*UML*" y Redes de Petri para lograr la modelación de sistemas automatizados, se presentan algunos ejemplos y se exponen las ventajas y desventajas que trae consigo la fusión.

3.1- Integración *UML-RP*

Para lograr la implementación de un sistema de fabricación automatizado y altamente integrado, es necesario no solo controlar el proceso de fabricación en sí, sino también administrar y controlar el flujo de información a través de todo el sistema computarizado que existe en las industrias modernas. En aras de crear ambientes integrados para aplicaciones de fabricación, diseñar productos y procesos mediante herramientas avanzadas ¨*CAD*¨ (¨*Computer Aided Design*¨), realizar planes estratégicos de producción y controlar eficientemente el proceso; se ha asistido recientemente al empleo en conjunto de la Ingeniería de ¨*software*¨ y las herramientas de modelado formal.

Como se plantea en (Chang-Pin Lin et al., 2004), la reciente evolución que se está experimentando en el diseño y el análisis orientado a objetos, con herramientas formales de modelado dentro de la Ingeniería de ¨*software*¨, ha promovido el desarrollo del Lenguaje Unificado de Modelado. ¨*UML*¨ ha sido aprobado por el ¨***OMG***¨ (¨*Object Management Group*¨), y se ha convertido en una herramienta industrial estándar para el modelado en todos los aspectos del desarrollo de software. No obstante, los diagramas ¨*UML*¨ solos no son ideales del todo en el marco del análisis, síntesis de control y validación. Especialmente en el contexto de la automatización, ya que presentan problemas para describir exactamente el funcionamiento dinámico de sistemas, paralelismos,

sincronismos y concurrencias en procesos. Por lo que ha resultado necesario y conveniente tener un conjunto de reglas de traducción que permitan obtener un modelo en un dominio operacional diferente.

Por su parte las RP, como se planteó en el capitulo anterior, son ampliamente utilizadas para modelar y analizar sistemas industriales debido a su semántica formal, naturaleza grafica, expresividad, y la disponibilidad de técnicas de análisis basadas en sus propiedades (Francesco Basile et al., 2008).

La intención es utilizar e incrementar el confort de la expresividad de las Redes de Petri y de esa manera hacer factible el modelado de sistemas complejos, pero al mismo tiempo obtener los beneficios de los sistemas orientados a objetos, incluyendo interfaces limpias y bien definidas, componentes de software reutilizables y librerías de componentes extensibles.

Varios autores han publicado sobre la utilización de los diagramas ¨*UML*¨ en combinación con lenguajes formales como las Redes de Petri para modelar el funcionamiento de sistemas de automatización.

3.2- Metodologías de integración de *UML-RP*

A través de la literatura consultada se han encontrado diferentes métodos y vías para unificar ¨*UML*¨ y las RP. Aunque existen diferentes formas de hacerlo, de acuerdo a la aplicación que se quiera desarrollar o al problema que se le quiera dar solución; la mayoría de los métodos propuestos de forma general se basan en utilizar las facilidades y capacidades del diseño en ¨*UML*¨ y las ventajas para la representación formal y el análisis de las Redes de Petri.

En (Francesco Basile et al., 2008) se plantea que tanto la planta como los requerimientos de control deben ser primeramente modelados en ¨*UML*¨ de forma separada, con una distinción clara entre la planta y el controlador. Posteriormente, de esos diagramas es posible realizar la traducción a modelos

en RP siguiendo un conjunto de reglas específicas para ello (Grao et al., 2004) o utilizando un software apropiado que realice la traducción directa (Santiago Pérez et al., 2007). O sea, de la descripción informal a un modelo semi-formal en ¨*UML*¨, y de ¨*UML*¨ al formalismo de las RP.

Metodología 1

Una metodología adecuada es la propuesta en (Chang-Pin Lin et al., 2004) (Fig.3.1), la cual comienza con la definición por parte del usuario de los casos de uso reflejando, de acuerdo al tipo de proceso, los procedimientos específicos del flujo de información.

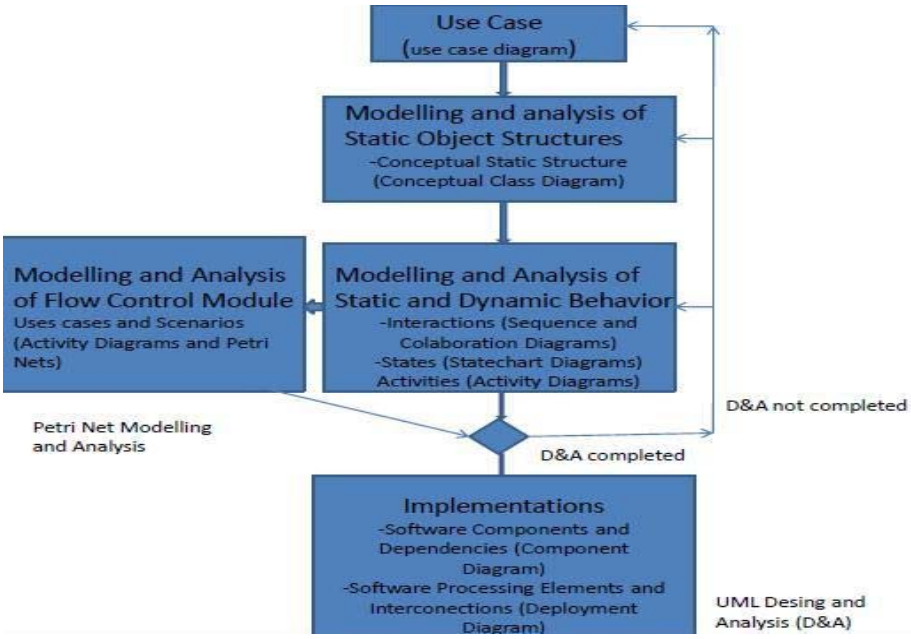


Figura 3.1-Forma de Integración *UML-RP* propuesta en (Chang-Pin Lin et al., 2004)

Este es entonces modelado y analizado en ¨*UML*¨ mediante los diagramas de clases conceptuales para generar los objetos requeridos y su estructura de datos estática. El siguiente paso es usar los diagramas de secuencia, interacción, estados y actividades para analizar el comportamiento estático y dinámico del proceso, teniendo en cuenta aspectos temporales. Para detectar errores de diseño y análisis (¨***D&A***¨ en la figura) durante los procedimientos de

creación de los diagramas, se propone convertir los diagramas de actividades en Redes de Petri, y luego utilizar las técnicas de análisis propias de esta herramienta para verificar la exactitud del sistema que se desarrolle. Finalmente, los diagramas de componentes y de despliegue pueden ser utilizados para llevar a cabo la implementación del módulo de control del flujo de trabajo.

Las técnicas de análisis en Redes de Petri de esta metodología están basadas principalmente en las propiedades estructurales de **Limitación Estructural** y **Vivacidad Estructural**. Luego de obtenido el modelo en RP, este se analiza para ver si la red es estructuralmente viva y limitada de acuerdo a (Murata, 1989). No obstante, prácticamente es más importante descubrir las causas de la ¨no-limitación¨ y la ¨no-vivacidad¨ para resolver los problemas ocurridos durante el diseño del sistema. En caso que la red presente estos problemas se localiza el estado de ¨no-limitación¨ y la transición de la ¨no-vivacidad¨. De acuerdo al tipo de error que se detecte, ya sea estructural o de traducción, es posible hacer cambios en la red sin afectar sustancialmente el sistema principal.

Resulta muy conveniente señalar que, en la metodología anterior se planteó la necesidad de utilizar conjuntamente varios tipos de diagramas ¨*UML*¨ para proveer una descripción completa de todos los aspectos de interés del sistema.

Metodología 2

En otros casos, como en (Francesco Basile et al., 2008) no es necesario ni factible usar tantos diagramas diferentes. Aquí se explica una metodología muy completa para utilizar *UML-RP* en el modelado de un sistema distribuido de control de energía eléctrica en una industria. El carácter del caso de estudio tomado en cuenta hace posible describir la dinámica del proceso solamente a través de *diagramas de casos de uso* (¨***UCDs***-*Use Cases Diagrams*¨) y *diagramas de actividades* (¨***ADs***-*Activities Diagrams*¨). Los aspectos relacionados con el tiempo, la comunicación y la sincronización, generalmente

modelados con diagramas de secuencia, no son considerados predominantes y se incluyen en los ¨*ADs*¨. El procedimiento de diseño (Fig.3.2) no es cerrado, sino que puede ser variado de acuerdo a las necesidades. Presenta cuatro pasos principales a seguir:

1. Modelado del proceso.
2. Diseño del controlador.
3. Refinamiento.
4. Análisis del Sistema de Control.

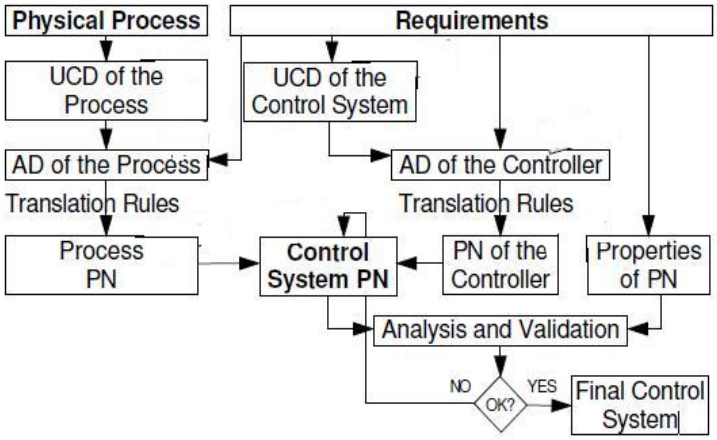


Figura 3.2- Forma de Integración *UML-RP* propuesta en (Francesco Basile et al., 2008).

Modelado del proceso: A grandes rasgos se describe el proceso a controlar a través de un ¨*UCD*¨, cada entidad de interés es considerada como un actor del diagrama. Se usa un ¨*AD*¨ para detallar el funcionamiento del proceso y luego se traduce a un modelo en RP de acuerdo a la (Fig.3.3) y a reglas específicas de traducción, explicadas con más detalles en (Grao et al., 2004).

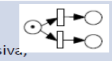
	Diagrama de Actividad	Red de Petri
1	Estados y Actividades, círculos y cuadros redondeados	Lugares de la red
2	Arcos que conectan actividades con alguna condición	Arcos entre lugares con transiciones en el medio. El mensaje tiene que ser traducido a transiciones que asocien distintos componentes de la red.
3	Acciones (cuadros redondeados con pequeños círculos en la parte baja derecha) y Acciones que transfieren cantidades enteras de objetos	Transiciones de salida/entrada de lugares correspondientes Acciones complejas pueden significar una estructura particular de la PN como la presencia de peso en los arcos.
4	Estado inicial	Colocación de un token en la red
5	Rama 	Opción  si la rama es exclusiva, el lugar inicial debe tener solo un token

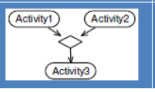
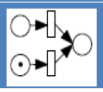
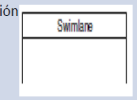
	Diagrama de Actividad	Red de Petri
6	Fusión 	Convergencia 
7	Ramificación 	Paralelismo 
8	Unión 	Sincronización 
9	Lineas de Unión (Swimlanes) 	Están relacionadas con los casos de uso y los actores, traduce cada uno individualmente y entonces hace una composición para unir su funcionamiento

Figura 3.3- Traducción de los ¨*ADs*¨ a *RP*.

Diseño del controlador: Se describen los requisitos con ¨*UCDs*¨, integrando estos con nuevos actores y acciones que representen los requisitos del sistema de control. Los ¨*ADs*¨ referentes a los actores del sistema son usados para detallar su comportamiento, cuando no están lo suficientemente bien expresados con los ¨*UCDs*¨; además proveen una descripción de las especificaciones de control y son el punto de partida para la traducción al lenguaje formal en RP. En este punto los ¨*ADs*¨ permiten definir especificaciones estáticas (evasión de condiciones prohibidas), especificaciones dinámicas (comportamiento deseado del proceso) y especificaciones cualitativas (evasión de ¨*deadlocks*¨[3], limitación física a los recursos del sistema etc.). Estas especificaciones pueden ser traducidas de forma tal que permitan usar las metodologías de diseño típicas para obtener el controlador correcto. Como por ejemplo, un controlador basado en Redes de Petri puede ser directamente inferido de los ¨*ADs*¨ anteriormente diseñados.

Refinamiento: El proceso de refinamiento es una fase delicada del diseño del sistema de control. El control final en RP puede necesitar ser refinado usando técnicas como la colocación de peso en los arcos o realizar un marcaje inicial. Como ejemplo, diferentes estados de los ¨*ADs*¨ se pueden fusionar en un único

[3] deadlock: estancamiento, punto muerto, atascadero, abarrancadero.

lugar de la RP y se podrían diferenciar colocando marcas iniciales en ellos. Además de esto, en dependencia del tipo de red que se use, se podrían tener en cuenta consideraciones de tiempo.

Análisis del Sistema de Control: La ventaja de obtener un modelo formal en Redes de Petri para el Sistema de Control reside en la posibilidad de usar técnicas de análisis basadas en sus propiedades, como se trató en el ejemplo anterior.

Metodología 3

Otra forma de integración es también abordada en (Santiago Pérez et al., 2007), en este caso para la evaluación de ¨***performance***¨ de sistemas de software. Esta metodología es muy útil para evaluar sistemas automatizados como los ¨*SCADAs*¨ y los SCD. Se propone utilizar técnicas de Ingeniería de ¨*Performance*¨ de ¨*Software*¨ (¨*SPE- Software Performance Engineering*¨) con *UML-RP*.

Primeramente la arquitectura y comportamiento del sistema se describe a través de un conjunto de diagramas ¨*UML*¨, que corresponden al diseño del sistema. Este diseño ¨*UML*¨ es anotado de acuerdo a un perfil ¨*OMG*¨ estándar, llamado el diseño anotado. Luego el diseño anotado se convierte en un formalismo de modelación, en este caso Redes de Petri (específicamente en ¨*GSPN*¨), llevándose a cabo un análisis cuantitativo de los principales índices de ¨*performance*¨. Aunque no es objetivo de este trabajo profundizar sobre el ¨*performance*¨ de ¨*software*¨, cabe destacar que este abarca el análisis de la carga de trabajo, utilización eficiente, tiempos de respuesta, rendimiento, entre otros aspectos.

Para la conversión se utiliza la herramienta ***Argo-SPE***[4] sobre el software de modelado *ArgoUML*[5], mencionado en el Cap-1. *Argo-SPE* es un conjunto de consultas de performance que pueden ejecutarse para obtener el análisis cuantitativo del sistema modelado. Cada consulta de performance está relacionada a un diagrama ¨*UML*¨ donde es interpretada, pero su cómputo se obtiene en un modelo de RP.

Metodología 4

Una forma de proceder que se diferencia de las anteriores se describe en (Han and Jafari, 2003). En principio esta metodología es propuesta para un Sistema de Fabricación Flexible (¨*Flexible Manufacturing System,* ***FMS***¨) basado en teoría de agentes inteligentes, pero podría ser utilizada en cualquier tipo de sistema con características similares.

A grandes rasgos, lo que se propone es realizar una descomposición de forma jerárquica de todos los componentes del sistema. En este caso el ¨*FMS*¨ se descompone en Agentes, Componentes Integrados (¨*ICs-Integrated Components*¨) y Componentes Elementales (¨*ECs-Elementary Components*¨) como se muestra en la (Fig.3.4).

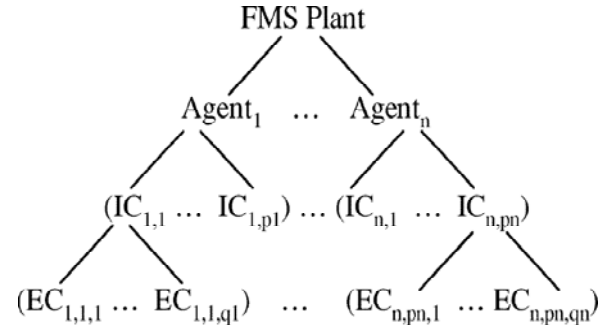


Figura 3.4- Descomposición jerárquica del ¨*FMS*¨

[4] -*plug-in* para *ArgoUML* (Anexo 4)

[5] - Software de modelado en *UML* (anexo 3)

Tanto los ¨*ICs*¨ como los ¨*ECs*¨ pueden ser un conjunto de sensores y actuadores. Mientras que un agente es típicamente definido como un componente que consiste en sensores, actuadores, una unidad de comunicación, un controlador, y es capaz de ejecutar varias tareas.

Primeramente se usan los diagramas de clases de ¨*UML*¨ para modelar el comportamiento estático de toda la planta, creando un diagrama para cada componente. Entre las clases deben existir relaciones de dependencia, asociación y generalización, de forma que se garantice la herencia de clases a sub-clases. Asimismo la definición de los objetos, sus métodos y atributos, deben seguir la jerarquía de la (Fig.3.4), en este caso.

Seguidamente se utilizan las Redes de Petri para realizar el análisis dinámico en los cuatro niveles jerárquicos. Inicialmente se crea una RP modelando cada subsistema de la planta y luego, con las especificaciones de control establecidas mediante una lógica funcional (referidas en el articulo como ¨*Rules Set*¨), se va generando el sistema controlado. Finalmente, el modelo de la planta con control obtenido, garantiza las propiedades de **Vivacidad**, **Repetitividad** y **Limitación estructural**

La diferencia está en que no se realiza la traducción directa de los diagramas ¨*UML*¨ a RP, sino que se crea un modelo diferente en RP de cada clase, siguiendo las especificaciones de estas.

Existen otras formas de combinar *UML-RP*, la mayoría destinadas a modelar y analizar aplicaciones muy específicas. Básicamente las principales diferencias entre ellas radican en la utilización de uno u otro tipo de diagrama para realizar la descripción del sistema. Si en el modelado del sistema se desea incluir todos los aspectos relacionados con el funcionamiento dinámico y estático, especificaciones de tiempo, flujo de información, comunicación, sincronización etc., es preciso entonces usar los diagramas ¨*UML*¨ pertinentes para ello (clases,

secuencia, colaboración, estados). Sin embargo, como se planteó en la segunda metodología, se puede asumir que algunas de estas especificaciones no son de gran relevancia para el objetivo final y obviarlas en un solo diagrama (actividades).

Otra diferencia radica en el tipo de uso que se le dé a la Red de Petri obtenida a partir de modelo ¨*UML*¨. Por ejemplo en la primera metodología se usan las RP para detectar y corregir formalmente posibles errores ocurridos durante el diseño. Sin embargo la implementación se realiza a partir de diagramas ¨*UML*¨. En el segundo método se obtiene un modelo de Sistema de Control en RP, que posteriormente puede ser implementado en un ¨*PLC*¨ de forma directa. Por otra parte en la tercera metodología el uso de las RP es básicamente analizar y validar índices de comportamiento de una aplicación o software específicos.

El tipo de Red de Petri que se use depende en gran medida de los requisitos del sistema y de la implementación final que este tenga. Generalmente se usan las RP ordinarias y las ¨*TPN*¨ , especialmente las ¨*GSPN*¨, aunque en artículos como (Israel Benítez et al., 2010a), se utilizan las redes jerárquicamente extendidas ¨*GHENeSys*¨.

Es importante señalar que no existe un método estándar generalizado a seguir para utilizar *UML-RP*. No obstante se considera que los procedimientos descritos anteriormente abarcan en su conjunto gran parte de los pasos a seguir para emplear de forma eficiente estas herramientas, al menos dentro de la esfera de la Automatización. Una metodología que podría seguirse independientemente del proceso es la mostrada en la (Fig.3.5).

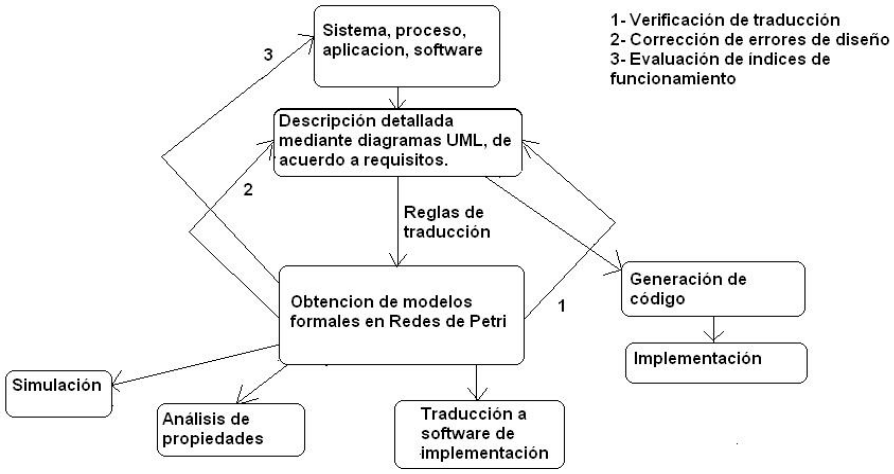


Figura 3.5- Metodología generalizada.

Para comprender más a fondo los procedimientos planteados anteriormente, se presentan algunos ejemplos de sistemas reales modelados con *UML-RP*.

3.3-Presentación de ejemplos.

El primer ejemplo se basa en el empleo de la segunda metodología propuesta y se explica con más detalle en (Francesco Basile et al., 2008).

Se trata de la necesidad de desarrollar un sistema de control de energía eficiente para una planta de producción de alimentos.

Modelado del Proceso: La planta presenta altos requerimientos energéticos, por lo que se necesita llegar a un consenso entre producción y costos de electricidad. Está equipada con una unidad de *cogeneración*[6] que provee la mayor parte de la energía, pero cuando esta no se encuentra disponible o la demanda es muy alta, se utiliza la red nacional. Si la compañía excede un consumo máximo acordado es multada severamente. El equipamiento con que

[6] La cogeneración es el procedimiento mediante el cual se obtiene simultáneamente energía eléctrica y energía térmica útil.

se cuenta es irregular y obsoleto. Además debido a la ausencia de un sistema remoto de medición y control es difícil administrar el consumo en todos los locales. Por esto se requiere un sistema que supervise el estado de consumo y que aplique políticas de restricción de energía según el nivel de producción.

Diseño del Controlador: La unidad de cogeneración tiene que ser modelada como una máquina insegura con un estado propio de condición de fallo. Los consumidores en la planta son localmente controlados y la red eléctrica nacional se modela como una fuente infinita de energía.

El principal requisito es evitar, a través del sistema de control, que se excedan los límites de consumo acordados, y en caso de que esto ocurra activar las políticas de restricción. En el ¨*UCD*¨ (Fig.3.6) se representa una descripción funcional del comportamiento del sistema de control, y a partir de este se obtiene el ¨*AD*¨, en el cual se introducen todas las especificaciones que necesita el sistema para operar correctamente.

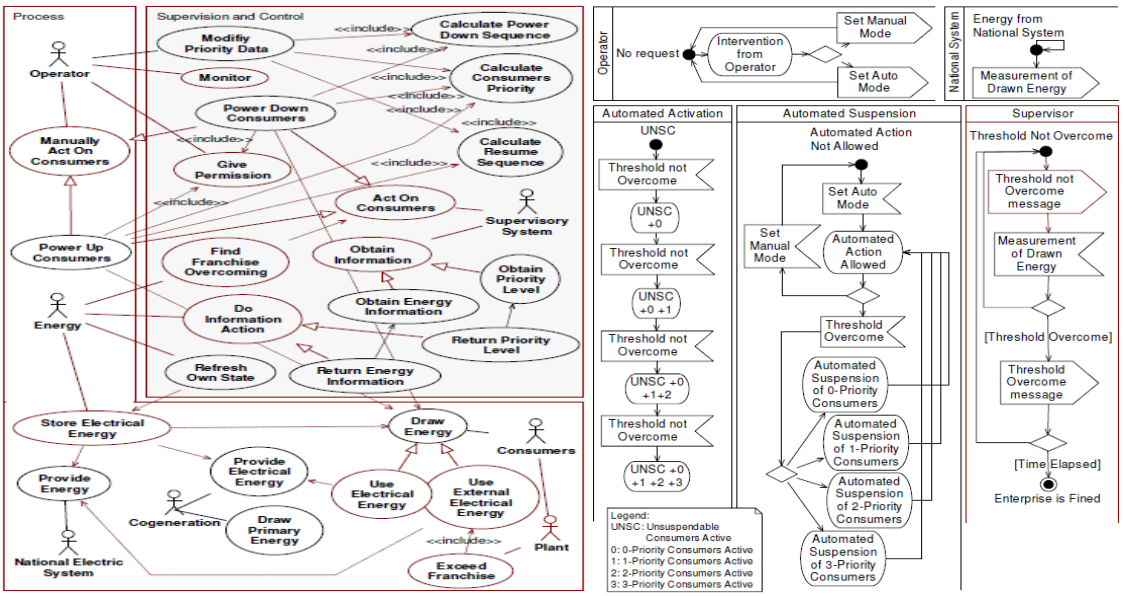


Figura 3.6- Descripción Funcional del Sistema de Control en ¨*UCDs*¨ y ¨*ADs*¨.

Finalmente, a partir de los ¨*ADs*¨ se obtiene un modelo en RP del sistema de control en lazo cerrado (Fig.3.7), donde las líneas negras representan el

proceso, y las grises la acción de control. Los rectángulos vacios representan todos los eventos que pueden ser controlados, mientras que los rellenos de negro son eventos incontrolables.

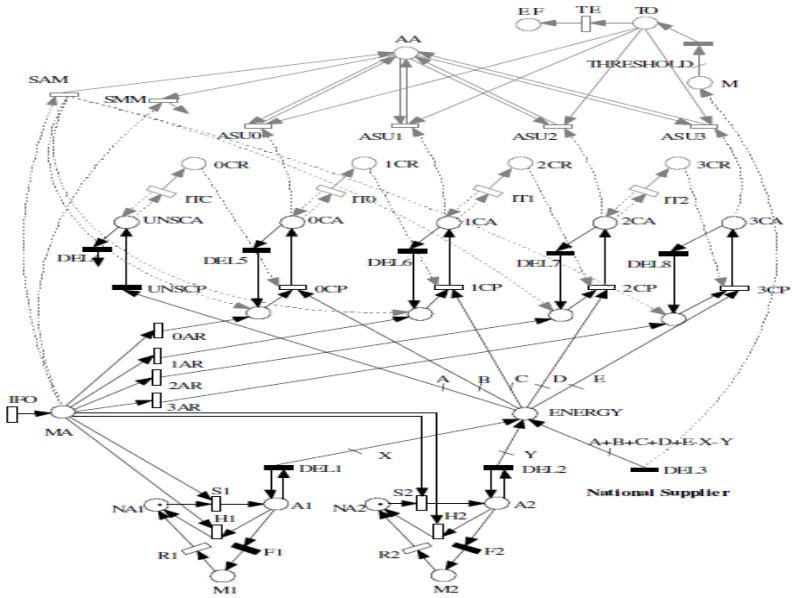


Figura 3.7- Modelo en RP del Sistema de Control.

Refinamiento: El proceso de refinamiento es llevado a cabo durante la fase de construcción de la Red de Petri. Mediante este se colocan pesos de arcos específicos en algunos puntos.

Análisis del Sistema de Control: Como se expuso anteriormente, luego de obtenido el modelo en PN´s se pueden verificar todas las propiedades. Además con el uso de software apropiado, se puede simular e implementar el sistema.

En el segundo ejemplo se modela el funcionamiento de un ¨*FMS*¨ (Fig.3.8) utilizando *UML-RP* y además teoría de Agentes Inteligentes (K.M. Kavi et al., 2002). Este ejemplo es detallado en (Israel Benítez et al., 2010a).

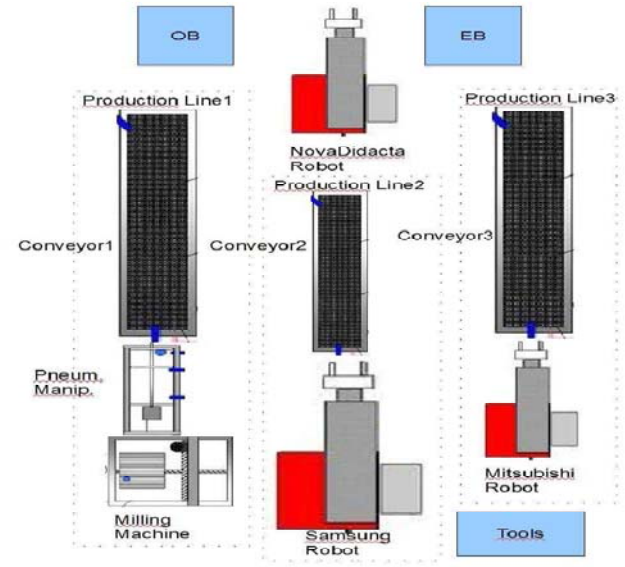


Figura 3.8- Esquema del ¨*FMS*¨.

En el sistema se identifican tres niveles jerárquicos: sistema, línea de producción y máquinas.

Sistema: Compuesto por Agentes de Producción (¨*PAg- Production Agents*¨), los cuales son responsables de la planificación y aseguramiento de la producción. Deben organizar todas las piezas que se necesiten producir en una lista.

Línea de Producción: Compuesto por Agentes de Control (¨*CAg- Control Agents*¨), que se encargan de la planificación de las operaciones necesarias en la producción de las piezas, y el orden en que estas se fabrican.

Máquinas: Compuesto por Agentes de Recursos (¨*RAg- Resource Agents*¨), encargados de controlar las acciones durante la fabricación y detección de fallas de todo tipo.

Para la creación del ¨*UCD*¨ (Fig.3.9) se identifican todos los actores del sistema: Usuario, Administrador, Cliente, ¨*PAg*¨, ¨*CAg*¨, ¨*RAg*¨, esteras de transportación, máquina de cortes, robots, manipulador neumático. En el diagrama de clases se

modela la estructura estática del sistema, identificando las relaciones entre las clases, sus atributos, métodos y las características de los agentes.

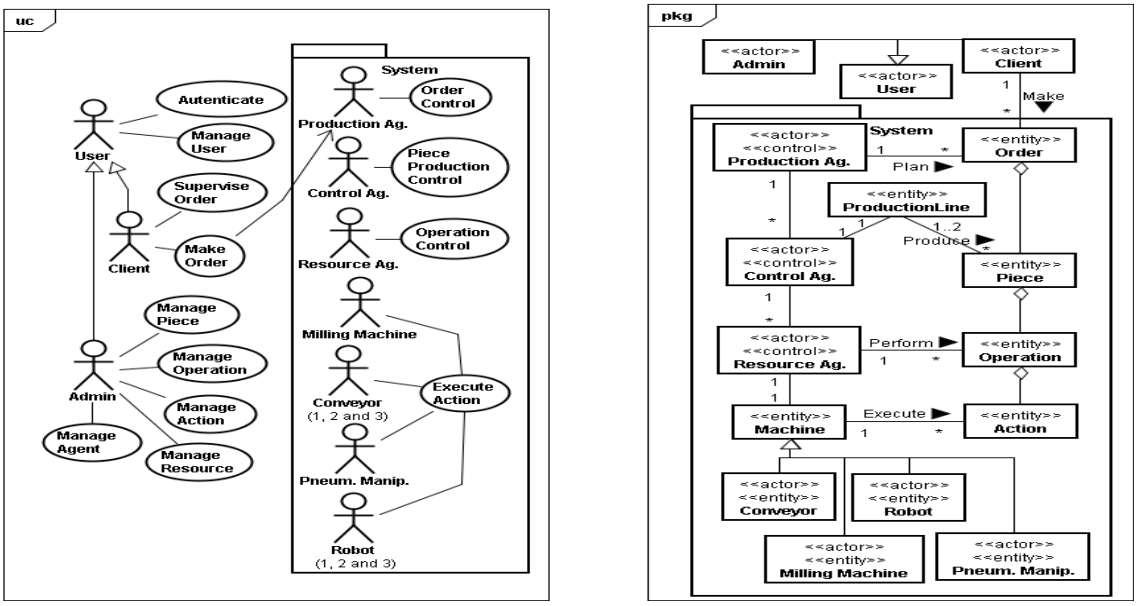


Figura 3.9- Diagrama de casos de uso y de clases.

Se usa un diagrama de estado para poder analizar el comportamiento de un objeto específico, incluyendo las reglas para cada transición. Adicionalmente se incluyen un diagrama de secuencia para identificar como los grupos de objetos interactúan entre sí a lo largo del tiempo, ilustrar el intercambio de mensaje y las relaciones entre los agentes. Posteriormente, a partir del diagrama de secuencia se obtiene el modelo en RP. En la (Fig.3.10) se muestra el modelo para las esteras transportadoras del ¨*FMS*¨.

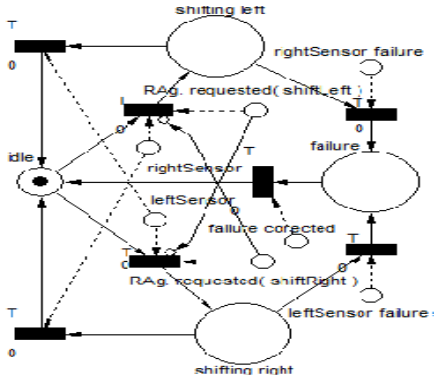


Figura 3.10- Modelo en Redes de Petri ¨*GHENeSys*¨ de las esteras transportadoras.

En este caso el tipo de red que se usa son las ¨*GHENeSys*¨.

El tercer y último ejemplo de trata de la modelación de un Sistema de Control Anti-Incendios para un central azucarero, explicado en (Bastidas et al., 2003).

El sistema tiene como propósito principal detectar el fuego mediante el monitoreo de un conjunto de sensores (temperatura, fuego, humo) y actuar sobre los dispositivos tecnológicos para minimizar el potencial de daños. En caso de fuego activar los extintores más cercanos a este y comunicar a otros sistemas de la planta como el ¨*HVAC*¨ (¨*Heating, Ventilation and Air Conditioning*¨), sistema de luces, control de acceso etc., para que estos puedan desarrollar las estrategias pertinentes. Además es necesario avisar a la brigada Anti-Incendio.

Los ¨*UCDs*¨ son construidos teniendo en cuenta los principales actores y las relaciones entre estos y el sistema (Supervisión+ Sistema de Control+ Planta).

Cada caso de uso es detallado en un diagrama de colaboración indicando la relación entre el Sistema Supervisorio y los sub-sistemas de control local (previamente identificados). Para modelar los objetos, flujo y procesamiento de la información, se construye un diagrama de clases (Fig.3.11)

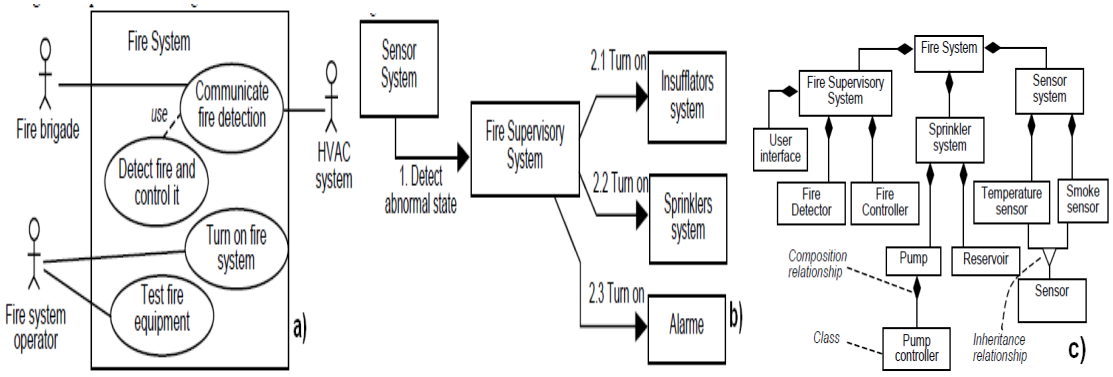


Figura 3.11- a)-¨*UCD*¨, b)-diagrama de colaboración, c)-diagrama de clases.

Lo siguiente es construir un modelo en RP de cada clase ¨*UML*¨ (Fig.3.12). El tipo de red para cada clase no se especifica, ya que puede variar de acuerdo a las características de las clases y los sistemas. La única restricción es que las interfaces entre los objetos deben ser modeladas por disparo de transiciones en todas las clases.

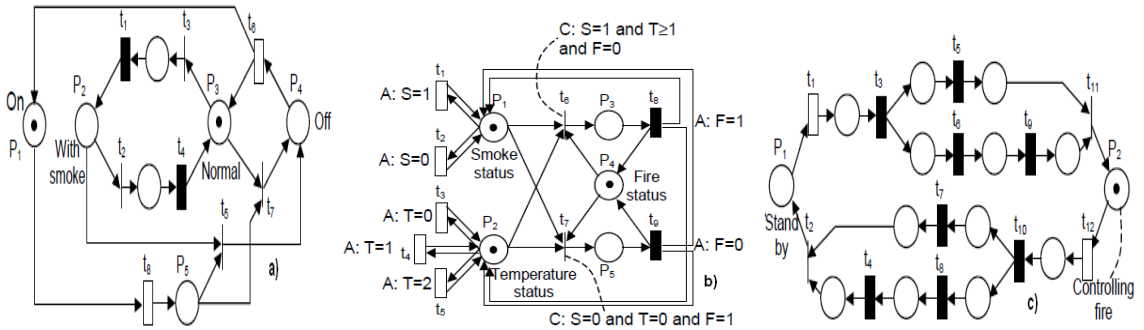


Figura 3.12- a)- Sensor de Humo, b)- Detector de Fuego, c)- Controlador de Fuego.

Para el modelado del flujo de actividades, cada caso de uso es desglosado en ¨*ADs*¨, que muestran las actividades llevadas a cabo por el sistema. Luego los ¨*ADs*¨ se convierten a modelos en RP (Fig.3.13). Finalmente se construye un diagrama de colaboración mediante el cual se puede implementar el módulo de control del sistema.

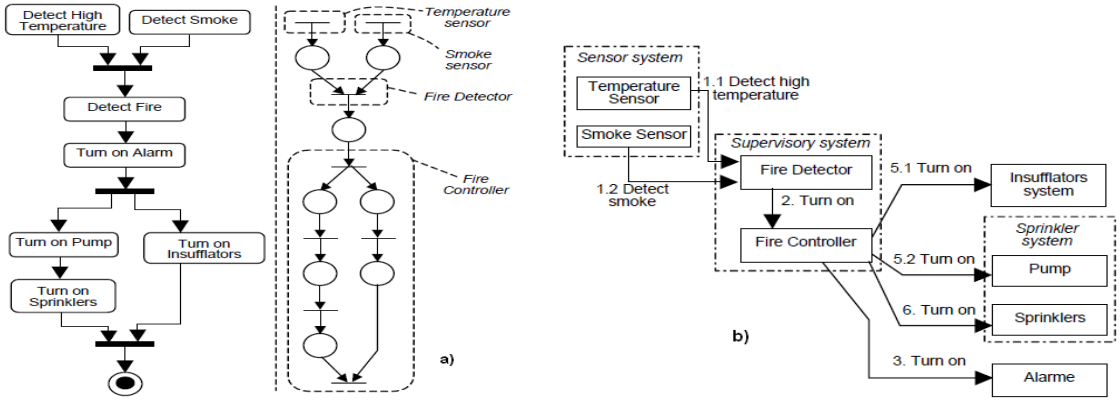


Figura 3.13- a)- Conversión de ¨*ADs*¨ a RP, b)- diagrama de colaboración final.

Luego de conocer varias formas de integrar *UML-RP* para el modelado de sistemas automatizados y haber observado el uso práctico que poseen mediante

ejemplos reales, sería muy útil conocer las principales ventajas y desventajas que trae consigo esta integración.

3.4- Ventajas y Desventajas del uso de UML-RP.

Ventajas

Con la unión de ¨*UML*¨ y las Redes de Petri se satisface la necesidad de obtener una herramienta capaz de modelar la red de automatización completa de una compañía o industria, abarcando todos los niveles jerárquicos y con un elevado nivel de abstracción y robustez.

Su utilización práctica ha demostrado la simplicidad y compatibilidad con las herramientas de diseño de sistemas automatizados industriales y la posibilidad de utilizar los nuevos estándares presentes en las normas IEC-61131 y IEC-1499, incluyendo sistemas de comunicación industrial.

Por otra parte, con *UML-RP* es posible detectar la mayoría de los errores lógicos ocurridos en los sistemas durante una etapa temprana de su desarrollo, por lo que los costos de diseño y mantenimiento se reducen drásticamente. Esto es muy útil en sistemas de hardware y software muy complejos, donde es usualmente difícil descubrir errores lógicos de diseño que ocurren a través de un largo tiempo de desarrollo, pruebas y frecuentes caídas del sistema.

Como ¨*UML*¨ cuenta con capacidades estándares para el diseño orientado a objetos y las Redes de Petri con formalismos de análisis, los desarrolladores de sistemas explotan los beneficios de la formalidad mediante el uso de las reglas de traducción y "*software*" específicos. Se pueden analizar y ejecutar los modelos sin tener que ser expertos en Redes de Petri. De este modo los beneficios de ¨*UML*¨ para describir el funcionamiento de sistemas son plenamente utilizados, mientras que el análisis se realiza de forma más eficiente en el dominio de las RP.

También el uso de *UML-RP* permite realizar representaciones más compactas de los modelos, ya que solo se necesita modelar los estados que son importantes para la secuencia lógica del sistema. Esto posibilita simplificar las especificaciones de complejos Sistemas de Automatización Distribuidos y reducir el tiempo de diseño.

Debido a la naturaleza gráfica que presenta esta integración es posible realizar la simulación de los modelos obtenidos, mediante la cual se puede comprende mejor la dinámica del sistema y encontrar incongruencias en el mismo.

Es posible además, combinando *UML-RP* con teoría de agentes inteligentes, modelar sistemas multi-agentes para aplicaciones de control, especialmente en ¨*FMS*¨. Esto permite optimizar la interrelación entre los agentes para alcanzar una mayor eficiencia y flexibilidad en la producción y el chequeo de las propiedades del modelo para garantizar la ausencia de ¨*deadlocks*¨. La integración asegura la validación del modelo y que los requerimientos del usuario sean analizados y producidos inmediatamente. Por consiguiente el nivel de inteligencia en la toma de decisiones automatizada se incrementa.

Desventajas

Una de las principales desventajas que presenta la integración *UML-RP* es la poca difusión que tiene entre las principales compañías que se dedican a la automatización industrial. Aunque se han implementado varias aplicaciones con resultados positivos, muchos diseñadores prefieren seguir utilizando metodologías basadas en otras herramientas.

Además, es difícil encontrar una amplia documentación sobre el tema que permita desarrollar metodologías generalizadas para cualquier tipo de aplicación. De igual manera, tampoco se ha generalizado un ¨*software*¨ que soporte este tipo de integración. El diseño y análisis del sistema se realizan, en la mayoría de los casos, en software diferentes para RP y ¨*UML*¨.

Conclusiones del Capítulo

1. Existen diversas metodologías que permiten integrar eficientemente ¨*UML*¨ y las Redes de Petri para modelar y analizar Sistemas Automatizados.
2. Aunque el uso conjunto de estas herramientas ha ido aumentando dentro del sector industrial, su desarrollo pleno está limitado por la falta de documentación y un ¨*software*¨ generalizado que permita utilizar ambas a la vez.

Conclusiones

1. Debido a las características y a la complejidad que presentan los sistemas automatizados actuales, se hace necesario emplear herramientas de modelado eficientes como ¨*UML*¨ y Redes de Petri.
2. Tanto las RP como ¨*UML*¨, de forma independiente, presentan características positivas para modelar Sistemas Automatizados. No obstante con ¨*UML*¨ es difícil describir el comportamiento dinámico de muchos sistemas, mientras que las RP carecen de la capacidad de especificación y representación de ¨*UML*¨
3. El uso en conjunto de *UML-RP* aprovecha las facilidades para el diseño orientado a objetos de ¨*UML*¨ y las combina con las técnicas de análisis formales de las RP, logrando obtener una herramienta capaz de modelar y/o analizar eficientemente cualquier tipo de Sistema Automatizado.

Recomendaciones

1. En futuras investigaciones profundizar sobre el uso en conjunto de ¨*UML*¨, Redes de Petri y teoría de Agentes Inteligentes
2. Se recomienda investigar si es posible emplear ¨*UML*¨ y Redes de Petri de manera inversa a la abordada en este trabajo. O sea, a partir de modelos en RP obtener diagramas en ¨*UML*¨

Referencias Bibliográficas

ANTONIO BARRIENTOS, PEÑIN, L. F., BALAGUER, C. & ARACIL, R. 1997. *Fundamentos de Robótica,* Madrid, Universidad Politecnica de Madrid.

BASTIDAS, G., VILLANI, E., F.JUNQUEIRA & MIYAGI, P. E. 2003. Open Distributed Supervisory System Design using Petri nets. *Escola Politécnica, Universidade de São Paulo, São Paulo, Brazil*.

BEREA, J. 2006. Modelado con Redes de Petri. *Departamento de Ingenieria de Sistemas y Automática. Universidad de Vigo, España*.

BOOCH, G., RUMBAUGH, J. & IVAR JACOBSON 1998. El Lenguaje Unificado de Modelado. *Addison Wesley.*

CASTELLANOS, E. I. 2008. "Sistemas de Automatización" Editorial Feijóo ed. Santa Clara.

COCA, L. V. 2007. *Identificacion y Representacion de las Principales Variables de la Etiquetadora KOSME de la Ronera Central "Agustin Rodriguez Mena".* Universidad Central "Martha Abreu" de Las Villas.

CHANG-PIN LIN, YI-PIN LIN & JENG, M. D. 2004. Design of Intelligent Manufacturing Systems by Using UML and Petri Net. *Department of Electrical Engineering. National Taiwan Ocean University,* Proceedings of the 2004 IEEE International Conference an Networking, Sensing & Control.

DAVID, R. & ALLA, H. 1992. Petri Nets and GRAFCET: Tools for Modelling Discrete Event Systems. *New York : PRENTICE HALL Editions*.

DU, Y., JIANG, C. & ZHOU, M. 2007. Modeling and Analysis of Real-Time Cooperative Systems Using Petri Nets. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 37, NO. 5*.

FRANCESCO BASILE, CHIACCHIO, P. & GROSSO, D. D. 2008. Modelling automation systems by UML and Petri Nets. *Proceedings of the 9th International Workshop on Discrete Event Systems Göteborg, Sweden*.

GHOMRI, L. & ALLA, H. 2007. Modeling and Analysis using Hybrid Petri Nets. *Laboratoire d'Automatique de Tlemcen, Algeria and Laboratoire d'Automatique de Grenoble, San Martin, France*.

GÓMEZ, J. V. 2010. *Sistemas automatizados: Vida para las empresas* [Online]. Ciudad Mexico. Available: www.logistica.enfasis.com [Accessed 10/01 2011].

GRADY BOOCH, IVAR JACOBSON & JAMES RUMBAUGH 2003. El Proceso Unificado de Desarrollo de Software. *Addison Wesley, Series Editor.*, Rational Software Corporation.

GRAO, J. P. L., MERSEGUER, J. & CAMPOS, J. 2004. From UML Activity Diagrams To Stochastic Petri Nets: Application To Software Performance Engineering. *Departamento de Informática e Ingeniería de Sistemas, Universidad de Zaragoza. Zaragoza, España.*

HAN, W. & JAFARI, M. A. 2003. Component and Agent-Based FMS Modeling and Controller Synthesis. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, VOL. 33, NO. 2*.

HERRERA, F. & RODRIGUEZ, F. 1990. *Proyectos de Automatización*, Ministerio de Educacion Superior.

IEEE 2007. IEEE Standard for SCADA and Automation Systems. USA: IEEE Power Engineering Society.

ISRAEL BENÍTEZ, BRUNO MENDES, JOSÉ RÚBEN SICCHAR, DANIELLE GORDIANO VALENTE & FREITAS, R. C. D. 2010a. A DESIGN METHOD FOR FLEXIBLE MANUFACTURING SYSTEM BASED ON PETRI NETS AND UML. *Universidade de Oriente (FIE-UO, Cuba) and Universidade do Estado do Amazonas (EST-UEA, Brasil)*.

ISRAEL BENÍTEZ, JOSE R. SILVA, SANTIAGO SOTOMAYOR & JOSE R. SICHAR 2010b. MODELADO FORMAL DE LA AUTOMATIZACIÓN DEL MOLINO DE CLÍNKER. *Ciencia en su PC,* # 3**,** p. 71-85.

ISRAEL BENÍTEZ, KSENIA ARIAS, SILVA, R. & ESTRADA, Y. 2002. *¨Las Redes de Petri IEC-1131 compatibles en la Automatización¨.* Departamento de Control Automático. Monografia, Universidad de Oriente.

JOAQUIN L. FERNÁNDEZ, SANZ, R., BARBOSA, P. M. & DIEGUEZ, A. R. 2007. ESPECIFICACIÓN Y MODELADO DE TAREAS DE ALTO NIVEL PARA ROBOTS MÓVILES. APLICACIÓN A LA INTERPRETACIÓN DE ÓRDENES VOCALES. *Departamento de Ingeniería de Sistemas y Automática, Campus Lagoas-Marcosende Universidad de Vigo 36200 Vigo, Spain*.

K.M. KAVI, M. ABORIZKA & KUNG, D. 2002. A Framework for Designing, Modeling and Analyzing Agent Based Systems. *Proceedings of the 5th International Conference on Algorithms & Architectures for Parallel Processing.* .

KOTZIAN, J. & SROVNAL, V. 2004. Design and Optimization of Distributed Control System using UML Model. *Proceedings of the 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS'04).* Technical University of Ostrava. Faculty of Electrical Engineering and Computer Science. Department of Measurement and Control.

LARMAN, C. 2004. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. *Addison Wesley Professional*.

LUEBBERS, P. G. 2004. Neural Networks. *¨PROCESS/INDUSTRIAL INSTRUMENTS AND CONTROLS HANDBOOK¨* [Online].

MORA, F. 2002. UML: Lenguaje Unificado de Modelado. *DCCIA, Universidad de Alicante.*

MORENO, E. G. 1999. Automatización de procesos industriales: robótica y automática Valencia: Universidad Politecnica de Valencia.

MURATA, T. 1989. Petri Nets: ¨Properties, Analysis and Applications¨. *Proceedings of the IEEE, vol 7, NO. 4*

O'BRIAN, L. 2007. "Providing the business value proposition for automation". *Revista ABB- ABB Special Report. Automation Systems.*

O.R.NATALE, L.GLIELMO & F.VASCA 2002. Data modeling for batch processes data with application to winemaking. *Proceedings of the 41st IEEE Conf. on Decision and Control, vol. 4, pp. 4101–4106*.

ORALLO, E. H. 2002. ¨El Lenguaje Unificado de Modelado (UML)¨. ***ACTA*** *(Autores Cientifico-Técnicos y Académicos)* ***Universidad Politécnica de Valencia***.

PENG, S. S. & ZHOU, M. C. 2004. Ladder Diagram and Petri-Net-Based Discrete-Event Control Design Methods. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS*.

REISIG, W. & ROZENBERG, G. 2005. *"Informal Introduction to Petri Nets".* Humboldt-University zu Berlin, Germany and Leiden University, Netherlands.

ROZENBERG, G. & ENGELFRIET, J. 2005. *¨Elementary Net Systems¨.*

SALMON, A. O. 2009. *¨Herramienta para la enseñanza de modelado de sistemas de automatización con redes de Petri GHENeSys¨.* Tesis en opción al Título de Master en Informática Educativa., Universidad de Oriente.

SANTIAGO PÉREZ, MARIO DISTEFANO, ANTONIO PASEO & RANZUGLIA, A. 2007. UML y REDES DE PETRI EN LA EVALUACION DE PERFORMANCE DE SISTEMAS. *Grupo de Modelos Industriales Paralelos, Facultad Regional Mendoza, Universidad Tecnológica Nacional. Mendoza, Argentina*.

SARASOLA, A. F. 2003. "PROYECTO DE SISTEMA DE AUTOMATIZACIÓN. ESTACIÓN PARA LLENADO Y TRANSPORTE DE LÍQUIDO". GENIA (Entornos Integrados de Automatización) del Área de Ingeniería de Sistemas y Automática de la Universidad de Oviedo.

SARASOLA, A. F. 2008. "Modelado y programación de una célula de fabricación flexible mediante Redes de Petri". GENIA (Entornos Integrados de Automatización) del Área de Ingeniería de Sistemas y Automática de la Universidad de Oviedo.

T. TSAI & R. SATO 2004. A UML model of agile production planning and control system. *Computers in Industry, vol. 53, pp. 133–152*.

THRAMBOULIDIS, K. & TRANORIS, C. 2003. Integrating UML and the Function Block Concept for the Development of Distributed Control Applications. *Electrical and Computer Engineering University of Patras. Patras, Greece*.

VEGA, V. M. 2007. "Desarrollo, Tendencias Actuales y Perspectivas de los Sistemas SCADA". *Conferencia impartida en el marco del evento UCiencia 2007.*

VELÁSQUEZ, J. 2004. Como justificar Proyectos de Automatización. *Industrial Data* [Online], vol.7. [Accessed 28/2/2011].

VYATKIN, V., KARRAS, S. & PFEIFFER, T. 2005. Architecture for Automation System Development Based on IEC61499 Standard. *3rd IEEE Intemational Conference on Industrial Informatics (INDIN),* University of Auckland. Department of Electrical and Computer Engineering, New Zealand, .

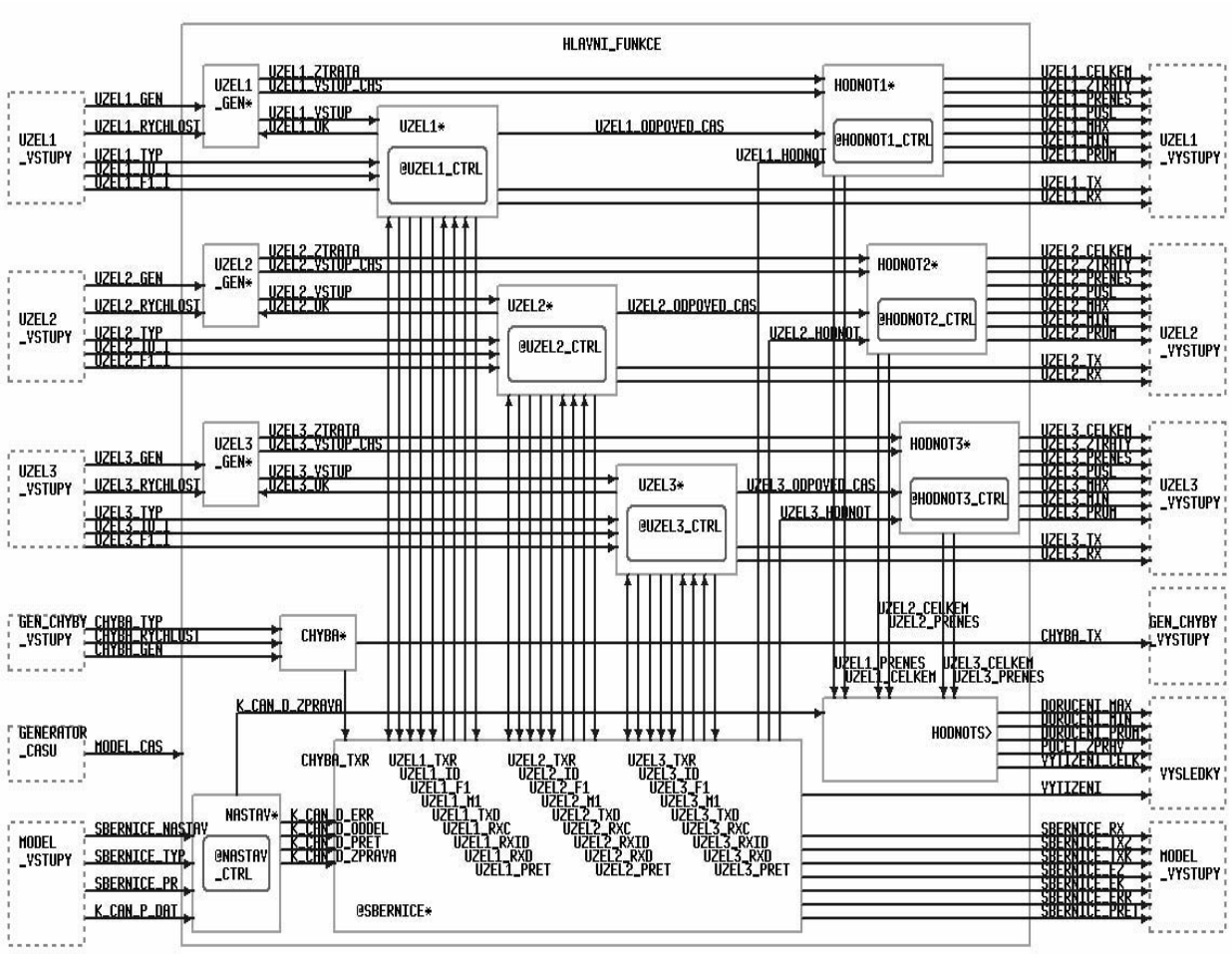
XUSHENG YANG, WANXING SHENG & WANG, S. 2006. A New Architecture for Distribution Control and Automation System. *Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications*

[Online].

Anexos

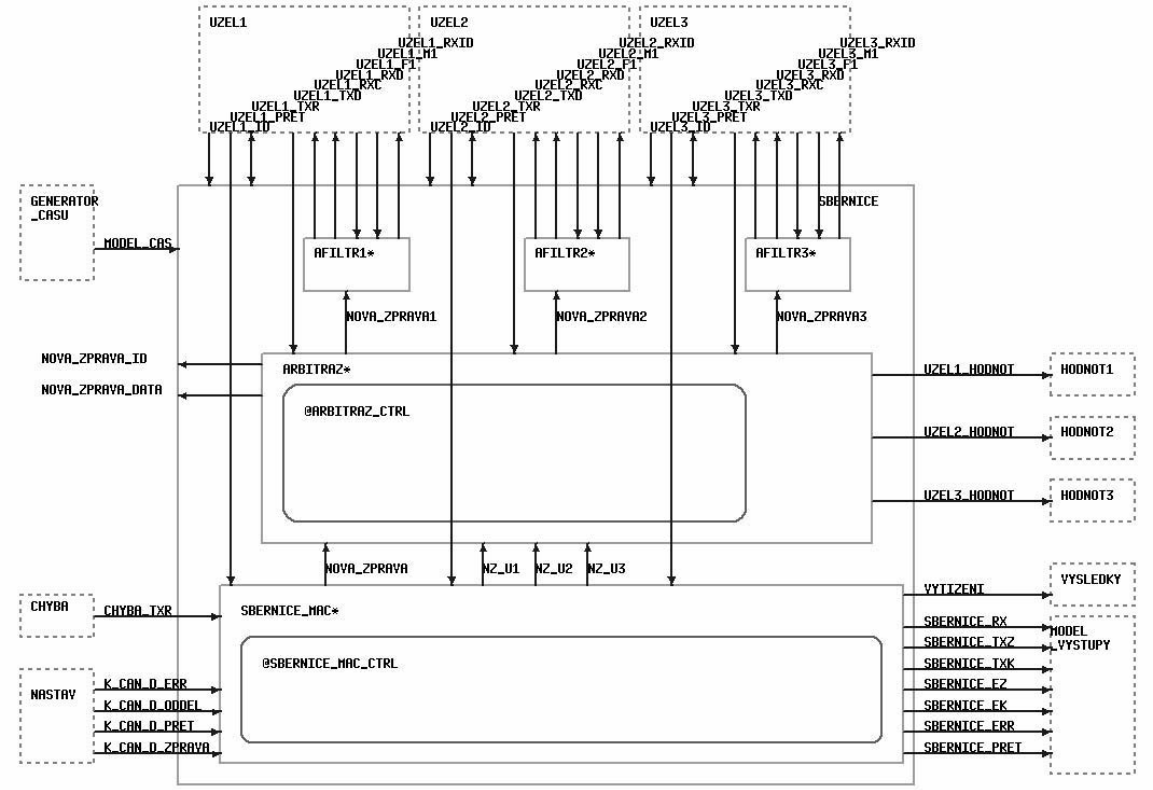
Anexo 1

Diagramas de Actividades del modelo del Sistema de Control Distribuido



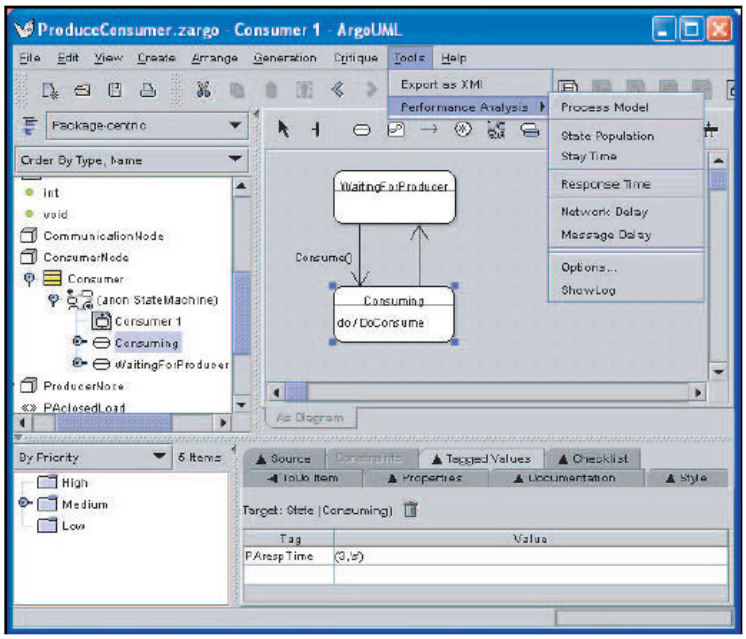
Anexo 2

Diagramas de Actividades del modelo del bus.



Anexo 3

ArgoUML: ArgoUML es una aplicación de diagramado de ¨*UML*¨ escrita en ¨*Java*¨ y publicada bajo la Licencia ¨*BSD*¨ (¨*Berkeley Software Distribution*¨). Dado que es una aplicación "*Java*", está disponible en cualquier plataforma soportada por "*Java*".



Características:

1. Construido en diseños críticos suministra una revisión no obstructiva del diseño y sugerencias para mejoras.
2. Interfaz de módulos extensible.
3. Soporte de Internacionalización para inglés, alemán, francés, español y ruso.
4. Restricciones OCL para Clases.
5. Soporte para lenguajes de generación de Código: Java, PHP, Python, C++ y Csharp (C#)
6. Permite realizar Ingeniería Inversa.
7. Disposición (layout) automática del diagrama de clases.

8. Generación de ficheros PNG, GIF, JPG, SVG, EPS desde diagramas.
9. Soporte para comentarios para múltiples elementos.
10. Todos los diagramas de la versión ¨*UML*¨1.4 están soportados.

Anexo 4

ArgoSPE: ArgoSPE es una herramienta desarrollada por la universidad de Zaragoza (España) para la evaluación de "*performance*" de sistemas de "*software*". Ha sido implementada como un conjunto de módulos Java, que son ¨*plugged*¨ en la herramienta de código abierto *ArgoUML*. Desde el punto de vista del usuario, ArgoSPE es un conjunto de consultas de performance que pueden ejecutarse para obtener el análisis cuantitativo del sistema.

Puede usar directamente la herramienta ¨*GreatSPN*¨ (¨*Great Stochastic Petri Net*¨) desarrollada por la universidad de Turín (Italia) para computar las métricas especificadas usando los modelos ¨*GSPN*¨, que genera automáticamente.