Aplicación de Técnicas de Inteligencia Artificial al Diseño de Sistemas Informáticos para el Control de Sistemas de Producción

Pedro Rafael MURO MEDRANO

Tesis Doctoral

Departamento de Ingeniería Eléctrica e Informática
Centro Politécnico Superior
Universidad de Zaragoza
María de Luna 3, 50015, Zaragoza

Abril 1990

Agradecimientos

A lo largo de estos años han sido numerosas las personas que, de alguna u otra forma, me han ayudado a llevar a buen término esta tesis, demasiadas para poder ser mencionadas de forma completa sin olvidos involuntarios. Científicamente mi agradecimiento se reparte entre las personas de los dos centros donde he llevado a cabo mis trabajos:

A mis compañeros y profesores en el grupo de Ingeniería de Sistemas e Informática de la Universidad de Zaragoza. A Javier Martínez, mi director de tesis, y a Manuel Silva por haber creado un atractivo ambiente de investigación. Al resto de miembros del GISI, antiguos y nuevos, que han sido compañeros de trabajo y tertulia y amigos, siempre dispuestos a hechar una mano. Me gustaría agradecer de manera especial a Jose L. Villarroel; su contribución en el desarrollo de parte de este trabajo es incontable, sin la ayuda de sus apasionadas discusiones, KRON nunca hubiera sido posible.

A la Universidad de Carnegie Mellon por proporcionarme la posibilidad de trabajo en un ambiente científico extraordiariamente enriquecedor, que ha sido fundamental en mi desarrollo como investigador. Y al profesor Mark S. Fox, director de Intelligent Systems Laboratory del Robotics Institute, por haberme aceptado en su laboratorio. A los miembros del proyecto PHOENIX durante mi estancia en Carnegie Mellon y particularmente a Dirk Matthys y Jean-Yves Potvin cuya amistad, comentarios y discusiones supusieron para mi una ayuda inestimable. Me gustaría agradecer de forma muy especial a Stephen F. Smith, director del proyecto PHOENIX, por sus facetas como maestro, como investigador y como director de investigación; su interés y sus enseñanzas me proporcionaron un estímulo fundamental.

Me gustaría agradecer también a la Diputación General de Aragón--CONAI por su soporte económico, tanto para mi estancia en España como en Estados Unidos.

Finalmente un profundo agradecimiento a mi padre que me proporcionó su ayuda y apoyo incondicional a lo largo de todos estos años de estudio, desafortunadamente este agradecimiento llega ya demasiado tarde. Al resto de mi familia, mi madre y mis hermanas, por su ayuda y apoyo total. A Mariangeles por su amistad, consejos, aliento, ayuda y compañía.

A María, mi madre

A Mariani, Maritere, Josemi, Marisantos, Esperanza

Y a Mariangeles

Indice

Lista de Figuras

0.1 Introducción

Un sistema de producción persigue el objetivo de elaborar una serie de productos para satisfacer la demanda del cliente (medida basicamente por la calidad del producto y su fecha de entrega) obteniendo, al mismo tiempo, un beneficio para el fabricante. Para conseguir estos objetivos, un sistema de producción debe poseer un considerable grado de automatización y flexibilidad. La organización del control de sistemas de fabricación ha sido planteada en la literatura según diversos criterios jerárquicos de los que pueden abstraerse los siguientes niveles:

Planificación de producción : Esquemáticamente hablando, a partir de las órdenes del cliente y el estado y restricciones de la planta, su objetivo es generar la lista de órdenes de producción a ser realizadas en cada periodo de tiempo. Cada orden implica una serie de operaciones de fabricación, ensamblaje, transporte, almacenamiento, etc.

Planificación de operaciones : Involucra la asignación de tiempos para llevar a cabo las operaciones especificadas, junto con las asignaciones de los recursos de producción necesarios. Dependiendo de la filosofía de producción adoptada, la toma de decisiones a este nivel se lleva a cabo con antelación al proceso de producción, en el momento de la ejecución, o (más típicamente) implica alguna combinación de ambas aproximaciones.

Coordinación de los elementos de la planta (máquinas, dispositivos de transporte, robots, almacenes, etc.) de forma que cooperen para llevar a cabo las operaciones. La función del coordinador es activar las acciones locales requeridas para ejecutar cada operación.

Control local de los elementos de la planta en la ejecución de las tareas asociadas a acciones locales.

El reto de construir aplicaciones "inteligentes" para problemas de control de procesos y toma de decisiones en sistemas de producción requiere una "hábil" combinación de diversas tecnologías y metodologías. La inteligencia artificial ha surgido en los últimos años como un área extraordinariamente prometedora, principalmente en el dominio de *metodologías basadas-en-el-conocimiento.* Esta tesis investiga la aplicación de técnicas y métodos de inteligencia artificial a los problema mencionados de control de procesos y toma de decisiones de producción. Más concretamente, el trabajo se focaliza en dos aspectos: la representación del conocimiento y la toma de decisiones de control.

En el primer capítulo se aborda la problemática genérica de representación del conocimiento para sistemas de eventos discretos. El estudio se centra en las aproximaciones desarrolladas según dos perspectivas: técnicas de representación del conocimiento de Inteligencia Artificial y técnicas de modelado basadas en redes de

Petri. En el área de la Inteligencia Artificial existen potentes herramientas para la representación en diversos dominios. Sin embargo se hecha en falta la disponibilidad de técnicas que aborden de forma específica la problemática que supone el modelado del conocimiento de sistemas de eventos dicretos. En este capítulo se realiza un repaso de distintas aproximaciones conocidas centrando la atención en las representaciones basadas en "frames". Por otra parte, las redes de Petri constituyen una elegante y potente familia de herramientas para la representación de sistemas dinámicos concurrentes de eventos discretos. Sin embargo, por si solas carecen de características importantes para la representación de otro tipo de conocimiento más genérico.

Tras analizar ambos tipos de técnicas y los beneficios de su utilización conjunta, se presenta la aproximación adoptada en este trabajo. La herramienta de representación propuesta, facilita la inclusión del conocimiento necesario para soportar diferentes funciones y, por otra parte, posibilita el tratamiento de las particularidades inherentes a los sistemas de eventos discretos. Dicha herramienta, denomina **KRON**, aborda de forma específica la problemática mencionada y está inspirada en la integración de tres aproximaciones complementarias: técnicas de representación basadas en "frames", programación orientada a objetos y redes de Petri de alto nivel. La programación orientada a objetos proporciona la base de una metodología de modelado; los "frames" aportan un esquema genérico de representación del conocimiento y las redes de Petri de alto nivel constituyen un formalismo que permite manejar las características dinámicas.

KRON está construido utilizando un lenguaje de representación basado en "frames", que se amplía con primitivas sobre métodos, relaciones, funciones y objetos que permiten integrar las características de las redes de Petri de alto nivel (RAN). KRON constituye una herramienta de representación del conocimiento multi-nivel que consta de una jerarquía con cinco niveles de representación: nivel de implementación, nivel lógico, nivel epistemológico, nivel conceptual y nivel de dominio, de los cuales, sólo los tres últimos se trataran de una forma específica.

El capítulo segundo aborda el nivel de representación epistemológico, en el que se consideran los objetos y las primitivas básicas para la integración de los componentes de una RAN en un lenguaje base de tipo "frame". Los colores, lugares y transiciones están implementados en base a ciertos objetos especiales denominados objetos de marcado, de estado y de acción, respectivamente, interconectados por distintos tipos de relaciones. Las relaciones de red representan los arcos de la RAN y establecen las conexiones causales entre estados y acciones. Dichas relaciones forman parte de una jerarquía de especialización en la que se integra una gran variedad de otras relaciones que servirán para la composición de colores, sincronización de objetos, etc. La construcción de un modelo consiste en la definición prototipos de objetos, cuyas características de comportamiento dinámico quedan definidas por la subred que subyace en el propio prototipo. El modelo completo del sistema se consigue mediante un proceso posterior de instanciación y sincronización de dichos objetos.

En el capítulo tercero se abordan algunos aspectos adicionales como la utilidad de KRON para la representación de conceptos tales como propiedades causales, temporales y de composición de estados. A continuación se apuntan algunas posibilidades de análisis del modelo en base a sus características vistas como RAN y se propone un método para el mecanismo de interpretación de la red. Este mecanismo está dirigido por conflictos y sigue un procedimiento análogo al ciclo de control utilizado en el lenguaje OPS5.

El capítulo cuarto trata de la aplicación de la herramienta de representación KRON en el dominio de aplicación específico que es el objeto del trabajo: los **sistemas de fabricación**. En este nivel de representación se define la terminología y la semántica referida a sistemas de fabricación en términos de los conceptos, objetos, funciones y métodos definidos por los niveles de representación inferiores de KRON (implementación, lógico, epistemológico y conceptual). El sistema de representación incorpora primitivas que definen objetos prototipos de recursos, operaciones, planes de trabajo, órdenes de fabricación, etc. Estos pueden ser instanciados y sincronizados a múltiples niveles de abstracción. Pueden tener propiedades y disponen de una especificación de su comportamiento dinámico definida por una red KRON. Los objetos predefinidos están diseñados formando una jerarquía de especialización en la que también se hereda la dinámica especificada por la red. Esta jerarquía puede ser extendida o modificada según las necesidades de forma interactiva. La metodología de representación comporta una economía en el modelado y resulta especialmente útil para aplicaciones de prototipado rápido. Al mismo tiempo la representación pretende ser independiente de una planta concreta para posibilitar su aplicación a diferentes problemas.

En los capítulos siguientes se aborda la utilización de técnicas de IA en la toma de decisiones para el control de sistemas de fabricación. La estrategia de toma de decisiones que se propone consta de dos niveles: un **distribuidor de operaciones**, que está en contacto directo con un sistema coordinador y da respuesta a problemas de decisión a corto plazo, y un **planificador de operaciones** que trabaja haciendo previsiones de planes de producción a medio y largo plazo.

En el capítulo cinco se hace una revisión de métodos utilizados para planificación de operaciones y a continuación se propone un planificador basado en las ideas de búsqueda dirigida por restricciones y razonamiento oportunista con una visión común sobre planificación predictiva y reactiva. Su construcción está basada en los principios de arquitecturas de pizarra estandard. En esta arquitectura, el control está dirigido por eventos y existen una serie de fuentes de conocimiento utilizadas para planificación y análisis en respuesta a dichos eventos. Para abordar el problema de elegir la fuente de conocimiento más adecuada a cada problema, se propone una metodología en la que, a partir de una caracterización de los conflictos, se generan las heurísticas de meta-conocimiento de control, que guiarán a una fuente de conocimiento de gestión de alto nivel.

La función del *distribuidor de operaciones* es dar respuesta, en "tiempo real", a los problemas de decisión que surgen durante el control de un sistema de fabricación. Los problemas de decisión toman, al nivel del modelo de control, la forma de conflictos. Cada vez que el controlador detecta un conflicto, se plantea su solución aplicando una política de control. En este contexto, el distribuidor de operaciones constituye el marco donde son ejecutadas las políticas de control, que no son más que una llamada parametrizada al mencionado distribuidor. La arquitectura del distribuidor está inspirada en la del planificador de operaciones disponiendo, eso sí, de una gran flexibilidad para modificar su ciclo de control. De esta forma la política de control puede consistir únicamente en una simple regla heurística de despacho o, en el otro extremo, se procede a realizar una elaborada interpretación del plan de operaciones generado por el planificador y provocar, en su caso, una reacción para actualizar dicho plan. La política de control puede construirse modularmente utilizando la biblioteca de fuentes de conocimiento de que dispone el distribuidor. Por otra parte, el sistema de monitorización proporciona una realimentación al resto de funciones del sistema de control, dando soporte de esta forma, al proceso de toma de decisiones. Esta realimentación consiste tanto en actualizar la información de la base de conocimiento, para que refleje los hechos que están sucediendo realmente en la planta, como en detectar posibles contingencias para informar tanto al sistema de coordinación como al sistema de decisión creando y enviando los eventos de control pertinentes.

El objetivo del capítulo septimo es realizar un estudio experimental con dos propósitos, en primer lugar, determinar un conjunto de criterios que proporcionen una base para decidir entre estrategias reactivas alternativas, y en segundo término, desarrollar un conjunto de heurísticas, basadas en estos criterios, para coordinar el proceso reactivo. Para ello se han realizado dos series de experimentos, todos basados en una sección de una fábrica de ensamblado de tarjetas de computador. La primera serie trata de analizar el comportamiento del planificador de operaciones trabajando en modo reactivo. La idea es realizar un análisis comparativo de las estrategias reactivas alternativas, con respecto a actualizaciones específicas del estado, bajo diferentes circunstancias. La segunda serie analiza las decisiones generadas por el distribuidor de operaciones. El método aplicado en este caso ha consistido en incluir indeterminaciones en la información relativas a ciertos parámetros (p.e. tiempos de proceso y condiciones de terminación), a través de un proceso de simulación de eventos discretos se evalan los efectos compuestos de distintas estrategias sobre cierto periodo de tiempo

HISTORIAS DE KRONOPIOS Y FAMAS

TRISTEZA DEL CRONOPIO

A la salida de Luna Park un cronopio advierte
que su reloj atrasa, que su reloj atrasa, que su reloj.
Tristeza del cronopio frente a una multitud de famas que remonta.
Corrientes a las once y veinte y él, objeto verde y húmedo, marca a las once y cuarto.
Meditación del cronopio: "Es tarde, pero menos tarde para mí que para los famas,
para los famas es cinco minutos más tarde,
llegarán a sus casas más tarde,
se acostarán más tarde.
Yo tengo un reloj con menos vida, con menos casa y menos acostarme,
yo soy un cronopio desdichado y húmedo."

Mientras toma café en el Richmond de Florida,
moja el cronopio una tostada en sus lágrimas naturales.

Historias de cronopios y famas
JULIO CORTAZAR

Capítulo 1

Representación del Conocimiento en Sistemas de Eventos Discretos

Este capítulo aborda la representación de sistemas de eventos discretos, centrandose en las aproximaciones desarrolladas según dos perspectivas: técnicas de representación del conocimiento de Inteligencia Artificial y técnicas de modelado basadas en redes de Petri.

En el área de la Inteligencia Artificial se han desarrollado una gran variedad de herramientas para la representación del conocimiento en diversos dominios. Considerando la representación de sistemas de eventos discretos, se realiza un repaso de distintas aproximaciones conocidas y se centra la atención en las representaciones basadas en "frames". Esta representación se ilustra presentando un modelo de un sistema de fabricación.

En cuanto a la perspectiva de representación con redes de Petri, además de resumir el estado del arte, se realiza una presentación de esta familia de herramientas apoyada sobre otro ejemplo tomado del mismo dominio.

Finalmente, tras analizar ambos tipos de técnicas y los beneficios de su utilización conjunta, se presenta la aproximación adoptada en nuestro trabajo. La herramienta de representación a desarrollar, debe facilitar la inclusión del conocimiento necesario para soportar diferentes funciones y, por otra parte, posibilitar el tratamiento de las particularidades inherentes a los sistemas de eventos discretos. La herramienta de representación propuesta, denomina **KRON**, aborda de forma específica la problemática mencionada y está inspirada en la integración de tres aproximaciones complementarias: técnicas de representación basadas en "frames"[MINS 75], programación orientada a objetos [STEF 85] y redes de Petri de alto nivel (RAN) [JENS 86,GENR 86]. La programación orientada a objetos proporciona la base de una metodología de modelado; los "frames" aportan un esquema genérico de representación del conocimiento y las redes de Petri de alto nivel constituyen un formalismo que permite manejar las características dinámicas.

1.1 Introducción

La **representación del conocimiento** es un área de trabajo en Inteligencia Artificial que aborda la identificación, representación y utilización del conocimiento en subsiguientes procesos de razonamiento. Los puntos clave relacionados con el desarrollo de sistemas "inteligentes" incluyen la selección de [BARR 82,DELG 86]:

- las estructuras de información apropiadas para representar el conocimiento y
- los apropiados mecanismos de razonamiento que comportan dos objetivos: dar solución a cuestiones que no son conocidas de manera directa y asimilar nueva información en el sistema de representación.

En los primeros trabajos en el campo de la IA, los métodos de resolución empleaban técnicas con conocimiento pobre[1] que ponían especial énfasis en el método de búsqueda. Sin embargo, pronto se evidenció la importancia del conocimiento para abordar problemas prácticos pasandose a una nueva fase, diametralmente opuesta, en la que sólo se consideró crucial el propio conocimiento, marcando la aparición de lo que se ha venido en llamar *aplicaciones de IA de primera generación* [FOX 89], identificadas por la amplia utilización de lenguajes basados en reglas. La evolución natural de las técnicas de representación produjo una solución simbiótica, a la vez que se trató de solventar las deficiencias de representación de las reglas incorporando representaciones del conocimiento más potentes y estructuradas. A esta *segunda generación* pertenecen los llamados **sistemas basados en el conocimiento**, cuyo nombre pone de manifiesto la idea de que el conocimento del dominio juega un papel fundamental en la resolución del problema. Asumiendo la perspectiva del usuario de la aplicación, Fox indica varios propósitos fundamentales que deben cumplir los sistemas de representación de esta nueva generación:

- Abarcar el conjunto de conceptos requeridos para resolver el problema.
- Representar los conceptos de manera precisa y no ambigüa a todos los niveles de granularidad.
- Proporcionar una representación simple que pueda ser comprendida y utilizada por más de una aplicación.
- Permitir que pueda ser comprendida fácilmente por las personas que construyan las aplicaciones.

Desde el punto de vista de la programación, [WAH 89] propone cuatro criterios para evaluar un esquema de representación del conocimiento: flexibilidad,

[1] "Weak methods" en su denominación inglesa.

"amistosidad" con el usuario, expresividad y eficiencia de procesamiento. Los tres primeros criterios tienen como objetivo simplificar la tarea de programación y comprensión. Por otra parte, la eficiencia o tratabilidad de un esquema de representación es un indicator de la eficiencia o tratabilidad de sus aplicaciones.

Un esquema de representación debe satisfacer los requerimientos enunciados anteriormente tanto desde el punto de vista del usuario como del "software". La representación debe soportar además, las peculiaridades inherentes a su entorno de aplicación que, en el presente caso, pertenece al dominio de los *sistemas dinámicos de eventos discretos* (SDED).

A modo de presentación, se puede decir que los SDED están caracterizados por el hecho de que la evolución del sistema con el tiempo depende de complejas interacciones de las temporizaciones de diversos eventos discretos, tales como la llegada o salida de un trabajo o la inicialización o finalización de una tarea o mensaje. El "estado" de tales sistemas dinámicos cambia únicamente en dichos instantes de tiempo en oposición a los sistemas continuos en los que los cambios se producen de manera continuada [HO 89].

Los sistemas continuos disponen de un marco bien establecido para su modelado, concretado en la utilización de ecuaciones diferenciales. Sin embargo, no existe tal consenso para el caso de los sistemas de eventos discretos. En general, el problema se ha abordado utilizando herramientas que permiten un análisis y manipulación posterior del modelo mediante técnicas matemáticas. De entre estas herramientas se pueden destacar las siguientes:

- Modelos de cadenas de Markov y modelos de autómatas (donde se incluyen Redes de Petri y máquinas de estados finitos).
- Modelos de redes de colas [BUZA 86].
- Modelos algebraicos [INAN 89] y lenguajes formales [RAMA 89].
- Modelos de procesos semi-Markov generalizados (donde se incluyen los trabajos sobre lenguajes de simulación de eventos discretos generales) [GLYN 89].

En §1.2 se revisan algunas aproximaciones utilizadas para la representación de SDED desde la perspectiva de la IA. En §1.3 se revisan algunas aproximaciones desde una perspectiva más formal y específica de SDED, en concreto centrada en RdP. Estas dos perspectivas promueven la idea de una herramienta de representación integrada y sientan las bases de KRON como herramienta de representación.

1.2 Modelado de sistemas de eventos discretos utilizando técnicas de representación de IA

El desarrollo de la IA ha provocado la aparición de gran variedad de esquemas de representación de propósito general. Sin embargo, se dispone de relativamente pocos argumentos científicos que proporcionen una guía para la selección del esquema de representación apropiado para una aplicación dada. Es decir, ningún esquema es claramente superior a los otros para todas las aplicaciones [BARR 82,WAH 89].

Con objeto de mostrar una panorámica de distintas aproximaciones, se describen a continuación brevemente los esquemas generales de representación del conocimiento que han recibido una mayor atención en la literatura: lógica de predicados, sistemas de producción, redes semánticas, frames, representaciones procedurales y representaciones coneccionistas.

1. *Lógica de predicados*: La lógica de predicados estudia las relaciones de implicación entre aserciones y conclusiones. La lógica ofrece generalmente una manera natural de expresar ciertas nociones y existen métodos asentados para determinar el significado de las expresiones en el formalismo lógico [KOWA 74]. Esta aproximación permite utilizar un lenguaje de programación lógica, generalmente PROLOG, para expresar aserciones y las inferencias que pueden ser realizadas.

   A modo de ejemplo, consideremos la utilización de la lógica de predicados, mediante el lenguaje PROLOG, para la representación del conocimiento siguiente: "Toda operación necesita un recurso donde realizarse. Torneado es una operación":

   - inst (torneado , operación) :- .
   - necesita (*obj* , recurso) :- . inst (*obj* , operación) :- .

   De lo que se puede concluir aplicando *modus-ponens*:

   - necesita (torneado , recurso) :- .

2. *Sistemas de producción*: Los sistemas de producción utilizan conjuntos de reglas (con partes de condición y acción, generalmente de la forma "*si entonces*") para resolver problemas [NEWE 72]. Se ha encontrado que los sistemas de producción proporcionan un mecanismo útil para controlar la interacción entre especificaciones de conocimiento declarativo y procedural por lo que se utilizado extensivamente en sistemas expertos e ingeniería del conocimiento. Su desventaja se debe fundamentalmente a su limitada potencia expresiva y su ineficiencia debida al alto nivel de procesamiento, que se necesita para el control del mecanismo de inferencia, cuando la base de reglas alcanza cierto tamaño.

El ejemplo del apartado anterior puede expresarse también mediante un lenguaje basado en reglas, en concreto utilizando el lenguaje OPS5 [BROW 86] resultaría de la siguiente forma:

(*Operacion* ^ **name** *torneado*)

(p *necesidades-operacion*
  (*Operacion*
    ^ **name** < *obj* >)
⟶
  (modify *Operacion*
    ^ **name** < *obj* >
    ^ **necesita** *recurso*))

La ejecución de la regla con la variable < *obj* > ligada a *torneado* produce:

(*Operacion* ^ **name** *torneado* ^ **necesita** *recurso*)

3. *Redes semánticas*: Una red semántica es un grafo orientado cuyos nodos representan objetos, conceptos o situaciones, y cuyos arcos representan relaciones entre nodos [QUIL 66] (el mecanismo de inferencia básico es el de *búsqueda por intersección*). Las redes semánticas por sí solas han proporcionado resultados limitados; sin embargo, se utilizan en general como complemento de otros esquemas de representación fundamentalmente por sus pedagógicas características gráficas.

4. *Frames*: Los "frames" son una de las estructuras para la representación del conocimiento desarrolladas más recientemente y, dada su capacidad de expresión y flexibilidad, son ampliamente utilizadas. La idea de "frame" es una versión elaborada de la idea de red semántica mencionada anteriormente. Los "frames" fueron introducidos por Minsky en 1975 [MINS 75] y consisten básicamente en una composición estructurada de información que incluye información procedural y declarativa en relaciones internas predefinidas dispuesta en forma de atributos[2]. Los atributos pueden contener información muy variada como restricciones, metaconocimiento, heurísticas, métodos, etc. Funcionalmente pueden disponer de varios mecanismos de inferencia: *razonamiento por herencia*, *razonamiento por espectativas* o *razonamiento por defecto*. Existen diversas variedades de "frames" como scripts, units, schema, etc. así como de lenguajes que soportan el esquema de definición anterior: FRL, KRL, UNITS, CRL, etc.

   El ejemplo puede representarse usando CRL (Carnegie Representation Language) disponible en Knowledge Craft [KCRA 86] que utiliza el "schema" como primitiva:

[2]Denominados "slots" en terminología inglesa.

{*operacion*
is-a : object
necesita : recurso }

{*torneado*
instance : operacion }

Si se desea obtener en el schema *torneado* el valor del atributo *necesita*, éste se trata de obtener a través de la jerarquía de especialización definida por las relaciones de herencia *is-a* e *instance* resultando:

{*torneado*
instance : operacion
necesita : recurso }

5. *Representaciones procedurales*: El punto clave en una representación procedural, a diferencia de otro tipo de representaciones declarativas, es que el sistema debe conocer cómo utilizar su conocimiento (cómo encontrar los hechos relevantes, hacer inferencias, etc.) y que la expressión de este comportamamiento se realiza mejor en un lenguaje procedural. El esquema procedural puede representar conocimiento heurístico y realizar inferencias lógicas extensivas, tales como razonamiento plausible. Este esquema puede ser llevado a cabo de forma bastante eficiente puesto que permite procesos de deducción más directos evitando búsquedas superfluas. Su grave inconveniente es que se encuentra muy limitado por los constructores disponibles en el lenguaje (por ejemplo, programas convencionales en Fortran o Pascal no resultan muy adecuados para realizar procesamiento simbólico), así como problemas derivados de la completitud y consistencia de la información representada.

6. *Representaciones coneccionistas*: Los modelos coneccionistas presentan un esquema de representación distribuída: los conceptos se representan sobre multitud de módulos o unidades. Las representaciones distribuidas permiten procedimientos automatizados para el aprendizaje de conceptos y representaciones y soportan ejecuciones paralelas. Recientemente están consiguiendo mayor aceptación sobre todo en entornos de inteligencia artificial distribuída. Su gran inconveniente reside en la dificultad para interpretar el estado del sistema y las representaciones internas requiriendo además, para su aprendizaje, un largo periodo de entrenamiento.

Concluyendo esta revisión de los esquemas de representación se puede decir que ninguno es claramente superior al resto. Esta es la razón por lo que la mayoría de las herramientas comerciales de representación del conocimiento dispongan de varios de los esquemas mostrados anteriormente.

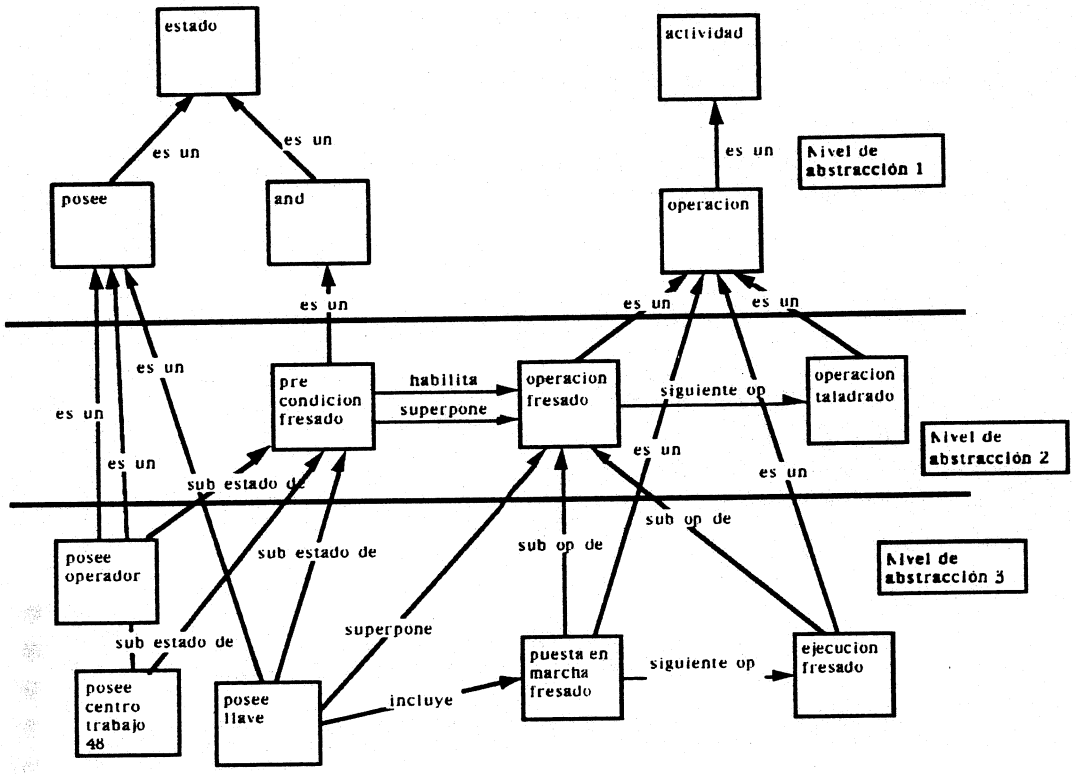

Figura 1.1: Red semántica de actividades del ejemplo.

1.2.1 Ejemplo de representación utilizando frames

Con objeto de completar las consideraciones acerca de aproximaciones más específicas para la representación del conocimiento en sistemas de eventos discretos, se expone a continuación un ejemplo correspondiente al dominio de sistemas de fabricación. La aproximación que se va a presentar fué desarrollada por Mark Fox [FOX 83,FOX 84,FOX 85] para el diseño del sistema ISIS. Siguiendo una terminología usual utilizada en fabricación, el problema a modelar se establece de la siguiente forma:

> La operación de fresado precede a la operación de taladrado. Está descompuesta en dos pasos: puesta en marcha y ejecución. La puesta en marcha tiene lugar en dos horas y el tiempo de ejecución es de 10 minutos. Se requieren dos recursos: una llave de cinco kilos y un operador. La llave únicamente es requerida durante el tiempo de puesta en marcha. La operación se realiza en el centro de trabajo número 48.

Analizando la información detallada en el ejemplo, se reconocen varias entidades relacionadas, que pueden ser representadas en base a frames. En este caso se va

{*operacion-fresado*
is-a : operacion
centro-de-trabajo : centro-trabajo-48
siguiente-operacion : operacion-taladrado
sub-operacion :
puesta-en-marcha-fresado ejecucion-fresado
habilitado-por : precondicion-fresado }

{*puesta-en-marcha-fresado*
is-a : operacion
duracion :
{**instance** : intervalo-tiempo
duracion : 2 horas }
siguiente-operacion : ejecucion-fresado
sub-operacion-de : operacion-fresado
incluido-por : posee-llave }

Figura 1.2: Objetos que representan las operaciones *operacion-fresado* y *puesta-en-marcha-fresado.*

a utilizar la implementación particular de "schemas" (lenguaje CRL) mencionada anteriormente. La figura 1.1 muestra una representación gráfica, en forma de red semántica, de algunos de los objetos utilizados en el modelado de dichas entidades, así como arcos que indican sus relaciones.

En esta aproximación relacional, se divide el conocimiento en dos tipos: actividades y estados. Una descripción de estado representa una instantánea del entorno antes de que se realice una actividad. El estado de la *precondición fresado* debe ser cierto para que pueda realizarse la actividad *operación fresado.* Estados y actividades están conectados vía relaciones causales. Un estado describe qué debe ocurrir en el entorno para habilitar la ocurrencia de una actividad.

Por otra parte, las entidades pueden definirse a varios niveles de abstracción. La operación de fresado se detalla en dos subactividades: puesta en marcha y ejecución de la máquina (la figura 1.2 muestra algunos de sus objetos representativos).

Las relaciones específicas del dominio se definen en términos de otras relaciones más primitivas independientes del dominio formando una jerarquía de especialización; por ejemplo, la *operación fresado* es una *operación* y *operación* es una *actividad.* La semántica de herencia de cada relación puede ser también definida por el usuario, este es el caso de la relación *sub-operacion* que permite agregar información de las suboperaciones [SATH 85]. Unicamente las relaciones *is-a* e *instance* se encuentran predefinidas. Las relaciones pueden incluir también

consideraciones de tipo temporal y causal.

1.3 Aplicación de las Redes de Petri al Modelado de Sistemas de Eventos Discretos

El interés de las redes de Petri como herramienta para el diseño de sistemas dinámicos discretos ha sido extensamente reconocida en la literatura. T. Murata en un reciente "survey" [MURa 89] justifica este interés: "Las redes de Petri son una herramienta de modelado gráfica y matemática aplicable a multitud de sistemas. Son una herramienta prometedora para la descripción y estudio de sistemas de procesamiento de la información caracterizados por ser concurrentes, asíncronos, distribuídos, paralelos, no deterministas, y/o estocásticos. Como herramienta gráfica, las redes de Petri pueden utilizarse como una ayuda para comunicación visual similar a los diagramas de flujo, diagramas de bloques y redes. Adicionalmente, las marcas se utilizan en estas redes para simular las actividades dinámicas y concurrentes de sistemas. Como herramienta matemática, es posible establer ecuaciones de estado, ecuaciones algebraicas, y otros modelos matemáticos que modelan el comportamiento de los sistemas. Las redes de Petri pueden ser utilizadas por teóricos y prácticos. Así, proporcionan un potente medio de comunicación entre ellos: los prácticos pueden aprender de los teóricos cómo construir sus modelos más metódicamente, y los teóricos pueden aprender de los prácticos cómo construir sus modelos de forma más realista".

Los campos de aplicación de las red de Petri han sido muy variados: evaluación de prestaciones, protocolos de comunicación, modelado y análisis de sistemas de software distribuido, sistemas de bases de datos distribuidas, programas paralelos y concurrentes, sistemas de control para manufactura, sistemas de memoria multiprocesador, sistemas de computación de flujo de datos, sistemas tolerantes a fallos, sistemas de automatización de oficinas, logica programable y circuitos VLSI, estructuras y circuitos asíncronos, lenguajes formales, programas lógicos, etc. (para una buena bibliografía a este respecto consultese por ejemplo [SIL 85b,MURa 89]).

Las redes de Petri constituyen una familia de herramientas que pueden ser agrupadas en tipos de modelos:

1. *Modelos autónomos*: constituidos por las redes de Petri ordinarias (con arcos etiquetados con 0/1) y diferentes subclases como los grafos de marcados, las máquinas de estados, las redes de libre elección o las redes simples, o extensiones como las redes generalizadas, las redes con arcos inhibidores, redes de alto nivel, etc.

2. *Modelos interpretados*: están constituidos por un modelo autónomo al que se asocia una interpretación. Así se encuentran intrepretaciones estocásticas,

temporizadas, sincronizadas con eventos y acciones, etc.

Recopilando las características que aporta el uso de las redes de Petri en la representación y tratamiento de sistemas habría que destacar las siguientes [PETE 81,ALJA 87,SILV 89]:

1. Un formalismo gráfico y preciso que facilita el entendimiento acerca del comportamiento esperado del sistema entre los diferentes equipos que participan en el proceso de diseño.
2. Una bien fundada teoría para la verificación cualitativa de las propiedades de la red (vivacidad, ausencia de bloqueos, limitación, etc.) [COLO 87,PETE 81].
3. Un marco de trabajo favorable para el análisis cuantitativo (evaluación de prestaciones) [TPN 85,PNPM 87], actualmente en desarrollo.

Por otra parte, las peculiaridades de la RdP facilitan la construcción del software de aplicación, así como de herramientas de ayuda al diseño:

4. Utilización de lenguajes de descripción basados en RdP [MART 85,MART 86, NARA 85].
5. La independencia con la tecnología de implementación facilita la generación de código para el software de control a partir del modelo de red [MUR 86a, COLO 86].
6. El modelo de red permite una directa simulación del sistema.

1.3.1 Redes de Petri en el modelado

Adicionalmente a las características gráficas y a su semántica simple y bien definida, en relación con la creación de los modelos, las redes de Petri permiten [SILV 89]:

1. La posibilidad de un modelado progresivo utilizando metodologías de diseño por *refinamientos sucesivos* o por *composición modular* (veasé, por ejemplo, [MART 84,ALAN 84,MART 85,ZHOU 88]). Esta característica es necesaria al abordar sistemas con cierta complejidad. El mecanismo de refinamiento permite la construcción de modelos de redes estructurados jerárquicamente, que da lugar a la síntesis de redes con buen comportamiento mediante la agregación incremental de subredes.
2. La integración de interpretación temporal (determinista o estocástica) ha dado lugar a variantes temporizadas y estocásticas de RdP.

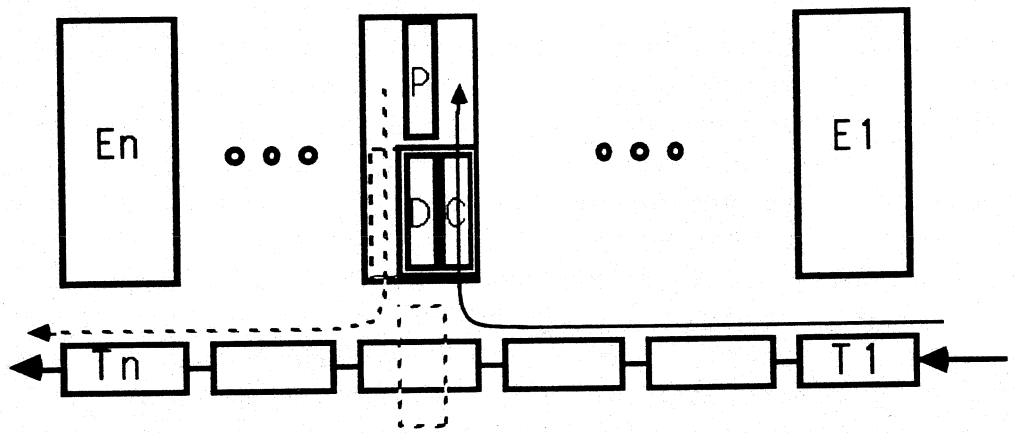

Figura 1.3: Esquema de distribución en planta del taller de fabricación.

3. El modelado de *verdadero paralelismo* (en lugar de su observación secuencial de acuerdo con la idea de "interleaving").

Al tratar con modelos complejos, además de utilizar métodos de composición y refinamiento, resulta más útil el modelado con redes de alto nivel (RAN) como, redes predicado/transición [GENR 86] o redes coloreadas [JENS 86]. Estas redes permiten conseguir modelos más compactos manteniendo, al mismo tiempo, la capacidad descriptiva de las redes de Petri con lo que se consigue un aceptable compromiso entre potencia de modelado, concisión y legibilidad del modelo.

En los apartados siguientes se presenta una introdución de un modelo representativo de red de alto nivel, en concreto las redes de Petri coloreadas, para pasar a continuación a contemplar una aplicación de modelado a través de un ejemplo de sistemas de manufactura.

1.3.2 Ejemplo de representación utilizando RAN

Para proporcionar una visión intuitiva de las posibilidades de modelado con RdP y su funcionamiento, se va a presentar a continuación un ejemplo de modelado utilizando redes de Petri coloreadas. El ejemplo que se propone ha sido ampliamente discutido en la literatura [VALE82,ALLA 82,MART 84,MART 85,MART 87]. Se trata de modelar el comportamiento de un taller flexible en el que se realizan operaciones de sellado de carrocerías de automóviles (el esquema del taller se muestra en la figura 1.3).

Descripción del sistema

El sistema consta de n estaciones de trabajo $(E_1, E_2, \ldots, E_n)$ y de un sistema de transporte unidireccional formado por n tablas con rodillos $(T_1, T_2, \ldots, T_n)$. Cada estación tiene un puesto de trabajo, P, y dos tablas, C y D, que actuan respectivamente como almacenes de carga y descarga de la estación y realizan funciones de carga y descarga del puesto de trabajo P, respectivamente. La tabla C almacena temporalmente cada carrocería que llega a la estación de trabajo hasta que P pasa a encontrarse libre, en cuyo momento la carrocería se carga en P. La tabla D almacena las carrocerías descargadas de P hasta que son evacuadas por el sistema de transporte. El puesto de trabajo P y las tablas de carga y descarga de cada estación tienen capacidad para almacenar una única carrocería . De esta forma, en cualquier momento el número teórico máximo de carrocerías en una estación es tres (una en C, otra en D y una tercera en el puesto de trabajo P).

Cada tabla T_i del sistema de transporte puede contener un máximo de una carrocería . Las tareas básicas que puede llevar a cabo el sistema de transporte son:

1. cargar la tabla T_1 desde el exterior,
2. transferir una carrocería desde la tabla T_i a la tabla T_{i+1}, para $1 \leq i < n$,
3. cargar la estación E_i con una carrocería dispuesta sobre la tabla T_i, para $1 \leq i \leq n$,
4. descargar una carrocería de la estación E_i (situada sobre la tabla D correspondiente) y cargarla en la tabla T_i, para $1 \leq i \leq n$ y
5. evacuar una carrocería dispuesta sobre la tabla T_n hacia el exterior.

Las carrocerías entran en el sistema de fabricación a través de la tabla T_1 y salen vía la tabla T_n, después de visitar una de las estaciones. En principio, no se hace ninguna hipótesis restrictiva acerca de la naturaleza de las carrocerías o sobre la forma de asignar la estación que, cada una de ellas, debe visitar.

En lo sucesivo, y para dar generalidad a la presentación, se hablará de piezas en lugar de concretar al caso de carrocerías.

Modelado con redes de Petri coloreadas

Para proceder a la construcción del modelo puede seguirse alguna metodología de modelado de sistemas con RdP, como por ejemplo la propuesta en [MART 84]. El método de trabajo consta de tres etapas:

1. Definición del *conjunto de colores* implicados en el modelo.

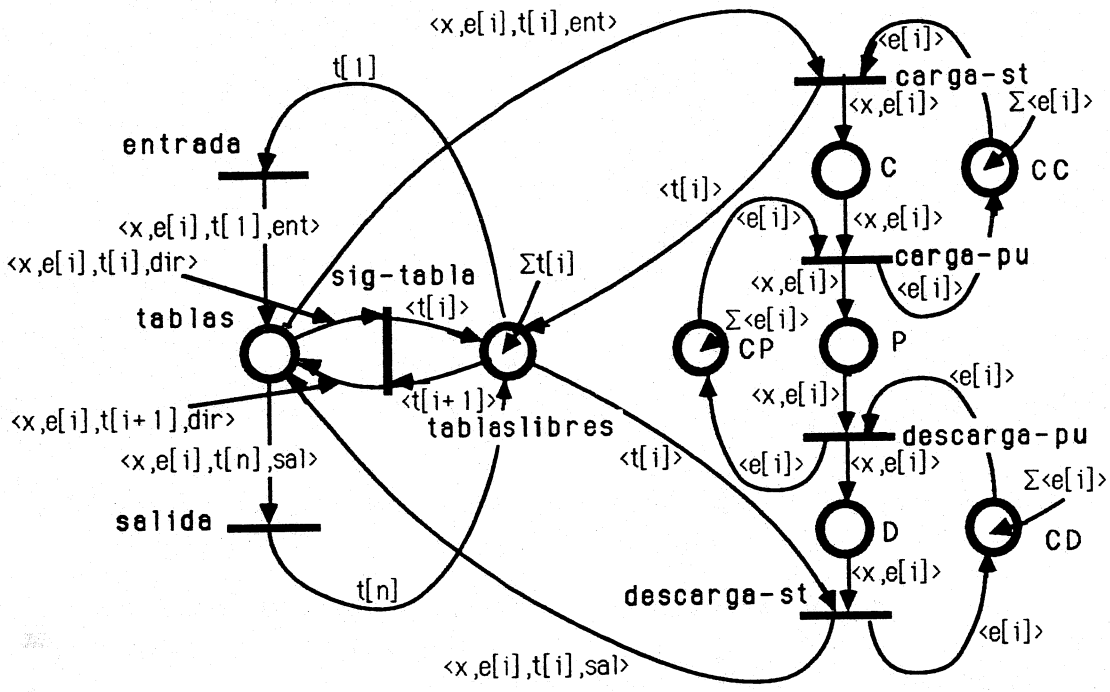

Figura 1.4: RdPC que modela el taller de fabricación del ejemplo.

2. *Construcción de módulos*: se reconocen los posibles subsistemas en que puede descomponerse el sistema y se realiza su modelado separado con redes de Petri.

3. *Conexionado de módulos*: se realiza fusionando aquellas transiciones que representen sincronizaciones entre subsistemas.

En el sistema a modelar se identifican inmediatamente dos subsistemas: estaciones y transporte. Detalles sobre su modelado separado con RdP y posterior sincronización pueden encontrarse en [MART 84]. La figura 1.4 muestra la RdPC producto del modelado propuesta en [MART 87].

El comportamiento dinámico de las *estaciones* se ha modelado según una secuencia de tres lugares, C, P y D, junto con las restricciones de capacidad de los almacenes intermedios, CC y CD y de la estación de trabajo CP. Las estaciones se distinguen por distintos colores (*color* = *dato*) involucrados en el modelo formando el conjunto $E = \{e[i], 1 \leq i \leq n\}$.

El comportamiento del sistema de transporte está modelado por una subred que representa una cola FIFO con una interacción especial con la subred de estaciones. Las distintas tablas son también distinguibles por medio del conjunto de colores $T = \{t[i], 1 \leq i \leq n\}$. Esta subred está formada por los lugares *tablas* y *tablaslibres* y las transiciones *entrada* y *salida*.

El orden de las tablas y estaciones es de gran importancia en la construcción del modelo y vendrá reflejado por el índice de cada color que impone, por tanto, una relación de orden.

Para caracterizar el estado de una pieza dentro del sistema de transporte, son necesarios dos datos: la estación a la que debe visitar y su posición actual. Esta información compuesta se consigue agregando dos colores simples: estación $s[i]$ y tabla $t[i]$, así se considera el conjunto de colores:

$$C = S \times T = \{(s[i], t[i]) \mid 1 \leq i \leq n \ \wedge \ 1 \leq j \leq n\}$$

Aunque el comportamiento del coordinador del sistema es independiente del tipo de piezas que llegan, esta información es tenida en cuenta en el modelo a través del conjunto de colores $P = \{x, y, z, \ldots\}$.

Para representar el sentido de una pieza (entrando para su procesamiento o saliendo una vez procesada) se utiliza el conjunto de colores $ES = \{ent, sal\}$.

Las **transiciones** que aparecen en el modelo corresponden a tareas elementales llevadas a cabo en el taller. Estas transiciones se describen a continuación, junto con los subconjuntos de colores asociados con su disparo:

- **entrada** $\{ (pieza, e[i], t[1], ent), pieza \in P, \ 1 \leq i \leq n \}$: carga de la tabla T_1 desde el exterior, el color $e[i]$ corresponde a la estación que debe visitar la pieza;

- **salida** $\{ (pieza, e[i], t[n], sal), pieza \in P, \ 1 \leq i \leq n \}$: descarga de la tabla T_n hacia el exterior;

- **carga-st** $\{ (pieza, e[i], t[i], ent), pieza \in P, \ 1 \leq i \leq n \}$: carga una de las estaciones (la estación $e[i]$) desde el sistema de transporte;

- **carga-pu** $\{ (pieza, e[i]), pieza \in P, \ 1 \leq i \leq n \}$: carga un puesto de trabajo desde la tabla C correspondiente;

- **descarga-pu** $\{ (pieza, e[i]), pieza \in P, \ 1 \leq i \leq n \}$: descarga un puesto de trabajo sobre la tabla D correspondiente;

- **descarga-st** $\{ (pieza, e[i], t[i], sal), pieza \in P, \ 1 \leq i \leq n \}$: descarga una de las estaciones (estación $e[i]$) en el sistema de transporte;

- **sig-tabla** $\{ (pieza, e[i], t[j], dir), pieza \in P, \ (1 \leq i \leq n) \wedge (1 \leq j \leq n-1) \wedge ((dir = ent) \wedge (j < i)) \vee ((dir = sal) \wedge (j \geq i)) \}$: transfiere una pieza de la tabla T_j a la tabla T_{j+1}.

Los arcos estan etiquetados por funciones con los siguientes significados:

- $tabla(x, e[i], t[j], dir) = (t[j])$; proyecta el dato tabla (tercera componente del registro).
- $estacion(x, e[i], t[j], dir) = (e[i])$; proyecta el dato estación (segunda componente del registro).
- $sigpos(x, e[i], t[j], dir) = (x, e[i], t[j+1], dir)$; actualiza la posición de la pieza en el sistema de transporte incrementando en una unidad el índice de la tabla.

Estas funciones son especializaciones de otras funciones más generales [MART 84]:

función proyección $proy_i(c_1, \ldots, c_i, \ldots, c_n) = (c[i])$

función siguiente $sig(c[i]) = (c[i+1])$

Los **lugares** que aparecen en el modelo permiten identificar visiones parciales del estado del sistema. Los lugares y sus conjuntos de colores asociados son los siguientes:

- **tablas** $\{ (pieza, e[i], t[j], dir), pieza \in P,\ 1 \leq i \leq n,\ 1 \leq j \leq n, dir \in ES \}$: identifica las tablas que se encuentran ocupadas.
- **tablaslibres** $\{ (t[j]),\ 1 \leq j \leq n \}$: identifica las tablas que se encuentran libres.
- **C** $\{ (pieza, e[i]), pieza \in P,\ 1 \leq i \leq n \}$: identifica las estaciones cuya tabla de carga está ocupada.
- **CC** $\{ (e[i]), 1 \leq i \leq n \}$: identifica las estaciones cuya tabla de carga está libre.
- **P** $\{ (pieza, e[i]), pieza \in P,\ 1 \leq i \leq n \}$: identifica las estaciones cuya tabla de proceso está ocupada.
- **CP** $\{ (e[i]), 1 \leq i \leq n \}$: identifica las estaciones cuya tabla de carga está proceso.
- **D** $\{ (pieza, e[i]), pieza \in P,\ 1 \leq i \leq n \}$: identifica las estaciones cuya tabla de descarga está ocupada.
- **CD** $\{ (e[i]), 1 \leq i \leq n \}$: identifica las estaciones cuya tabla de descarga está libre.

El sistema evoluciona mediante el disparo de sus transiciones. Para poder proceder al disparo de una transición, es necesario que esta se encuentre sensibilizada con respecto a algún color. Considerese, por ejemplo, la transición *carga-pu*, para

que esta transición se encuentre sensibilizada respecto a algún color, es necesario que en los lugares de entrada a la transición (es decir, C y CP), existan al menos tantos colores como los indicados al aplicar sus funciones de arco respectivas al mencionado color. Así, para que se encuentre habilitada respecto al color $< x, e[3] >$, es necesario que un color $< e[3] >$ se encuentre presente en el lugar CP y que $< x, e[3] >$ se encuentre presente en el lugar C. El disparo de la transición se lleva a cabo en dos pasos:

1. Se **quitan** de los lugares de **entrada** a la transición los colores indicados al aplicar la función de arco correspondiente al color de disparo. Para el caso propuesto, se habría de quitar el color $e[3]$ de FP y el color $< x, e[3] >$ de C.
2. Se **añaden** a los lugares de **salida** de la transición los colores indicados al aplicar la función de arco correspondiente al color de disparo. Para el caso propuesto, se habría de añadir el color $< x, e[3] >$ al lugar P y $e[3]$ al lugar CC.

1.3.3 Definición de las redes de Petri coloreadas

Introduzcamos primero algunas nociones relativas a multi-conjuntos [JENS 86] (la teoría de multi-conjuntos es una extensión de la teoría de conjuntos).

Un *multi-conjunto* sobre un conjunto no vacío S, es una función $b \in [S \rightarrow \mathcal{N}]$, donde $\mathcal{N}$ es el conjunto de todos los naturales, incluyendo el cero. La diferencia con un conjunto es que un multi-conjunto puede contener múltiples ocurrencias de un mismo elemento.

El conjunto de todos los **multi-conjuntos finitos** sobre un conjunto no vacío S se denotará como S_{MS}.

En lo que sigue, únicamente trataremos con multi-conjuntos finitos, donde cada multi-conjunto s sobre S puede ser representado por la suma formal: $b = \sum b(s)s$, donde el entero no negativo $b(s)$ denota el número de ocurrencias del elemento s en el multi-conjunto b.

Una función $F \in [S \rightarrow R_{MS}]$, donde S y R son conjuntos no vacíos, y S_{MS} y R_{MS} son los multi-conjuntos finitos de S y R respectivamente, puede ser extendida, de manera única, a una **función lineal** $F \in [S_{MS} \rightarrow R_{MS}]$ llamada **extensión multi-conjunto** de F:

$$\forall b \in S_{MS} : F(b) = \sum b(s) \times F(s)$$

Se utilizará $[\ldots]_L$ para denotar el conjunto de todas las funciones lineales.

Definición [JENS 86]: Una red de Petri coloreada es una 6-tupla dada por:

$$< P, T, C, Pre, Post, M_0 >$$

donde:

1. P es un conjunto de **lugares.**
2. T es un conjunto de **transiciones.**
3. $P \cup T \neq \emptyset$ *y* $P \cap T = \emptyset$
4. $$C \in [P \cup T \rightarrow \{D_1, \ldots, D_k\}]$$
   La **función color**, C, se define desde $P \cup T$ en los conjuntos no vacíos de colores posibles $\{D_1, \ldots, D_k\}$. Asocia a cada lugar un conjunto de posibles **colores de marcado** y a cada transición un conjunto de posibles **colores de ocurrencia.**
5. Pre y $Post$ son las **funciones de incidencia previa** y **posterior** definidas sobre $P \times T$, tales que $Pre(p,t), Post(p,t) \in [C(t)_{MS} \rightarrow C(p)_{MS}]_L$ para todo $(p,t) \in P \times T$.
6. $\forall p \in P \;\; \exists t \in T : Pre(p,t) \neq 0 \vee Post(p,t) \neq 0$ y
   $\forall t \in T \;\; \exists p \in P : Pre(p,t) \neq 0 \vee Post(p,t) \neq 0.$
7. El **marcado inicial** M_0 es una función definida sobre P, tal que $[M_0(p) \in C(p)_{MS}]$ para todo $p \in P$.

Definición: El **marcado** de una RdPC es una función M definida sobre P tal que

$$M(p) \in C(p)_{MS}, \forall p \in P$$

$M(p)$ y $M_0(p)$ representan el marcado del lugar P para el marcado actual y el inicial respectivamente. Ambos valores son multiconjuntos, eventualmente nulos.

Al estar definidas como extensiones lineales, las funciones de una RdPC pueden expresarse mediante matrices.

Definición: La **matriz de incidencia** W de una RdPC está definia por:

$$\forall (p,t) \in P \times T \;, \quad W(p,t) = Post(p,t) - Pre(p,t) \qquad \forall (p,t) \in P \times T$$

Considerando la extensión lineal, la función de incidencia puede representarse como una matriz de funciones lineales.

Definición: Una transición $t \in T$ está **sensibilizada** para un marcado m,

$\Leftrightarrow \exists c \in C(t)$ tal que $Pre(p,t)(c) \leq M(p), \forall p \in P$

El disparo de una transición t sensibilizada consiste en quitar $Pre(p,t)(c)$ marcas y añadir $Post(p,t)(c)$ marcas a todos los lugares $p \in P$.

1.4 KRON: Integración de RdP y técnicas de IA

Como se ha mostrado en los apartados anteriores, existen diversas herramientas de modelado en el entorno de IA que pueden ser utilizadas para la representación de sistemas dinámicos discretos. Destacan, por su mayor aplicación, las que hacen uso de representaciones estructuradas del conocimiento, y en particular, herramientas de representación basadas en frames. Si bien las herramientas tipo frame son adecuadas en multitud de casos debido a su flexibilidad, potencia de representación e inferencia y su posibilidad de manipulación simbólica, adolecen de algunas carencias importantes al ser aplicadas al modelado de sistemas concurrentes relativamente complejos. Entre las razones que apoyan esta afirmación se pueden destacar las siguientes:

- Carecen de un formalismo que permita efectuar una validación, comprobar propiedades dinámicas y analizar la consistencia del modelo.
- Carecen de una metodología específica que aborde la problemática de sincronización de actividades entre objetos con comportamiento dinámico, lo que favorece la aparición de inconsistencias en la especificación del flujo de información entre dichas entidades.

Por otra parte, las RdP, y más específicamente las RdPC, resuelven algunas de las carencias presentadas por las herramientas basadas en frames pero a su vez por sí solas presentan otras limitaciones referentes a problemas genéricos de representación:

- Permiten únicamente expresar una perspectiva restringida del sistema completo. Existen multitud de conocimientos en un sistema, además de su comportamiento dinámico, que, si bien no son relevantes al especificar el flujo de información durante la evolución del sistema, pueden ser fundamentales al considerar otro tipo de deducciones. Tal es el caso en un sistema de manufactura, de la ubicación de las máquinas, su potencia, los gastos asociados, tasa de fallos, etc., conocimientos que pueden ser claves, por ejemplo, en aplicaciones como planificación de operaciones.
- No existe una relación inmediata entre los elementos que componen una RdPC y los objetos físicos o conceptuales que modela. Por otra parte, los colores no son más que tuplas de datos, perdiendose la información relativa a las relaciones reales que existen entre sus elementos.

1.4.1 Conexión de perspectivas RdP/IA

El conjunto de ventajas e inconvenientes de estas dos diferentes perspectivas de representación de sistemas dinámicos (una basada en RdP y la otra en técnicas

de IA) muestra que ambas perspectivas pueden complementarse, y surge de forma natural la idea de integración de ambas en una única herramienta. Esta integración está favorecida por el hecho de que existen multitud de analogías entre conceptos del dominio de RdP e ideas desarrolladas en IA relativas a representación, búsqueda, sistemas basados en reglas etc. (véase, por ejemplo, [VALE 87,MURO 89,FLEI 89]).

La idea de utilizar conjuntamente redes de Petri y técnicas de IA ha sido ya apuntada en varios trabajos. Los siguientes apartados revisan dichos trabajos agrupados temáticamente.

Implementaciones

Una primera serie de trabajos se han dedicado a una mera implementación de RdPs en herramientas clásicas de IA. En [BRU 86a] se destaca la similitud entre RdPs y sistemas basados en reglas y traduce una extensión de RdPs (redes PROT [BRU 86c]), empleando programas escritos en OPS5 [BROW 86]. Las marcas y lugares corresponden a elementos de memoria de trabajo de OPS5, adicionalmente, cada marca lleva consigo una etiqueta que indica el instante en que fué modificada. Las transiciones son implementadas como reglas en las que la parte condición comprueba las marcas de los lugares de entrada a la transición y puede incluir también su predicado (que se hace por medio del reconocimiento de los nombres de los lugares sobre los elementos de memoria de trabajo), la parte acción quita y pone marcas en los lugares de entrada y salida respectivamente. La decisión de la transición a disparar está impuesta por las técnicas de disparo de OPS5. Los motivos que justifican la elección de OPS5 son la facilidad de traducción que proporciona y la disponibilidad de un interfase amistosa para el usuario.

En [CAST 85] se utiliza el lenguaje PROLOG para la implementación de RdPC, su objetivo es la relización de búsquedas exhaustivas para explorar las evoluciones de la red con el tiempo. En un trabajo posterior [GENT 87] se utilizan técnicas de sistemas expertos para traducir las descripciones de usuario en una serie de RdPCs que adquieren la forma de reglas de producción.

Integraciones

En un segundo grupo se incluyen los trabajos que corresponden a integraciones más profundas, en el sentido de hacer un uso más intenso de las características de representación de IA o, según el otro sentido, de utilizar la semántica de la RdP como parte del proceso de control inferencial.

A [SIBE 85] se deben los primeros intentos en la integración de RdP con representaciones estructuradas del conocimiento, proponiendo sus *redes de Petri de alto nivel con estructuras de datos.* En estas redes, las marcas son sustituidas por

estructuras de datos llamadas *entidades*, que son instancias de *clases de entidad*[3], que a su vez son conjuntos de propiedades con valores por defecto, haciendo una aplicación del concepto de "frame" dado por Minsky [MINS 75]. El comportamiento es acorde con las reglas que gobiernan las RdPAN: una transición está sensibilizada si en sus lugares de entrada contiene suficientes entidades para que las variables de la función de incidencia puedan tomar las diferentes entidades, etc. El disparo de una transición retira entidades de lugares de entrada, altera sus valores de acuerdo a la acción, y pone finalmente las entidades en los lugares de salida.

En [VALE 87,ATAB 87] se representan las *redes de Petri de decisión* en base al mismo concepto de "frames". Consideran que las RdP corresponden con un tipo particular de aproximación basada en "frames" con tres clases de objetos específicos: lugares (utilizados para estructurar el conocimiento del estado), transiciones (corresponden con reglas de deducción o procedimientos de actualización del estado) y marcas (objetos asociados a un cierto estado). Proponen el establecimieto de relaciones específicas (es-una-causa-de, puede-alterar, en-el-estado-de) para definir las conexiones de la red. A nivel de aplicación, utilizan la estructura de la red como un árbol de reglas con el que ejecutar encadenamiento inferencial, hacia delante en modo ejecución y hacia atrás con objetivos de diagnóstico. El sistema ha sido realizado en el lenguaje LeLisp.

[PETE 86] aporta una aproximación a las RdP desde el punto de vista de la programación lógica. Propone un método para traducir programas lógicos expresados en lógica' de primer orden a una especificación en RdPC. De esta forma consigue convertir procesos involucrados en programación lógica tales como ligadura, unificación, sustitución y resolución en un problema de resolución de un sistema homogéneo de ecuaciones para un tipo particular de soluciones llamadas T-invariantes. Otros trabajos sobre problemas análogos pueden encontrarse en [LAUT 85,MUR 86b].

En el área de la programación basada en reglas, Fleischhack y Weber han propuesto [FLEI 89] un modelo de representación basado en redes predicado/transición denominado *red de sistema de producción* (SP)[4]. Las redes SP agregan el formalismo de los sistemas de producción y RdP quedando definidos de manera más precisa el concepto de ejecución de una regla y su influencia en la base de datos, por otra parte aprovecha algunas de las características mejor desarrolladas de cada perspectiva para el beneficio del sistema de representación combinado.

[RIBA 88] propone un esquema de representación del conocimiento basado en RdP binarias y aporta un algoritmo para hacer inferencias por el procedimiento de búsqueda por intersección; sin embargo la capacidad expresiva del esquema de representación es relativamente limitada, ya que las inferencias desarrolladas no son expresadas en el propio lenguaje de representación.

---

[3] "Entity Class" en su versión inglesa.

[4] "Production System net" en su nominación inglesa, o simplemente "PS net".

[LOON 88] asigna valores borrosos[5] a las marcas y transiciones y propone un algoritmo que permite hacer inferencias sobre la red y utilizarla para el soporte de decisiones.

Metodologías

Desde el punto de vista de la metodología de programación, la perspectiva según la programación orientada a objetos [STEF 85] ha sido abordada también en el entorno de la IA[6]. Cabría destacar aquí el trabajo de [BALD 87] que presenta un entorno de programación conceptual orientada a objetos basado en redes PROT que además de facilitar un prototipado rápido proporciona modelos ejecutables.

Otra aproximación orientada a objeto a resaltar es la propuesta en [HOLL 87].que desarrolla un entorno completo para la especificación y simulación de sistemás dinámicos con primitivas predefinidas para el dominio de fabricación.

1.4.2 KRON: Redes Orientadas a la Representación del Conocimiento

Los modelos reseñados anteriormente se han limitado a integraciones parciales que no resuelven, a nuestro parecer, de manera satisfactoria el problema de representación esbozado al comienzo del presente capítulo. Para abordar esta tarea, se plantea la necesidad de disponer de una potente metodología de modelado, así como, de un formalismo adecuado para representar, manipular y analizar la dinámica de estos sistemas. Además, dicho formalismo debe convivir con una técnica general para representar el resto del conocimiento útil para razonar acerca del modelo.

Con este objetivo se propone a continuación una herramienta para la representación del conocimiento basada en "frames" para sistemas discretos con comportamiento concurrente o paralelo que ha sido denominada **KRON** (**K**nowledge **R**epresentation **O**riented **N**ets).

KRON aprovecha la potente estructura de datos aportada por los "*frames*" para la expresión general del conocimiento, a la cual añade el formalismo de las *RAN* para la descripción del comportamiento dinámico de los sistemas. Adicionalmente, sigue el paradigma de la *programación orientada a objeto*, lo que proporciona la base para su metodología de modelado.

Un objetivo importante en la definición de las primitivas y conceptos de KRON es poder representar conceptos dependientes del dominio de aplicación en términos

[5] "fuzzy" en su terminología inglesa.

[6] De hecho, la mayor parte de los entornos de desarrollo importantes de IA hacen uso de esta metodología

{*taladradora*
is-a : maquina
localizacion : celda-trabajo-21
volumen : medio
velocidad : 3500 rpm
operaciones-realizadas : op-taladro1 op-taladro5 }

Figura 1.5: Representación externa de un "schema".

de otros independientes de dicho dominio.

En [BRAC 79] se propone una definición de varios niveles de representación que fue posteriormente desarrollada en [FOX 83,SATH 85]. Aquí se presentan estos niveles, ligeramente adaptados a las peculiaridades de la herramienta contemplada:

1. *Nivel de implementación.* Recoge las primitivas para la interpretación a nivel de máquina. En otras palabras, su objetivo es definir las estructuras de datos de más bajo nivel.

   Para evitar cualquier ambiguedad en la definición de los objetos, la especificación de las estructuras de representación será descrita en términos del lenguaje de representación del conocimiento, **CRL** (Carnegie Representation Languaje). CRL utiliza como primitiva elemental el objeto "*schema*" que es una de las implementaciones propuestas para los "frames". Sintacticamente, un "schema" (figura 1.5) está compuesto de un nombre de "schema" (*taladradora*) y un conjunto de atributos (tales como *is-a*[7], *localizacion, volumen, velocidad* y *operaciones-realizadas*), y se representará siempre encerrado entre llaves con el nombre del "schema" apareciendo en la parte superior.

2. *Nivel lógico.* Proporciona la interpretación lógica de la información almacenada. Concretamente el triplete schema-atributo-valor se interpreta como una aserción incluida en el "schema". Así por ejemplo, el valor *celda-trabajo-21* en el atributo *localizacion* se interpreta como la aserción: la *taladradora* esta en la *localizacion celda-trabajo-21*, que puede representarse en lógica de predicados por la fórmula atómica: *localizacion* (*taladradora, celda-trabajo-21*).

3. *Nivel epistemológico* describe los mecanismos para regular el flujo de información a través de la herencia. En este nivel se definen los conceptos

[7]Nombrado por su denominación inglesa al ser un atributo definido por el lenguaje CRL. Por coherencia con la sintaxis del lenguaje se mantendrá también su denominación original para todas las palabras clave en la misma situación.

de prototipo, instancia, niveles de agregación, objetos especiales, y relaciones estructurales que conectan estos conceptos. En la figura 1.5, la especialización está impuesta por la relación principal de herencia: *is-a*[8]. Estas nociones de herencia van a permitir establecer las jerarquías de especialización típicas de los lenguajes orientados a objetos [STEF 85]. Por estas similitudes, en adelante se hablará indistintamente de objetos para referirnos a los "schema"[9].

En el caso de KRON, se definen también a este nivel los objetos y relaciones especiales que marcan el flujo de información en los procesos dinámicos de los objetos. Los conceptos de acción y estado se van a definir a este nivel en su significado más elemental.

4. *Nivel conceptual.* Aporta conceptos comunes a los modelos en diferentes dominios, tales como nociones de tiempo, causalidad, posesion, etc.

5. *Nivel de dominio.* Proporciona los conceptos, palabras y expresiones específicos de un dominio de aplicación. Por ejemplo, máquinas, almacenes y operaciones en el dominio de sistemas de manufactura.

La aportación en este trabajo se circunscribe en la problemática de representación del conocimiento, y más específicamente en los aspectos relativos a la **integración** de *RdPAN* en entornos de representación de *IA*.

1.5 Conclusiones

En este capítulo se han estudiado alternativas al modelado de sistemas de eventos discretos. Se han explorado técnicas de representación utilizadas en IA y aproximaciones basadas en redes de Petri. Una crítica de ambas ha puesto de relieve ciertas limitaciones de cada una y algunos aspectos complementarios de ambas.

La creación de una herramienta de representación del conocimiento específica para sistemas de eventos discretos, exige la consideración de dos perspectivas. Por una parte, debe reunir características de representación desarrolladas en el entorno de la Inteligencia Artificial, y por otra, debe facilitar la manipulación de los problemas específicos de sistemas de eventos discretos, especialmente los relativos a su carácter dinámico.

KRON trata de integrar estas dos aproximaciones en una misma herramienta, aprovechando la potente estructura de datos aportada por los "*frames*" para la

[8]Por razones de economia, en la representación escrita de un objeto no se suelen escribir todos sus atributos (de especialización o heredados), únicamente se describiran aquellos que se consideren relevantes para la explicación.

[9]Es de resaltar que la otra característica importante de los lenguajes orientados a objeto relativa a la noción de encapsulación se utiliza usualmente de forma relajada en las aplicaciones de IA.

expresión general del conocimiento, a la cual añade el formalismo de las *redes de alto nivel* para la descripción del comportamiento dinámico de los sistemas. Adicionalmente, sigue el paradigma de la *programación orientada a objetos*, lo que proporciona una base para su metodología de modelado.

El objetivo fundamental de esta herramienta es facilitar la construcción de modelos de sistemas que puedan observarse indistintamente desde ambas perspectivas. Esto permitirá elegir en todo momento la perspectiva que proporcione las mayores posibilidades para resolver cada problema específico. Esta integración está facilitada de antemano por ciertas similitudes existentes entre ambas aproximaciones. Estos factores, juntamente con la metodología de modelado orientada a objetos, permiten satisfacer fundamentales requerimientos de **generalidad**, **accesibilidad** y **extensibilidad**.

CONSERVACION DE LOS RECUERDOS

Los famas para conservar sus recuerdos proceden a embalsamarlos en la siguiente forma: Luego de fijado el recuerdo con pelos y señales, lo envuelven de pies a cabeza en una sábana negra y lo colocan parado contra la pared de la sala, con un cartelito que dice: "Excursión a Quilmes", o: "Frank Sinatra."

Los cronopios, en cambio, esos seres desordenados y tibios, dejan los recuerdos sueltos por la casa, entre alegres gritos, y ellos andan por el medio y cuando pasa corriendo uno, lo acarician con suavidad y le dicen: "No vayas a lastimarte." Es por eso que las casas de los famas son ordenadas y silenciosas, mientras que en las de los cronopios hay gran bulla y puertas que golpean. Los vecinos se quejan siempre de los cronopios, y los famas mueven la cabeza comprensivamente y van a ver si las etiquetas están todas en su sitio.

Historias de cronopios y famas
JULIO CORTAZAR

Capítulo 2

Nivel de Representación Epistemológico

KRON está construido utilizando un lenguaje de representación basado en "frames"[1]. KRON incrementa dicho lenguaje con primitivas sobre métodos, relaciones, funciones y objetos que permiten integrar las características de las RAN (redes de Petri de alto nivel).

KRON es una herramienta de representación del conocimiento multi-nivel, especializada en la construcción de modelos para sistemas de eventos discretos. El nivel de representación epistemológico considera los objetos y primitivas básicas para la integración de los componentes de una RAN en un lenguaje base de tipo "frame". Los colores, lugares y transiciones están implementados en base a ciertos objetos especiales denominados objetos de marcado, de estado y de acción respectivamente, interconectados por distintos tipos de relaciones. Las más importantes son las relaciones de red que representan los arcos de la RAN y permiten establecer, además, las conexiones causales entre estados y acciones. Dichas relaciones forman parte de una jerarquía de especialización en la que se integra una gran variedad de otras relaciones que servirán para la composición de colores, sincronización de objetos, etc. La construcción de un modelo consiste en la definición prototipos de objetos, cuyas características de comportamiento dinámico quedan definidas por la subred que subyace en el propio prototipo. El modelo completo del sistema se consigue mediante un proceso posterior de instanciación y sincronización de dichos objetos.

La herramienta de modelado tiene una interpretación activa y su semántica soporta la simulación de eventos discretos concurrentes.

[1]Se han desarrollado de forma parcial implementaciones en base a dos lenguajes comerciales: CRL [KCRA 86] y LOOPS [LOOPS 83].

2.1 Introducción

El nivel de representación epistemológico explora la representación de las unidades conceptuales y sus interrelaciones. Los sistemas dinámicos trabajan con entidades dinámicas. El sistema de representación del conocimiento debe representar la información relevante relativa a dichas entidades. Esta información puede ser agrupada según tres categorías:

1. *Características estáticas.* Proporciona la descripción de un objeto que es invariante con el tiempo. Esto incluye sus propiedades físicas estáticas (como el tipo de operaciones permitidas o la media del tiempo de proceso de una operación en una máquina herramienta), y sus restricciones[2] de trabajo (como restricciones sobre el tamaño de pieza permitido o relativas a la calidad del trabajo necesario sobre la pieza).

2. *Descripción del comportamiento dinámico.* Proporciona la descripción de las especificaciones dinámicas, como posibles estados del objeto (p.e. disponibilidad o nivel de utilización de un recurso de transporte) y una descripción de las acciones a ejecutar sobre el objeto que conducen a un cambio de estado (p.e. comienzo y final del procesado en una máquina). Esta categoría de descripción incluye también la conexión o comunicación con otros objetos.

3. *Información sobre el comportamiento previsto y recolección de información histórica.* En aplicaciones de planificación, es deseable que el sistema de representación ofrezca la posibilidad de especificar cuándo un estado puede presentarse de acuerdo al plan de operaciones. Inversamente, la representación debe registrar también una recolección histórica de datos del objeto con motivo de comprobar en que medida se han cumplido los objetivos, determinar tiempos estandar, etc.

De todos los contextos de información mencionados previamente, KRON se focaliza en el punto 2. Su cometido es integrar las primitivas básicas de las RAN, como arcos, lugares, transiciones o colores, en un entorno basado en frames, para obtener una aproximación relacional y estructural (en base al establecimiento de relaciones entre objetos y características estructurales de las redes KRON que los componen) a la descripción de sistemas de eventos discretos.

2.2 Ejemplo de presentación

Las cuestiones concernientes a la semántica de representación se van a tratar examinando un problema que pertenece al dominio de planificación y control de

[2] "Constraints" según su terminología inglesa.

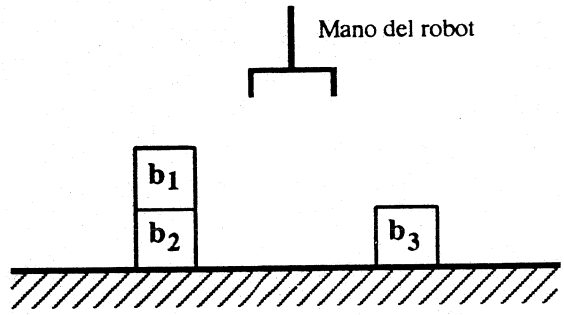

Figura 2.1: Escena del mundo de bloques con robot

alto nivel de robots. El ejemplo consiste en un mundo de bloques sobre el que puede actuar un robot. La figura 2.1 muestra una típica escena, que se enuncia aquí en la línea apuntada en [NILS 82]. En la escena hay varios bloques etiquetados (b_1, b_2, b_3), que pueden ser situados sobre una mesa o sobre otros bloques, y un robot que posee una mano móvil con la que puede coger y mover bloques.

Este ejemplo pertenece al dominio de sistemas dinámicos discretos, y su comportamiento puede ser modelado sencillamente usando redes de Petri de alto nivel (RAN). La figura 2.2 muestra un modelo construido con redes de Petri coloreadas (RdPC), donde un conjunto de colores simple, $B = \{b_1, b_2, b_3\}$, se utiliza para representar los bloques, y otro conjunto de colores compuestos, OB para indicar las parejas de bloques relacionados ($< b_i, b_j >$ marcando el lugar *sobre* indica, en este caso, que el bloque b_i está sobre el bloque b_j).

En este modelo, queda completamente definido el flujo de información del sistema, que condiciona su comportamiento dinámico. KRON tiene la capacidad de integrar esta red en el modelo de representación, asociándole toda su carga semántica. Pasamos a analizar el modelo de este sistema en KRON. Así, se puede decir que el marcado de los lugares *libre, sobre-mesa, cogido* y *sobre*, representa el estado del *mundo-de-bloques*. El significado de esta representación del estado es el siguiente:

- Una marca $< b_i >$ en el lugar *libre* representa un bloque que no tiene ningún otro encima, por lo tanto, podrá ser cogido por el robot. Adoptando una terminología de lógica podría expresarse mediante el predicado *libre*(b_i).

- Una marca $< b_i >$ en el lugar *sobre-mesa* representa un bloque apoyado sobre la mesa, no tiene ningún otro bloque debajo (y en terminología de lógica: *sobre-mesa*(b_i)).

- Una marca $< b_i >$ en el lugar *cogido* representa un bloque que ha sido agarrado por la mano del robot (y en terminología de lógica: *cogido*(b_i)).

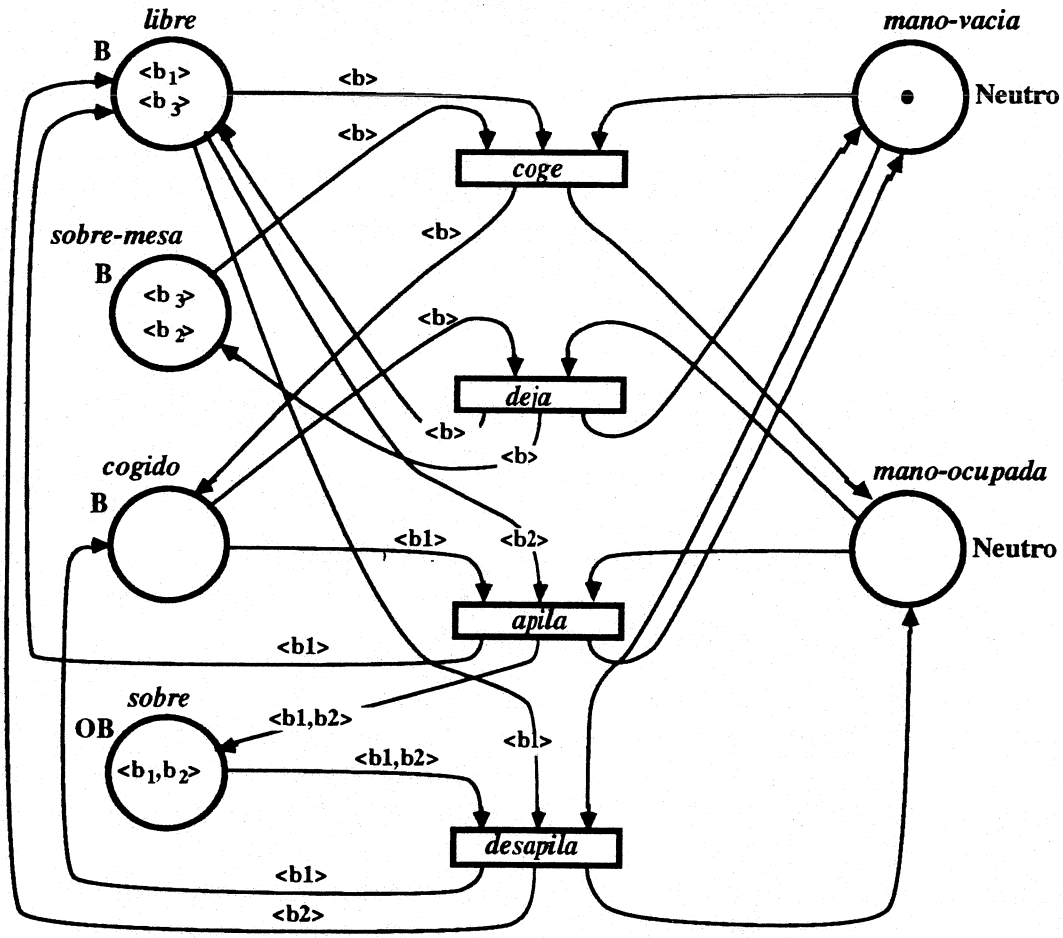

Figura 2.2: Red de Petri coloreada que modela el mundo de bloques con robot. Los conjuntos de colores utilizados son: $B = \{< b_i > /i = 1..3\}$ y $OB = \{< b_i, b_j > /i, j = 1..3 \text{ y } i \neq j \}$.

{*sobre*
is-a : atribucion
domain : (type is-a bloque)
range : (type is-a bloque)
value-restriction : (not (equal (get domain) (get range)))
inverse : debajo }

{*bloque*
is-a : objeto-marcado
sobre :
debajo :
volumen :
peso :
color : }

Figura 2.3: Objeto de marcado representativo del *bloque* y la relación *sobre*.

- Una marca $< b_i, b_j >$ en el lugar *sobre* indica que representa un bloque b_i sobre otro bloque b_j (y en terminología de lógica: $sobre(b_i, b_j)$).

Las transiciones pueden verse como acciones cuya ejecución comporta el cambio de estado del modelo y por consiguiente en el conjunto de predicados. Así, la acción *coge* requiere la verificación, para algún $b \in \{< b_i > /i = 1..3\}$, de los predicados: *mano-vacia, libre(b)* y *sobre-mesa(b)*. Su ejecución implica que estos predicados dejan de verificarse y pasan a ser ciertos: *mano-ocupada* y *cogido(b)*.

2.3 Objetos de Marcado

Las marcas o colores son objetos de información que se almacenan en los lugares de una RAN, evolucionan a través de la red y determinan su estado. En KRON, las marcas están representadas por un tipo de objeto especial denominado **objeto de marcado**.

Los bloques del mundo de bloques del ejemplo, pueden ser, por tanto, representados como objetos de marcado. Todos los bloques del ejemplo pertenecerán a una misma clase de objetos cuya superclase es *objeto-marcado*. Se les distingue por sus nombres diferentes y pueden contener información adicional relacionada con los atributos y relaciones de un bloque. La figura 2.3 muestra la definición del bloque prototipo. El modelo KRON del ejemplo consta de tres diferentes objetos de marcado (b_1, b_2 y b_3), con posibles características diferentes, pero con una estructura

{b_1
instance : bloque
sobre : b_2 }
{b_2
instance : bloque
debajo : b_1 }
{b_3
instance : bloque }
{*bloque*
is-a : objeto-marcado
instance+inv : b_1 b_2 b_3
sobre :
debajo :
volumen :
peso :
color : }

Figura 2.4: Objetos de marcado correspondientes a los bloques.

idéntica, en la figura 2.4 se representan las instancias para cada uno de los bloques.

Los colores compuestos se construyen añadiendo relaciones especiales a estos objetos de marcado. Estas relaciones conectan bidireccionalmente dos objetos y pueden pertenecer a alguna de las dos clases siguientes:

1. *Atribución*: Cuando pueden cambiar los objetos conectados por la relación (el bloque b_1 está *sobre* el bloque b_2).

2. *Especialización*: La relación es permanente (el bloque b_1 tiene *color* azul).

El dominio y rango de la relación son especificados en su definición e imponen restricciones en las clases de objetos que pueden estar conectados por la relación. En el ejemplo, se puede definir la clase *bloque* y la relación *sobre* de la forma mostrada en la figura 2.3.

En términos de RAN, las relaciones definidas permiten especificar los posibles conjuntos de colores que componen el objeto de marcado. A partir de las definiciones de la clase *bloque* y la relación *sobre*, se pueden encontrar los siguientes conjuntos de colores simbólicos:

$B = \{< b_i >|\ b_i$ instance *bloque*$\}$

$OB = \{< b_i, b_j >|\ b_i, b_j$ instance *bloque* y $b_i \neq b_j\ \}$

A nivel de implementación, la detección de los conjuntos de colores simples es extraordinariamente sencilla, puesto que se encuentran directamente en forma de lista en el atributo *instance+inv* de la clase de objeto impuesta en la restricción (figura 2.4).

Los objetos de marcado que representan la situación inicial en el ejemplo propuesto, pueden ser expresados de la forma definida en la figura 2.4.

2.3.1 Elementos de la memoria de trabajo

La interpretación dinámica de un modelo KRON se basa en la evolución de los objetos de marcado y está soportada por lo que más adelante se definirá como *mecanismo de control de red* (apartado 3.3). Dentro de dicho mecanismo, los objetos de marcado van a constituir los **elementos de la memoria de trabajo**, siguiendo la terminología utilizada en OPS5 (consultese la referencia [BROW 86, pag. 42]). Es decir, los elementos de marcado son los elementos relevantes para el algoritmo de control.

Como elemento de memoria de trabajo, sólo algunas características del objeto de marcado van a ser relevantes: su nombre (atributo implícito que se denota como "schema-name"), el atributo de relación dentro de la jerarquía de especialización ("is-a" e "instance" y sus valores) y los pares nombre de atributo y valor que corresponden a relaciones de atribución o especificación (el resto de atributos no resultan ser relevantes para dicho mecanismo de control). Estos atributos serán referidos como **atributos de memoria de trabajo**.

2.3.2 Formación de ordenaciones en los objetos de marcado

En la aproximación adoptada en KRON, se sigue una metodología de representación dirigida por datos, es decir, se deja gran parte de la carga representativa al propio objeto. Así, es posible establecer ordenaciones entre objetos de marcado a través de sus relaciones de especialización. En KRON se adopta un solución relacional, para lo que se establecen relaciones con gran contenido físico.

Así, la relación *sig-tabla* (figura 2.5), que une cada tabla con su siguiente no es más que otra característica en la definición de cada objeto tabla. La relación inversa *ant-tabla* impone la relación de orden en el sentido opuesto. La relación *con-estacion* permite conectar cada tabla con su estación (véase el apéndice B para obtener una visión completa de la implementación).

Como conclusión se puede decir que, se utiliza un método de programación más dirigido a los datos, y permite integrar parte de su significado físico en la definición

{*sig-tabla*
is-a : especificacion
domain : (TYPE is-a tabla)
range : (TYPE is-a tabla)
inverse : ant-tabla
description : "siguiente tabla" }

{*tabla*
is-a : objeto-marcado
inverse+inv : T_1 T_2 T_3 T_4 T_5 T_6
sig-tabla :
con-estacion :
pieza : }

{T_1
instance : tabla
sig-tabla : T_2
con-estacion : S_1 }
{T_2
instance : tabla
sig-tabla : T_3
ant-tabla : T_1
con-estacion : S_2 }
...
{T_6
instance : tabla
ant-tabla : T_5
con-estacion : S_6 }

Figura 2.5: Definión de relaciones para imponer relaciones de orden en las tablas de transporte.

{*mundo-bloques*
is-a : objeto-estado
estados : libre sobre-mesa cogido sobre
type: attribute
libre :
type: atributo-estado
sobre-mesa :
type: atributo-estado
cogido :
type: atributo-estado
sobre :
type: atributo-estado
sector-mundo-bloques :
type: part-of }

Figura 2.6: Objeto de estado que representa el prototipo del mundo de bloques.

del objeto, lo que resulta un punto más a favor en herramientas de representación del conocimiento.

2.4 Objetos de Estado y Atributos de Estado

En [FOX 83,FOX 85] se proponen tres tipos especiales de atributos para distinguir formalmente la clase de información contenida en la descripción de una entidad: "*attribute*" (define un atributo de un objeto que es cierto para el objeto como un todo, p.e. *masa* o *color*), "*has-part*" (especifica que el valor de un atributo es un componente de la definición del objeto, p.e. "la pierna-derecha *es-parte-de* un hombre") y "*structure*" (especifica una estructuración de los atributos que son del tipo anterior, p.e. "la pierna-derecha está *a-la-derecha-de* la pierna-izquierda"). KRON, además de los tipos de atributos anteriores, utiliza también otro tipo especial de atributo para describir la información del estado relacionada con la entidad que está siendo definida.

La información del estado en una RdP está representada por su marcado (distribución de las marcas contenidas en los lugares). Cada lugar está representado en KRON por un tipo de atributo especial llamado **atributo de estado** que se denotará en los objetos como *atributo-estado*. Todos los lugares que proporcionan descripciones del estado de una misma entidad se encuentran en el objeto que representa la entidad (siguiendo la noción de *asociación*, idea clave en la construcción de representaciones estructuradas del conocimiento), este tipo de objeto se denomina **objeto de estado**.

{b_1
sobre : b_2 }

{*mundo-bloques1*
instance : mundo-bloques
libre : b_1 b_3
cogido :
sobre-mesa : b_2 b_3
sobre : b_1 }

Figura 2.7: Descripción del estado del mundo de bloques del ejemplo.

En el ejemplo propuesto en el presente capítulo, pueden observarse claramente dos entidades diferenciadas: bloques y robot. Todos los bloques pueden pasar por los mismos estados y poseen características muy similares, por lo que cabe pensar en una entidad abstracta (que denotaremos como *mundo-de-bloques*) que recoge dichas características comunes. El prototipo de la entidad abstracta que representa el mundo de bloques puede representarse por un objeto de estado (*mundo-bloques*), conteniendo cuatro atributos de estado que representan los cuatro lugares implicados: *libre, sobre-mesa, cogido* y *sobre* figura 2.6). La figura 2.7 modela la instancia concreta del mundo de bloques del ejemplo.

Cada lugar de una RAN tiene asociado un conjunto de colores que limita los posibles colores con que puede ser marcado. Análogamente, cada atributo de estado de una red KRON tiene asociada una restricción sobre los posibles valores que puede albergar, que está dispuesta a modo de metaconocimiento asociado al atributo. Normalmente, es un predicado simple que impone restricciones sobre los prototipos de objetos de marcado permitidos.

Siguiendo un mecanismo de deducción análogo al utilizado para el mundo de bloques, es posible modelar el objeto de estado representativo del robot (figura 2.8).

2.4.1 Memoria de trabajo local

Contempla otra vez el mecanismo de control de la red. Sólo algunos de los objetos de marcado (elementos de la memoria de trabajo) van a encontrarse en cada atributo de estado. El conjunto de los objetos de marcado contenido en cada atributo de estado se denominará **memoria de trabajo local** y cada objeto constituirá un **elementos de la memoria de trabajo local.**

{*robot-prototipo*
is-a : recurso-transporte
mano-vacia :
mano-ocupada : }

{*robot*
instance : robot-prototipo
mano-vacia : neutro }

Figura 2.8: Objeto de estado que representa al prototipo robot y una instancia.

2.5 Objetos de Acción

El disparo de una transición en una RAN provoca una transformación del estado o situación inicial a otro estado o situación final. Por tanto, las transiciones son la unidad básica de acción en una RdP. Las transiciones se representan en KRON mediante otro tipo de objeto especial llamado **objeto de acción**. Un objeto de acción porta toda la información relativa a la transición de la RAN que representa (lugares de entrada y salida, funciones de los arcos, etc.). En el objeto puede disponerse además, cualquier otro tipo de información relacionada con dicha acción (temporización, probabilidad de activación, etc.). Desde la perspectiva de la IA y dentro del marco de los sistemas de producción [WINS 77,WATE 78], una transición en una red de Petri puede ser considerada también como una regla precondición/postconsición [ZISM 78,BRU 86a,VALE 87,FLEI 89]: la parte precondición de la regla da las condiciones de sensibilización mientras que la parte postcondición da las modificaciones en el marcado introducidas por el disparo de la transición.

En el desarrollo de KRON se tratará de mantener el significado según ambas perspectivas. Esto posibilita hacer uso, en todo momento, de la perspectiva que proporcione mayores ventajas. Así, por ejemplo, en la etapa de modelado y validación del comportamiento dinámico del sistema, se hace un uso más extensivo de las características del modelo como RAN. Como ejemplo interesante según la perspectiva de la IA, para la implementación del mecanismo de control de la red, se han aprovechado las características desarrolladas para el *lenguaje OPS5* [BROW 86] (como será reseñado en el apartado 3.3).

A diferencia de lo sucedido con la integración de los lugares, representados por atributos dentro de objetos, las transiciones van a ser representadas por objetos completos, las razones fundamentales que han llevado a esta decisión son dos:

{*objeto-accion*
is-a : objeto-KRON accion
pre-relaciones-red :
post-relaciones-red :
dato-asociado :
predicado :
asociado-a-conflicto : }

Figura 2.9: Prototipo de objeto de acción.

1. La similitud con reglas en sistemas de producción u operadores en espacios de búsqueda, aconseja la representación y manipulación diferenciada de cada transición. La representación elegida: aserciones para lugares y reglas para transiciones, profundiza en los argumentos de esta decición, justificada por la simplicidad de representación de las primeras en comparación con la carga semántica de las segundas.

2. El segundo argumento es de tipo metodológico, las transiciones van a ser la base del mecanismo de sincronización entre objetos. Ello obliga a adoptar una representación relativamente independiente, con relación al objeto de estado, que representa el objeto (una transición resultante de una sincronización de dos transiciones será acción de dos o más objetos de estado diferentes y, evidentemente, no puedrá formar parte excusiva de ninguno de ellos de forma particular).

La figura 2.9 muestra la forma general de un objeto de acción, de la cual se derivará el resto. Estableciendo la similitud con las RdP, en los atributos *pre-relaciones-red* y *post-relaciones-red* se situan las especificaciones de los arcos de entrada y salida de la transición. En el atributo *dato-asociado* se indican, indirectamente, las clases de objetos de marcado relevantes al objeto de acción, que junto con el atributo *predicado*, impondrán el conjunto de colores asociados a la transición que representa. En los siguientes apartados se contemplan con más detalle todas estas características.

Para terminar la presentación general de los objetos de acción, pasemos a contemplar las implicaciones de la introducción de los objetos de acción en el modelo del ejemplo considerado. La figura 2.10 muestra las subredes KRON que modelan el mundo de bloques y el robot considerados actuando independientemente. Como puede deducirse de la figura, en el mundo de bloques es posible la ejecución de cuatro acciones, modeladas en base a objetos de acción y representadas en la figura como rectángulos:

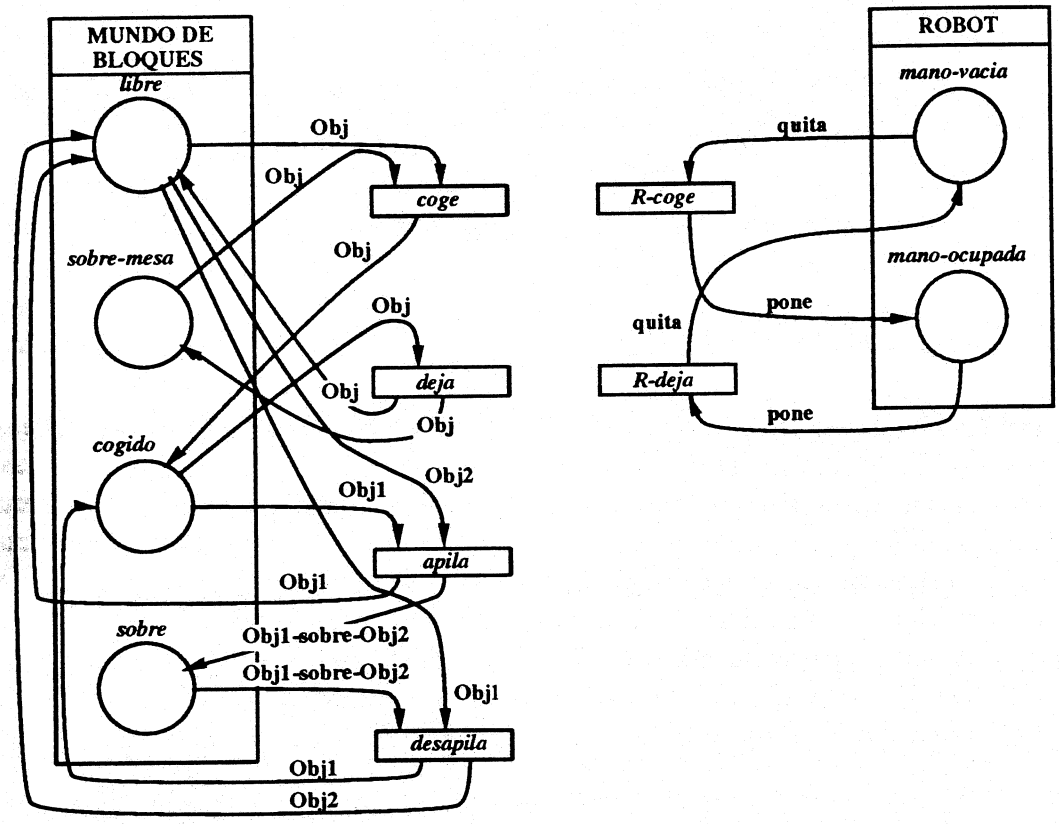

Figura 2.10: Subredes KRON que modelan las entidades del *mundo-bloques* y el *robot*.

- *coge* : representa la acción de coger algún bloque que se encuentra libre y sobre la mesa para pasar a estar cogido.
- *deja* : representa la acción de liberar sobre la mesa algún bloque que se encontraba previamente cogido.
- *apila* : representa la acción de apilar sobre un bloque libre otro bloque que se encontraba previamente cogido, .
- *desapila* : representa la acción de coger un bloque libre (en la cima) de alguna pila de bloques.

Por otra parte, sobre el robot se pueden distinguir dos acciones: coger un bloque con la mano o dejarlo. Estas acciones están representadas respectivamente por los objetos de acción *R-coge* y *R-deja* de la figura 2.10.

Los arcos de las redes constituyen el objeto específico de los siguientes apartados.

2.5.1 Relaciones de Red

La información relacionada con los arcos de la RAN está soportada en los objetos de acción por la definición de dos relaciones especiales. Estas relaciones conectan objetos de acción con atributos de estado a modo de *relaciones causales* y atributos de estado con objetos de acción jugando un papel de *condiciones de habilitación.* Se denominan respectivamente *post-relaciones-red* y *pre-relaciones-red.* La figura 2.11 muestra sus objetos de definición. Estas relaciones se asemejan a las relaciones "*is-a-cause-of*" y "*can-alter*" propuestas en [VALE 87]. En lenguajes basados en sistemas de producción se suelen llamar *precondiciones, parte condición* o *lado izquierdo*[3] y *postcondiciones, parte acción* o *lado derecho*[4] [BARR 82].

Los valores de las relaciones de red se representan en el objeto de acción como tripletes de datos indicando respectivamente el objeto de estado, el atributo de estado y su función de red (la figura 2.12 muestra el objeto representativo de la acción *desapila*). En las relaciones de red previas, cada triplete constituye una **condición de arco**, los valores contenidos en el atributo de estado (del objeto de estado) forman la **memoria de trabajo local**, la función de red del elemento condición especifica un patrón que ha de ser encajado[5] en la memoria de trabajo, cuando todos los elementos condición son satisfechos simultaneamente se dice que el objeto de acción está **sensibilizado**. El **disparo** del objeto de acción implica quitar los objetos de marcado, relacionados con la sensibilización, de sus atributos de estado y ejecutar el proceso inverso en las **acciones** que corresponden con los

[3] "Preconditions", "condition part", y "left-hand side" según sus denominaciones inglesas.
[4] "Postconditions", "action part", y "right-hand side" según sus denominaciones inglesas.
[5] "Matched" en su acepción inglesa.

{*post-relacion-red*
is-a : relacion-red
domain : (SET (TYPE is-a objeto-accion))
range : (slot (SET (TYPE is-a atributo-estado)))
inverse : post-relacion-red-con }

{*pre-relacion-red*
is-a : relacion-red
domain : (slot (SET (TYPE is-a atributo-estado)))
range : (SET (TYPE is-a objeto-accion))
inverse : pre-relacion-red-con }

Figura 2.11: Objetos que definen las relaciones de red.

tripletes de las relaciones de red posteriores. Este proceso de reconocimiento y disparo es el objeto de los siguientes apartados.

2.5.2 Condición de arco

Las relaciones de red conectan un atributos de estado con objetos de acción. Cada una de estas relaciones tiene asociada una **condición de arco** consistente en una especificación de restricciones sobre la *memoria de trabajo local* relativa al atributo de estado.

Las condiciones de arco están constituidas por una lista integrada por pares de elementos:

1. La memoria de trabajo local únicamente puede contener elementos de memoria de trabajo de un determinado tipo (impuesto por la restricción de tipo del atributo de estado). El primer elemento de cada par podrá ser cualquier atributo de la memoria de trabajo (véase el apartado 2.3) perteneciente a dicho tipo.

2. El segundo puede ser un nombre de variable u otra condición de arco:

   - Una variable está indicada por un identificador encerrado entre los símbolos "<" y ">", por ejemplo, $< v1 >$ y $< v2 >$, en
     {schema-name $< v1 >$ sobre $< v2 >$}
     Esta condición de arco reconocerá todos los elementos de la memoria de trabajo local que tengan un *nombre* y algún valor en el atributo *sobre*. Por cada reconocimiento posible, se crea una ligadura entre las variables

{*desapila*
is-a : objeto-accion
pre-relaciones-red :
(mundo-bloques1 sobre Obj1-sobre-Obj2)
(mundo-bloques1 libre Obj1)
post-relaciones-red :
(mundo-bloques1 cogido Obj1)
(mundo-bloques1 libre Obj2)
dato-asociado : $< vobj1, vobj2 >$
predicado : t }

donde:

pone $\equiv$	{**schema-name** $< vobj >$}
quita $\equiv$	{**schema-name** $< vobj >$}
Obj1 $\equiv$	{**schema-name** $< vobj1 >$}
Obj2 $\equiv$	{**schema-name** $< vobj2 >$}
Obj1-sobre-Obj2 $\equiv$	{**schema-name** $< vobj1 >$ **sobre** $< vobj2 >$}

Figura 2.12: Objeto de acción *desapila*.

$< v1 >$ y $< v2 >$ y el valor de los atributos *nombre* y *sobre* del elemento reconocido. Se dirá entonces, que las variables $< v1 >$ y $< vsobre >$ están **ligadas** a los valores que reconocen.

- Si el segudo elemento es otra condición de arco, los valores almacenados en el atributo funcionan a modo de memoria de trabajo local y el proceso de reconocimiento se ejecuta sobre sus elementos, por ejemplo:

  {schema-name $< v2 >$
  sobre {schema-name $< vo2 >$ sobre $< v3 >$ } }

  Esta condición reconoce todos los elementos de la memoria de trabajo local que tengan un *nombre* y algún objeto en el atributo *sobre* que tenga a su vez otro objeto en el atributo *sobre*.

Se dice que la condición **reconoce** un objeto de marcado de la memoria de trabajo local si los valores de los atributos del objeto de marcado satisfacen todas las restricciones impuestas por la condición de arco.

El ámbito de una variable es un objeto de acción, es decir, todos los usos de un identificador de variable en un mismo objeto de acción se refieren a la misma variable, pero no existe ninguna vinculación entre variables pertenecientes a diferentes objetos de acción.

Función	Definición	Observaciones
id	id(<c>)=<c>	identidad
proy	proy(<a1,...,an>, i,j,...,m)=<a_i , a_j,..., a_m>	proyección
sig	sig(<a(i)>)=<a(i+1)>	siguiente
prec	prec(<a(i)>)=<a(i-1)>	precedente
rod	rod(<a(i)>)=<a(i+1)> rod(<a(n)>)=<a(1)>	rotación derecha (<a(n)> : último color)
roiz	roiz(<a(i)>)=<a(i-1)> roiz(<a(1)>)=<a(n)>	rotación izquierda
sigt	sigt(<a(1),...,a(n)>, i,...,m) = = <a(1),...,sig(<a(i)>),...,sig(<a(m)>),...,a(n)>	siguiente n-tupla
prect	prect(<a(1),...,a(n)>, i,...,m) = = <a(1),...,prec(<a(i)>),...,prec(<a(m)>),...,a(n)>	precedente n-tupla
rodt	rodt(<a(1),...,rod(<a(i)>),...,rod(<a(m)>),...,a(n)>	rotación derecha de n-tupla
roizt	roizt(<a(1),...,a(n)>, i,...,m)= =<a(1),...,roiz(<a(i)>),...,roizt(<a(m)>),...,a(n)>	rotación izquerda de n-tupla

Figura 2.13: Biblioteca de funciones de RAN.

2.5.3 Condiciones de arco en KRON vs. funciones de arco en RAN

Como se adelantó al presentar los objetos de marcado en el apartado 2.3, KRON utiliza una aproximación relativamente orientada a los datos, es decir, gran parte de la carga de conocimiento se encuentra asociada al propio dato. Este hecho hace que la carga expresiva necesaria en las condiciones de arco sea inferior a sus equivalentes, funciones de arco, en el marco de las RANs. Estos factores van a repercutir en dos hechos importantes:

1. se reduce enormemente la biblioteca de posibilidades de las condiciones de arco en comparación con el repertorio necesario al trabajar con aproximaciones tradicionales de RANs y

2. permite implementar estas condiciones de manera más homogénea.

Para justificar estas afirmaciones, considérese la biblioteca de funciones de arco, expresada en la terminología usual de RdPC, que se muestra en la figura 2.13. En los párrafos siguientes se presentarán algunas de las funciones de dicha biblioteca y se mostrará su representación utilizando KRON.

La figura 2.14 muestra la definición de algunos objetos de marcado que se utilizarán como ejemplo. En primer lugar, hay que resaltar que las relaciones de orden subyacentes en los colores genéricos de las funciones utilizadas en las RANs (p.e. el conjunto de colores $\{a[i], 1 \leq i \leq n\}$ define una relación de orden a través del índice: $a[1] < a[2] < \ldots < a[n]$), se definen en KRON a través del establecimiento de relaciones de especificación.

{*A*
instance : color1
anterior :
siguiente : B
ant-rotacion : F
sig-rotacion : C }

{*B*
instance : color1
anterior : A
siguiente : C
ant-rotacion : E
sig-rotacion : D }

{*C*
instance : color1
anterior : B
siguiente : D
ant-rotacion : A
sig-rotacion : E }

{*D*
instance : color1
anterior : C
siguiente : E
ant-rotacion : B
sig-rotacion : F }

{*E*
instance : color1
anterior : D
siguiente : F
ant-rotacion : C
sig-rotacion : B }

{*F*
instance : color1
anterior : E
siguiente :
ant-rotacion : D
sig-rotacion : A }

{*Tupla*
instance : tupla-colores
prop1 : A
prop2 : F
...
propn : D }

Figura 2.14: Ejemplo de composición de objetos de marcado.

- La relación *siguiente*[6] permite establecer conexiones entre objetos de tipo *color1*:

  {*siguiente*
  is-a : especificacion-orden
  domain : (type is-a color1)
  range : (type is-a color1)
  inverse : anterior }

  Siguiendo los valores en la relación *siguiente* en las instancias de *color1* de la figura 2.14, se encuentra la ordenación A, B, C, D, E y F respecto al criterio impuesto por dicha relación.

- Como se deduce de la definición anterior, la relación *siguiente* dispone de su relación inversa, denominada *anterior*, que indica las conexiones en sentido inverso que da la secuencia: F, E, D, C, B y A.

- Se pueden definir otro tipo de relaciones de orden respecto a otros criterios sobre los mismos objetos de marcado, simplemente definiendo nuevas relaciones. Por ejemplo, la relación *sig-rotacion* se utiliza en los objetos del ejemplo para definir un orden en sentido rotacional dado por la secuencia base: $A, C, E, B, D, F, A, \ldots$. Identicamente *ant-rotacion* define el sentido en rotación inversa ($A, F, D, B, E, C, A, \ldots$.

Las funciones con orden sin rotación pueden aplicarse a objetos con la relación *siguiente*, p.e. objetos del tipo *color1*:

- **siguiente** $\equiv$ {siguiente $< vobj >$ }
  p.e. para el objeto A, {siguiente $< vobj >$ }
  liga a la variable $< vobj >$ el objeto B: $< vobj > \longrightarrow B$

- **precedente** $\equiv$ {anterior $< vobj >$ }
  p.e. para el objeto F, {anterior $< vobj >$ }
  se realiza la ligadura : $< vobj > \longrightarrow E$

Las funciones con orden con rotación pueden aplicarse a objetos con la relación *sig-rotacion*, p.e. objetos del tipo *color1*:

- **rotacion derecha** $\equiv$ {sig-rotacion $< vobj >$ }
  p.e. para el objeto A, {sig-rotacion $< vobj >$ }
  se realiza la ligadura : $< vobj > \longrightarrow C$

[6]Los nombres dados a las relaciones se han elegido, en este caso, de forma que tengan cierto significado para el objetivo con el que se plantea el ejemplo, en general se les identificará por un nombre que las relacione con su significado físico.

- **rotacion izquierda** $\equiv$ {ant-rotacion $< vobj >$ }
  p.e. para el objeto A, {ant-rotacion $< vobj >$ }
  se realiza la ligadura : $< vobj > \longrightarrow F$

Las funciones sobre tuplas pueden aplicarse a *objetos de marcado compuestos.* Estos se construyen creando relaciones a otros objetos de marcado que posean a su vez relaciones de atribución o especificación. Como ejemplo se puede ver en la figura 2.14 el objeto *Tupla* que dispone de varias relaciones (*prop1, prop2, ..., propn*) hacia objetos del tipo *color1* que contienen las relaciones *siguiente, anterior, sig-rotacion* y *ant-rotacion*, como ya se ha detallado anteriormente.

- **siguiente n-tupla** respecto a los atributos $propi, propj, \ldots, propm \equiv$
  {propi {siguiente $< vobji >$ }
  propj {siguiente $< vobjj >$ }
  ...
  propm {siguiente $< vobjm >$ } }
  p.e. para el objeto *Tupla*, la condición **siguiente n-tupla** respecto a los atributos $prop1, prop2, \ldots, propn$, resulta en las ligaduras:
  $< vobj1 > \longrightarrow B, < vobj2 > \longrightarrow nil, \ldots, < vobjn > \longrightarrow E,$

- **precedente n-tupla** respecto a los atributos $propi, propj, \ldots, propm \equiv$
  {propi {anterior $< vobji >$ }
  propj {anterior $< vobjj >$ }
  ...
  propm {anterior $< vobjm >$ } }
  p.e. para el objeto *Tupla*, la condición **precedente n-tupla** respecto a los atributos $prop1, prop2, \ldots, propn$, resulta en las ligaduras:
  $< vobj1 > \longrightarrow nil, < vobj2 > \longrightarrow E, \ldots, < vobjn > \longrightarrow C,$

- **rotacion derecha n-tupla** respecto a los atributos $propi, propj, \ldots, propm \equiv$
  {propi {ant-rotacion $< vobji >$ }
  propj {ant-rotacion $< vobjj >$ }
  ...
  propm {ant-rotacion $< vobjm >$ } }
  p.e. para el objeto *Tupla*, la condición **rotacion derecha n-tupla** respecto a los atributos $prop1, prop2, \ldots, propn$, resulta en las ligaduras:
  $< vobj1 > \longrightarrow C, < vobj2 > \longrightarrow A, \ldots, < vobjn > \longrightarrow F,$

- **rotacion izquierda n-tupla** respecto a los atributos $propi, propj, \ldots, propm \equiv$
  {propi {sig-rotacion $< vobji >$ }
  propj {sig-rotacion $< vobjj >$ }
  ...
  propm {sig-rotacion $< vobjm >$ } }
  p.e. para el objeto *Tupla*, la condición **rotacion izquierda n-tupla** respecto a los atributo $prop1, prop2, \ldots, propn$, resulta en las ligaduras:

$$< vobj1 > \longrightarrow F, < vobj2 > \longrightarrow D, \ldots, < vobjn > \longrightarrow B,$$

La función proyección puede aplicarse a cualquier objeto de marcado con varios atributos de memoria de trabajo.

- **proyección** ≡
  {propi $< vobji >$
  propj $< vobjj >$
  ...
  propm $< vobjn >$ }
  p.e. aplicando **proyeccion** al objeto Tupla se producen las siguientes ligaduras:
  $< vobj1 > \longrightarrow F, < vobj2 > \longrightarrow D, \ldots, < vobjn > \longrightarrow B,$

Como queda claramente reflejado en la relación de condiciones de arco anterior, la traducción de funciones de la RdPC es relativamente simple (en términos generales se puede decir que todas las condiciones de arco van a trabajar a modo de funciones de *proyección*) y puede lograrse una forma de especificar las funciones más cercana de su significado físico. Esta especificación relacional resulta más natural y favorece su manipulación simbólica, si bien en principio es menos adecuada para tratamientos numéricos.

2.5.4 Condiciones de arco actuando en relaciones de red previas

De forma similar a otros lenguajes de programación basados en reglas [BROW 86], las variables utilizadas por las condiciones de arco, tienen dos propósitos:

1. Crean un medio de comunicación con las relaciones de red posteriores del objeto de acción, con ello se posibilita la transmisión de los valores de los atributos a los que están ligados, de forma que puedan ser utilizados por las condiciones de arco de las relaciones de red posteriores.

2. Cada relación de red previa de un objeto de acción define ciertas condiciones en el reconocimiento de los elementos de la memoria de trabajo local. Las variables restringen también estos reconocimientos especificando valores que deben ser idénticos en las diferentes condiciones de arco de las relaciones de red previas del objeto de acción.

Existe todavía una restricción adicional a las ligaduras de las variables impuesta por el predicado asociado al objeto de acción, contenido en el atributo *predicado.*

Cuando a una variable específica puede ligarse el mismo valor (para cada ocurrencia en las condiciones de arco de las relaciones de red previas) y satisface

además el predicado asociado al objeto de acción, se dice que se ha encontrado una **ligadura consistente** para dicha variable. Si en un objeto de acción existe al menos una ligadura consistente, se dice que dicho objeto está **sensibilizado.**

2.5.5 Condiciones de arco actuando en relaciones de red posteriores

La estructura de las condiciones de arco en las relaciones de red posteriores es similar a las previas pero su funcionamiento se corresponde con la semántica de la parte acción de una regla. Los objetos de marcado identificados por estas condiciones de arco serán modificados de forma que cambien sus atributos a los valores indicados por las ligaduras de las variables.

Generalmente este proceso tiene un funcionamiento análogo a la acción *modify* que se utiliza en la parte derecha de las reglas en OPS5. En las ocasiones en que ninguna variable transmita la ligadura al nombre del objeto de marcado es necesario transmitir la clase a la que pertenecerá dicho objeto, y el funcionamiento se corresponde con la acción *make* de OPS5, de forma que se crea una nueva instancia de dicha clase.

2.6 Disparo de un objeto de acción

Los cambios de estado de una red KRON tienen lugar cada vez que una transición es *disparada* respecto de alguna de sus ligaduras consistentes (también llamadas *modos de disparo*). Para que un objeto de acción sea disparado es necesario, por tanto, que se encuentre sensibilizado.

El **disparo** de un objeto de acción *respecto de una ligadura* consiste en retirar, de los atributos de estado previos, los objetos de marcado que han dado lugar a dicha ligadura y depositar, en los atributos de estado posteriores, los nuevos objetos de acuerdo a las ligaduras impuestas por las condiciones de arco de las relaciones de red posteriores.

2.6.1 Ejemplo de disparo de un objeto de acción

Para clarificar estas ideas sobre el disparo de los objetos de acción, volvamos a retomar como ejemplo la situación del objeto *mundo-bloques*, indicada por los objetos de la figura 2.7. Analizaremos el proceso de disparo del objeto de acción *desapila* (representado en la figura 2.12).

Habrá que encontrar una ligadura consistente en *desapila* para, a continuación,

realizar el disparo del objeto respecto a dicha ligadura. El objeto *desapila* tiene dos relaciones de red previas que conectan con los atributos de estado previos *sobre* y *libre* disponibles en el objeto *mundo-bloques1*.

1. *(mundo-bloques1 sobre Obj1-sobre-Obj2)*

   La condición de arco es:

   $Obj1 - sobre - Obj2 \equiv$ {schema-name $< vobj1 >$ sobre $< vobj2 >$}

   Como el atributo *sobre* del objeto *mundo-bloques1* contiene únicamente el valor b_1, la única ligadura posible es: $< vobj1 > \longrightarrow b_1$ y $< vobj2 > \longrightarrow b_2$.

2. *(mundo-bloques1 libre Obj1)*

   La condición de arco es:

   $Obj1 \equiv$ {schema-name $< vobj1 >$}

   Como el atributo *libre* del objeto *mundo-bloques1* contiene los valores b_1, b_3, existen dos ligaduras posibles: $< vobj1 > \longrightarrow b_1$ y $< vobj1 > \longrightarrow b_3$.

A la vista de lo anterior, solo existe una única ligadura consistente dada por: $< vobj1 > \longrightarrow b_1$ y $< vobj2 > \longrightarrow b_2$. El disparo respecto a esta ligadura conlleva la retirada de los objetos de marcado que han dado lugar a la ligadura, es decir b_1 de *libre* y b_1 de *sobre* y por otra parte elimina b_2 del atributo *sobre* de b_1.

Para analizar lo que ocurre en los atributos de estado de salida, examinemos las relaciones de red posteriores:

1. *(mundo-bloques1 cogido Obj1)*

   Como la condición de arco es:

   $Obj1 \equiv$ {schema-name $< vobj1 >$}

   se introduce en el atributo *cogido* del objeto *mundo-bloques1*, el objeto de marcado cuyo *schema-name* corresponde con la ligadura de $< vobj1 >$, es decir b_1.

2. *(mundo-bloques1 libre Obj2)*

   Idénticamente al caso anterior, la condición de arco:

   $Obj2 \equiv$ {schema-name $< vobj2 >$}

   implica la introduccción en el atributo *libre* del objeto *mundo-bloques1*, el objeto de marcado cuyo *schema-name* corresponde con la ligadura de $< vobj2 >$, es decir b_2.

{*sincronizacion*
is-a : relation
domain : (TYPE is-a objeto-accion)
range : (TYPE is-a objeto-accion)
inverse : sincronizacion }

Figura 2.15: Definición de la relación de *sincronizacion*.

2.7 Sincronización de Objetos

En general, los sistemas dinámicos pueden ser descompuestos en un conjunto de subsistemas más simples que interaccionan, teniendo a su vez cada uno un comportamiento dinámico propio. Una vez se ha modelado el comportamiento de cada sistema por separado, es necesario establecer los adecuados mecanismos de comunicación y sincronización entre el conjunto de subsistemas con objeto de modelar completamente el comportamiento dinámico del sistema como un todo. Esto puede conseguirse mediante el establecimiento de diversos tipos de sincronizaciones entre las acciones de sus diferentes subredes[7].

Con este proposito se ha definido una relación especial denominada **relación de sincronización** (figura 2.15), que conecta dos objetos de acción cuyo funcionamiento está sincronizado. Los objetos de acción de una subred KRON que hacen posible la interconexión con otra subred se denominan **acciones de interfase.**

Un conjunto de objetos de acción conectados por relaciones de sincronización puede ser agrupado en un sólo objeto de acción que contenga la unión de todas las relaciones de red. El predicado del objeto de acción resultante es la composición *and* de los predicados de los objetos sincronizados.

El problema presentado en el ejemplo consiste en un robot trabajando en un mundo de bloques. Hasta el momento se ha tratado de la construcción de los modelos independientes que representan ambas entidades, sólo queda por definir la sincronización de acciones entre ambos modelos. Las acciones *coge* y *desapila* del mundo de bloques son ejecutadas por la acción *R-coge* del robot.

Sin embargo, esta sincronización debe ser exclusivamente bilateral porque las acciones *coge* y *desapila* no están sincronizadas entre sí. Esto se consigue creando una instancia del objeto *R-coge* por cada sincronización bilateral establecida. Esta nueva relación de sincronización es una especialización de la anterior y se denomina

[7]El problema de proponer un mecanismo general para la sincronización de redes es enormemente complicado, el presente trabajo se limita a ciertos tipos simples quedando fuera de su alcance el tratamiento en profundidad de una metodología genérica

{*desapilar*
is-a : objeto-accion
pre-relaciones-red :
(mundo-bloques1 sobre Obj1-sobre-Obj2)
(mundo-bloques1 libre Obj1)
(robot mano-vacia coge)
post-relaciones-red :
(mundo-bloques1 cogido Obj1)
(mundo-bloques1 libre Obj2)
(robot mano-ocupada pone)
dato-asociado : $< vobj1, vobj2 >$
predicado : t }

Figura 2.16: Objeto de acción *desapilar*.

sincronizacion-bilateral. La sincronización entre el objeto *desapila* y *R-coge* produce el nuevo objeto de acción *desapilar* (figura 2.16).

La figura 2.17 muestra graficamente la red KRON que modela el ejemplo completo. Existen cuatro posibles acciones u operadores (*coger, dejar, apilar* y *desapilar*) que permiten la modificación del estado del mundo de bloques.

Esta metodología de representación permite hacer una descripción consistente y precisa de los aspectos dinámicos del sistema. No obstante, la bondad de esta metodología se hace más patente cuando se aborda el tratamiento de sistemas dinámicos más complejos, donde la RAN subyacente en el modelo juega un papel más importante.

Considerese por ejemplo el problema en el que dos robots independientes están actuando sobre el mismo mundo de bloques. Para su modelado bastará disponer un segundo objeto de marcado *neutro* en el atributo *mano-vacia* del objeto *robot*. El modelo recoge la característica de similitud de comportamiento de los dos robots por lo que el comportamiento es el mismo que el de un único robot pero con capacidad duplicada. El método de interpretación de la red impide la realización de acciones que provoquen situaciones inconsistentes.

La figura 2.18 muestra un caso relativamente más complicado, en el que existen dos mundos de bloques distintos, *mundo-de-bloques1* y *mundo-de-bloques2*, sobre los que actúan sus correspondientes robots, *robot1* y *robot2*, los cuales pueden pasar bloques desde uno de los mundos de bloques hacia el otro, vía un almacén intermedio (*almacen*). Como se puede reconocer por la figura, el modelado del sistema resulta relativamente simple, una vez se han generado las clases de objetos para el mundo de bloques, el robot y el almacén. Siguientdo con metodología modular de diseño

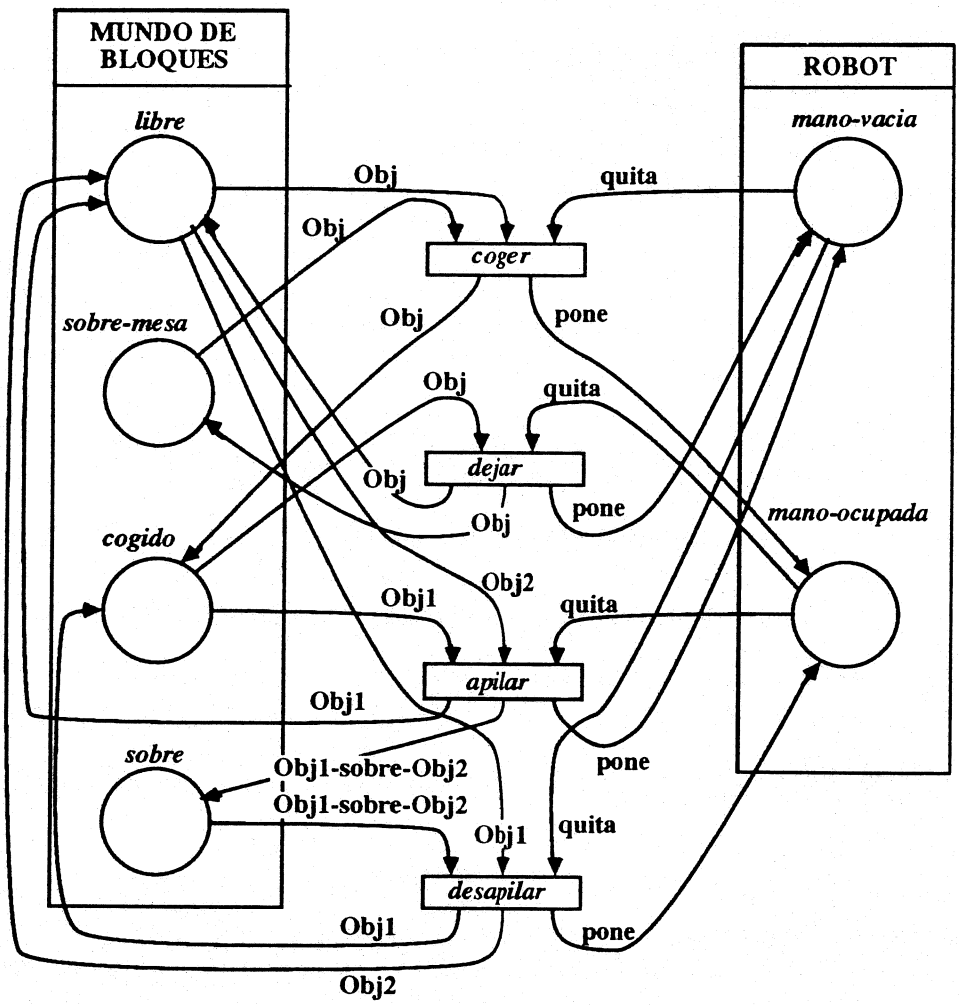

Figura 2.17: Red KRON que modela el sistema completo del mundo de bloques con robot.

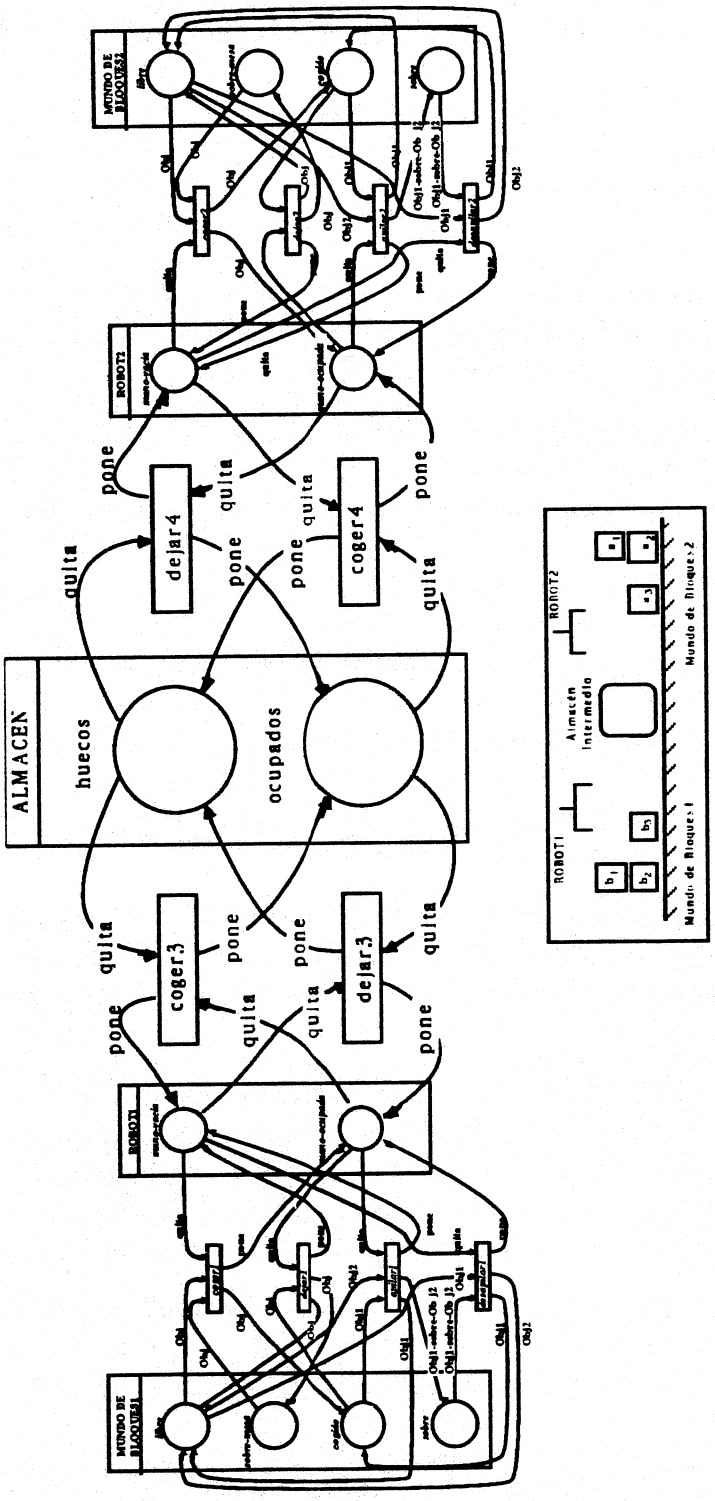

Figura 2.18: Modelo de sistema con dos mundos de bloques cuyos robots pueden comunicarse a través de un almacén intermedio.

presentada en KRON, bastará instanciar por duplicado el mundo de bloques y el robot e sincronizar sus acciones para, a continuación, hacer lo propio con el almacén intermedio. En la figura 2.18 se destacan, con un mayor tamaño, los nuevos objetos y sincronizaciones.

2.8 Conclusiones

El nivel de representación descrito permite representar conceptos básicos encontrados en inteligencia artificial y redes de Petri que resultan útiles para soportar diversas aplicaciones de sistemas de eventos discretos.

Se ha mostrado, a través de un ejemplo, una metodología de modelado con la que es posible construir fácilmente modelos de sistemas con un comportamiento dinámico discreto. La herramienta de modelado presentada facilita la definición de las entidades, cuyo comportamiento dinámico queda especificado en forma de RAN. En KRON se han definido una serie de objetos especiales (de marcado, de estado y de acción) y de relaciones (relaciones de red) que facilitan la exprexión de los elementos de una red de alton nivel (marcado, lugares, transiciones y arcos). KRON cuenta, asímismo, con un mecanismo particular de definición de clases e instanciación que permiten describir jerarquías de entidades en las que se heredan también las características identificadas por una RAN. Esta jerarquía soporta los mecanismos típicos de herencia, también de los aspectos relacionados con el comportamiento dinámico identificados por la RAN subyacente.

La integración de características de RAN en el entorno de representación permite que un único modelo pueda ser considerado desde las dos perspectivas de RdP e IA, facultando en todo momento la utilización de la perspectiva que ofrezca mejores posibilidades para la manipulación o el razonamiento sobre el modelo.

Por otra parte, el usuario puede utilizar las clases de objetos básicos para definir jerarquías de especialización de prototipos de entidades, conceptos y relaciones, de forma que puede extender la representación según sus necesidades. El conjunto de primitivas conceptuales apuntadas en este capítulo son independientes del dominio, por lo que KRON no fuerza una perspectiva de aplicación particular en el usuario.

Los ideas iniciales acerca de la integración de las perspectivas de RAN e IA se implementaron utilizando el lenguaje CRL perteneciente al entorno de trabajo KnowledgeCraft sobre un computador Explorer de Texas Instruments. El contexto de problemas desarrollados se circunscribían a aplicaciones de simulación y planificación. Estos trabajos iniciales dieron pié a la idea de KRON como herramienta de representación propia. Actualmente se dispone de una implementación parcial desarrollada en el entorno LOOPS sobre un computador XEROX[8].

[8]En el anexo A se muestra las funciones utilizadas para generar el modelo del ejemplo del

mundo de bloques que fué presentado en este capítulo, así como las clases de objetos e instancias generadas.

FLOR Y CRONOPIO

Un cronopio encuentra una flor solitaria en medio de los campos. Primero la va a arrancar.

pero piensa que es una crueldad inútil

y se pone de rodillas a su lado y juega alegremente con la flor, a saber: le acaricia los pétalos, le sopla para que baile, zumba como una abeja, huele su perfume y finalmente se acuesta debajo de la flor y se duerme envuelto en una gran paz.

La flor piensa: "Es como una flor".

Historias de cronopios y famas
JULIO CORTAZAR

Capítulo 3

Nivel Conceptual, Mecanismo de Control y Formalización

El presente capítulo tiene por objeto abordar algunos aspectos adicionales en la definición de KRON como herramienta para la representación del conocimiento. En primer lugar, se aborda la utilidad de KRON para la representación de algunos conceptos utilizados en IA como propiedades causales, temporales y de composición de estados, a partir de consideraciones estructurales de la red KRON. A continuación se apuntan algunas posibilidades de análisis del modelo en base a las características apuntadas por la red y se propone un método para el mecanismo de control de la red. Este mecanismo está dirigido por conflictos y sigue un procedimiento análogo al ciclo de control utilizado en el lenguaje OPS5. Para finalizar se presentan los objetos y relaciones de KRON desde una perspectiva más formal y se hace un sumario de las características de KRON como herramienta de representación.

3.1 Nivel de Representación Conceptual

El propósito del nivel de representación conceptual es detallar algunas características acerca de nociones tales como causalidad, relaciones temporales, posesión, etc.. Se tratará en este apartado de dar una visión intuitiva de cómo caracterizaciones estructurales de la red KRON permiten extraer cierto grado de información adicional sobre estos conceptos.

3.1.1 Tiempo

La especificación del tiempo es una característica importante en cualquier modelo de sistema dinámico. Existen dos alternativas tradicionales y complementarias para la especificación de características temporales de situaciones:

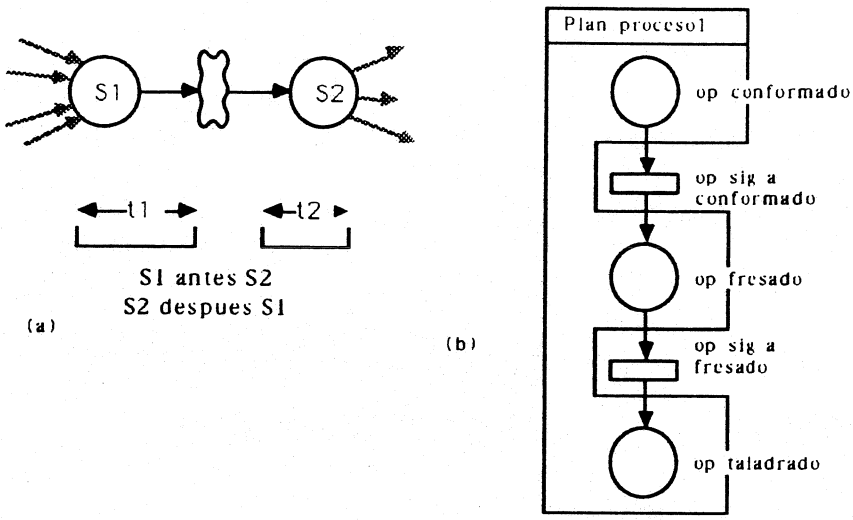

Figura 3.1: Estructura de red de Petri que implica relaciones temporales *antes/despues*.

1. La especificación de límites de tiempo en un calendario y duraciones. Esto ha producido dos tendencias en la representación del tiempo: ocurrencia de eventos sucediendo en instantes específicos [BRUC 72,McDE 82] o la ocurrencia en intervalos [ALLE 84,SMIT 83].

2. Especificación de relaciones temporales entre situaciones [ALLE 84,FOX 83, SMIT 83]. Esta aproximación es utilizada mayormente para descripciones prototípicas donde las relaciones proporcionan una guía en el proceso de inferencia temporal.

Para la primera alternativa de especificación temporal, KRON adopta la representación de tiempo propuesta por [SMIT 83]. Con respecto a la segunda alternativa, la estructura de la red proporciona un mecanismo para la especificación de relaciones temporales que se presenta en forma gráfica y puede estar más cerca de la noción de relación temporal que posee el usuario. Por otra parte, el formalismo subyacente proporciona un mecanismo para comprobar propiedades temporales, desarrollado a partir de la teoría de sincronía en redes de Petri [SIL 86b].

Para hacerse una idea más clara de este aspecto, considerese el modelado con KRON del sencillo plan de trabajo que se muestra en la figura 3.1b. El objetivo de este apartado es comprobar como la estructura de la red KRON puede ayudar a esclarecer nociones temporales del modelo, por ejemplo que la operación de fresado (*op-fresado*) ha de realizarse *antes* que la de taladrado (*op-taladrado*).

Para establecer la conexión entre la noción de tiempo dada por la relación *antes* y alguna estructura de red, consideraremos una situación representada por el marcado

de un atributo de estado. En este contexto, se puede decir que el hecho de que una situación $S1$ vaya a ocurrir **antes** que otra situación $S2$, puede ser modelado mediante una red KRON cuya estructura es similar a la mostrada en la figura 3.1a. Considerando una situación dinámica, una marca presente en el lugar $S2$ debe, necesariamente, haber estado presente previamente en el atributo de estado $S1$, e inversamente, un objeto de marcado presente en $S1$, estará presente en un futuro, *despues*, en $S2$. Según esto, se pueden sacar las siguientes conclusiones de la figura 3.1b:

- intervalo de tiempo de *op-conformado* anterior ($<$) al intervalo de tiempo *op-fresado*
- *op-conformado* se realizará *antes* que *op-fresado*
- *op-taladrado* se realizará *despues* que *op-fresado*

La relación temporal **encuentra** indica que el final de una situación está sincronizado con el comienzo de otra. Este hecho puede ser modelado conectando los atributos de estado representantes de ambas situaciones mediante un objeto de acción, teniendo el primer atributo de estado como entrada y el segundo como salida. La especificación estructural de la relación *encuentra* y su inversa *encontrada-por* se muestran en la figura 3.2a.

Otras relaciones temporales clásicas [ALLE 84,SMIT 83] pueden reconocerse también a través de características estructurales de una red KRON:

- **Durante** especifica que una situación $S2$ tiene lugar durante otra $S1$, su inversa es *incluye* (figura 3.2b).
- **Tiempo-igual** especifica una situación $S1$ que comparte el mismo intervalo de tiempo que otra situación $S2$, su inversa es también *tiempo-igual* (figura 3.2c).
- **Mismo-comienzo** especifica que una situación $S1$ comienza en el mismo instante que otra situación $S2$, su inversa es también *mismo-comienzo* (figura 3.2d).
- **Mismo-final** especifica que la situación $S1$ termina en el mismo instante que la situación $S2$, su inversa es también *mismo-final*, (figure 3.2e).
- **Superpone** especifica que una situación $S1$ comienza antes que otra situación $S2$, pero termina después que $S2$ haya comenzado y antes que $S2$ haya terminado, su inversa es *superpuesto-por* (figura 3.2f).

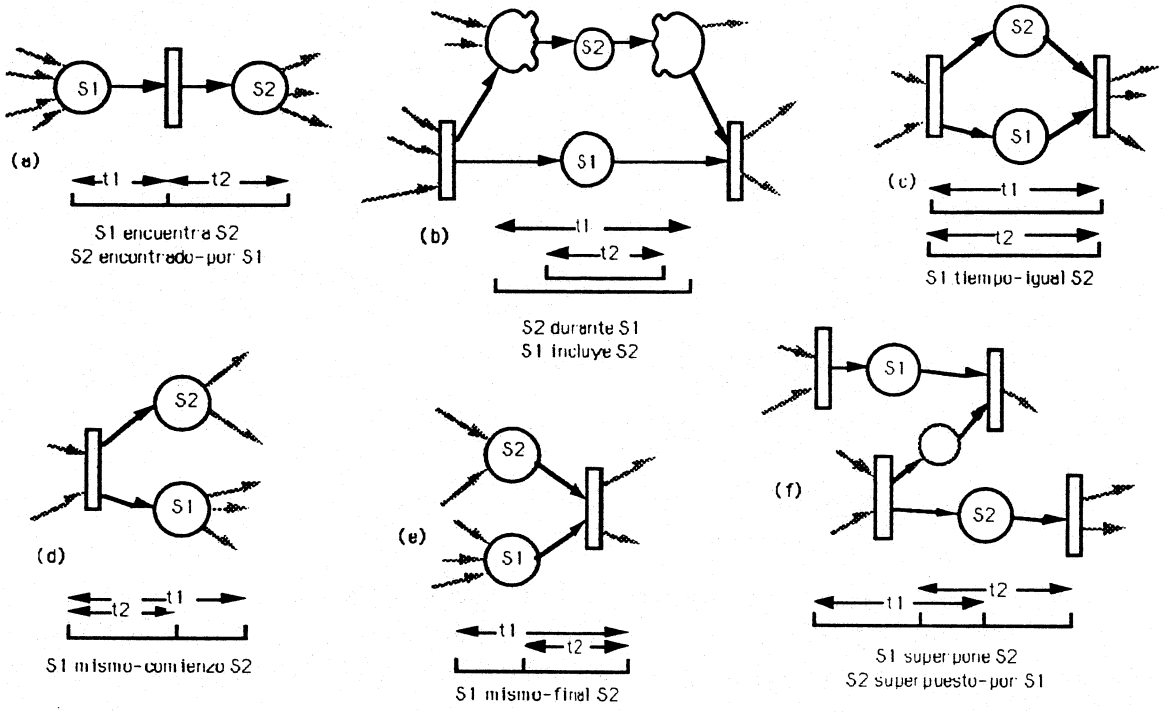

Figura 3.2: Estructuras de redes de Petri expresando varias relaciones temporales.

3.1.2 Causalidad

Cualquier herramienta de modelado de sistemas dinámicos debe tener la capacidad de representar estados y sus cambios. En un sistema causal, un cambio de estado es causado por la aplicación de una acción, ante esta posible ocurrencia es necesario especificar dos características:

1. Las condiciones bajo las cuales se considera que la acción ha ocurrido.
2. Los efectos ocasionados por dicha acción.

Las RdP permiten representar y abordar el tratamiento de algunas relaciones causales. Adicionalmente, las capacidades gráficas de las RdP proporcionan una caracterización intuitiva de los efectos causales, cercana a la percepción del usuario. Por otra parte, la base formal de las RdP exige un alto grado de disciplina en el modelado de las redes causales, lo que facilita el mantenimiento de su consistencia y elimina al mismo tiempo ambiguedades en su interpretación.

La causalidad se representa en KRON por medio de relaciones y características estructurales de la RdP subyacente. La base conceptual está derivada de la teoría de representación del conocimiento humano tipo *causa-efecto* propuesta en [RIEG 77] para representar mecanismos físicos, y la metodología de representación se deriva

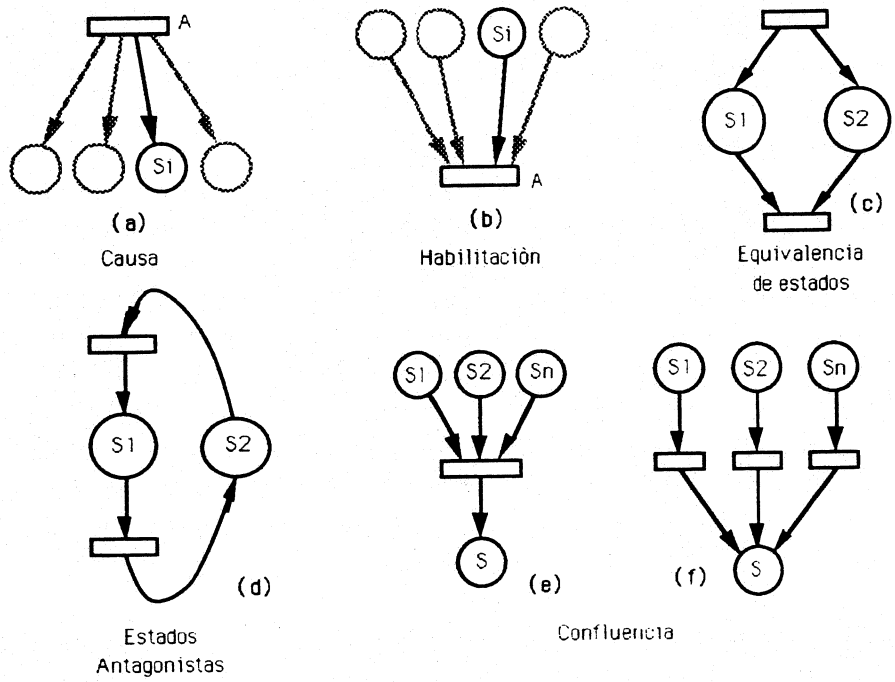

Figura 3.3: Relaciones y estructuras de red modelando diferentes conexiones causales.

en parte de la aproximación presentada en [FOX 85,SATH 85] para aplicaciones en fabricación y dirección de proyectos. KRON adopta algunas características de estas aproximaciones con objeto de integrar una perspectiva basada en RdP.

[RIEG 77] y [SATH 85] presentan una caracterización relacional de la causalidad, KRON extiende las capacidades de tipo relacional incluyendo características estructurales de la red. Así por ejemplo, los conceptos de causa y habilitación se derivan naturalmente de la noción de relación de red en una red KRON. Otros conceptos causales como equivalencia o antagonismo de estados son expresados por medio de condiciones estructurales de la RdP.

Siguiendo la perspectiva causal, una estructura de KRON puede ser interpretada como una red causal. En los siguientes párrafos se muestra brevemente como diferentes conexiones causales, propuestas en los trabajos anteriores, pueden ser caracterizadas en las estructuras de red de los sistemas modelados utilizando KRON:

- **Causa**. Una acción A causa que exista un estado B. Se representa por una relación de red posterior que conecta un objeto de acción con un atributo de estado. Siguiendo la semántica de las RdPs, el disparo de un objeto de acción, causa que el cambio se propague a los atributos de estado siguiendo los caminos de las relaciones de red posteriores. Figura 3.3a.

- **Habilitación.** Un estado Si habilita parcialmente que tenga lugar una acción A. Está representado por las relaciones de red previas que conectan los atributos de estado con un objeto de acción. Cada relación de red previa representa una condición parcial de puerta[1] impuesta por la condición de arco (apartado 2.5). Figura 3.3b.

- **Equivalencia de estados.** Una situación $S1$ es una formulación equivalente (es decir, representan expresiones sintácticamente diferentes de la misma situación subyacente) de otra situación $S2$. Existen diferentes posibilidades estructurales que pueden ser utilizadas para modelar esta equivalencia, en general corresponden con la idea de lugar implícito en una RdP [SIL 85b]. La figura 3.3c muestra una configuración típica para dos situaciones equivalentes (lugares equivalentes estructuralmente) que juegan el papel de dos puntos de vista diferentes sobre el mismo evento. La existencia o no existencia de cualquiera de las situaciones equivalentes implica la existencia o no existencia de la otra.

- **Estados antagonistas.** La situación $S1$ es antagonista con la situación $S2$. Esto implica que ambas situaciones no pueden existir al mismo tiempo. Este hecho puede modelarse estructuralmente con dos lugares en exclusión mutua como queda mostrado en la figura 3.3d (coincide con la idea de lugares complementarios en RdP).

- **Confluencia.** La situación S representa la culminación de múltiples fuentes causales $S1$, $S2$, ... , Sn. Las figuras 3.3e y 3.3f representan diferentes posibilidades estructurales para el modelado de la confluencia.

Ejemplo de representación causal

El concepto de causalidad y las conexiones causales representadas anteriormente se pueden comprender mejor analizando el siguiente ejemplo de precedencia en el secuenciamiento de operaciones de fabricación. Con objeto de establecer las diferencias con la aproximación adoptada en [FOX 83,FOX 85] y presentada en el punto 1.2 se seguirá el ejemplo allí presentado.

En el ejemplo considerado pueden distinguirse varias entidades como son operador, llave, centro de trabajo y operaciones con un comportamiento dinámico. La figura 3.4 representa las redes KRON subyacentes en estas entidades cuando son consideradas independientemente. Las figuras 3.4a, 3.4b y 3.4c representan tres de los recursos mencionados anteriormente (*operadores*, *centro-tabajo-48* y *llave* respectivamente), que se han modelado con un comportamiento dinámico muy simple consistente en dos posibilidades básicas: recurso trabajando o disponible. La

[1] "Gate condition" según su denominación inglesa.

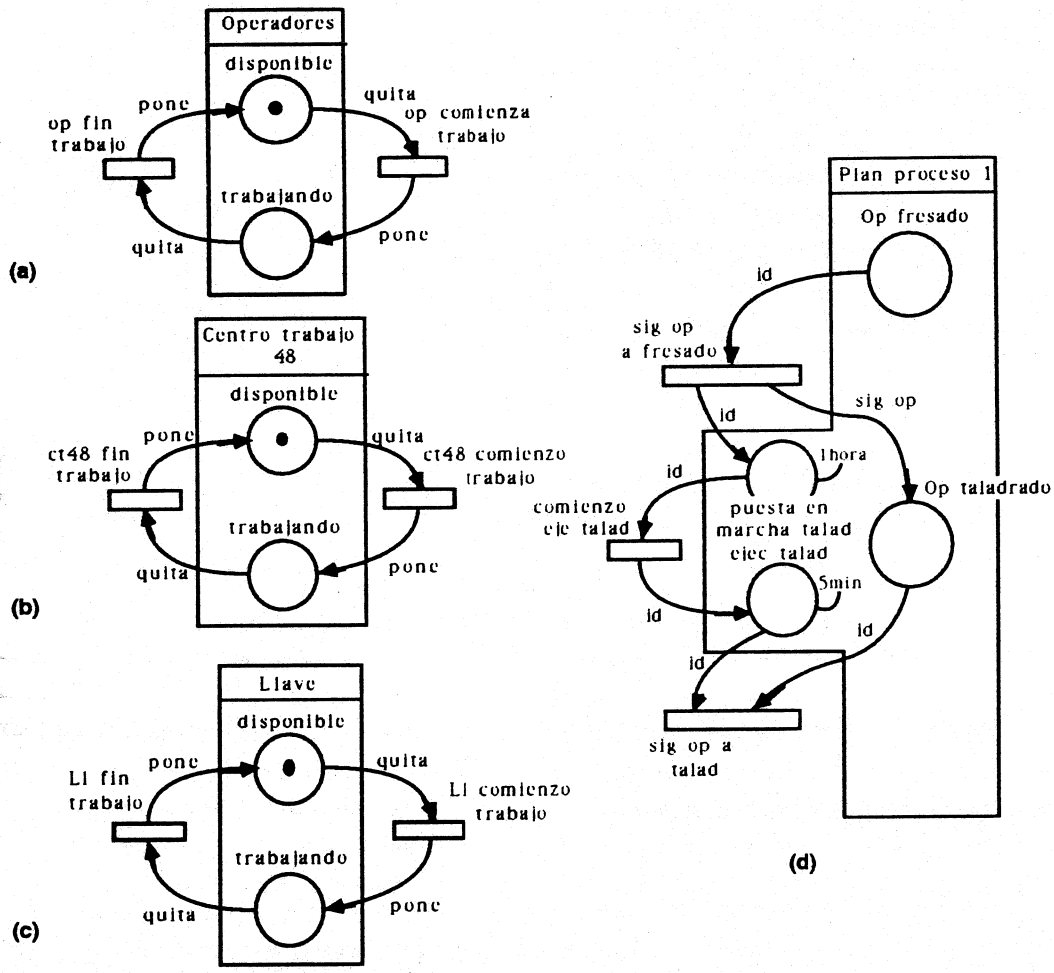

Figura 3.4: Ejemplo de secuenciamiento de operaciones de fabricación. Redes KRON representando las entidades independientes.

{*comienza-op-taladrado*
is-a : objeto-accion
pre-relaciones-red :
(centro-trabajo-48 disponible quita)
(operadores disponible quita)
(llave disponible quita)
(plan-proceso1 op-fresado obj)
post-relaciones-red :
(centro-trabajo-48 trabajando pone)
(operadores trabajando pone)
(llave trabajando pone)
(plan-proceso1 op-taladrado obj)
(plan-proceso1 puesta-en-marcha-talad obj)
dato-asociado : < *vobj* >
predicado : t }

donde:
obj ≡ {schema-name < *vobj* >}
quita, pone ≡ {schema-name < *vneutro* >}

Figura 3.5: Objeto de acción *comienza-op-taladrado.*

figura 3.4d representa parcialmente la red de un plan de proceso denominado *plan-proceso1* que contempla dos operaciones, *op-fresado* y *op-taladrado.* La operación de taladrado puede ser descompuesta a su vez en dos suboperaciones de *puesta-en-marcha* y *ejecucion* que la representan a un nivel de abstracción inferior.

La red causal completa se construye componiendo las redes causales parciales de sus componentes, siguiendo la metodología de modelado presentada en el capítulo anterior (sincronizando los objetos de acción). La red KRON resultante de la sincronización se muestra en la figura 3.6. El objeto de acción *comienza-op-taladrado* está representado en la figura 3.5.

Siguiendo el marco de representación de la causalidad definido arriba, se pueden reconocer varias conexiones causales atendiendo a la estructura de la red KRON:

- Para que de comienzo la operación de taladrado es necesario que se encuentre disponible un operador, una llave y el centro de coste 48, por otra parte se necesita una pieza que haya completado su operación de fresado.

- Mientras se está procesando la operación de taladrado, no se encuentra disponible ninguno de los recursos y pasan a una situación de *en-proceso.*

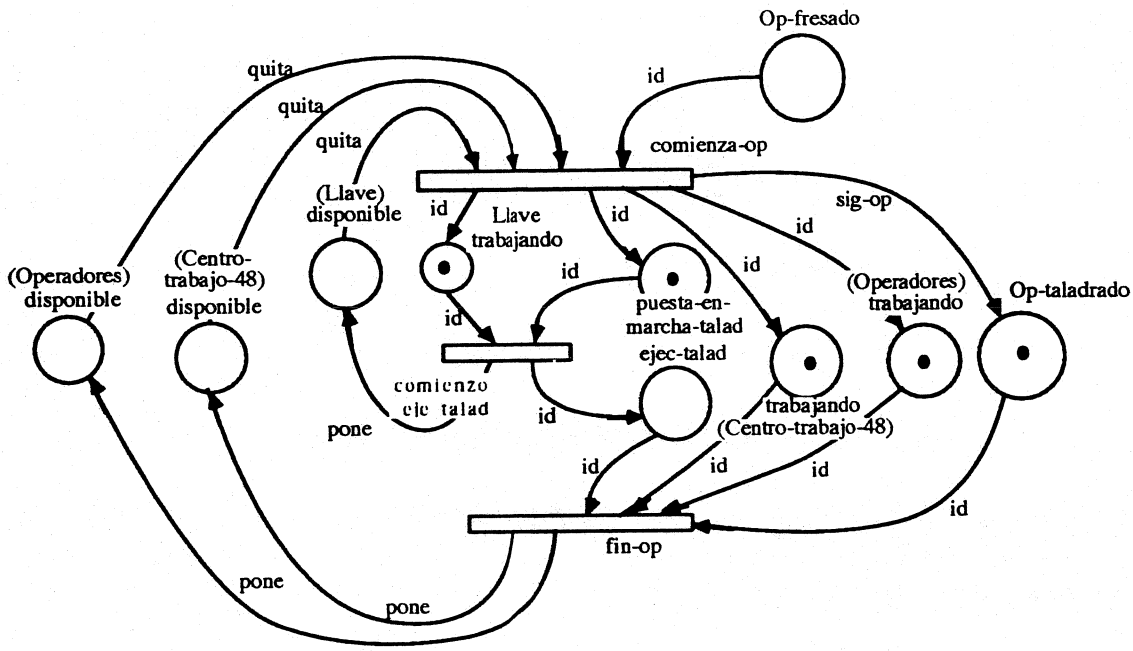

Figura 3.6: Ejemplo de secuenciamiento de operaciones de fabricación. Red KRON representando las entidades interconectadas.

- La secuencia de las situaciones de puesta en marcha y ejecución son *equivalentes* a la operación de fresado, y son también *tiempo-igual.*
- Las situaciones en proceso del operador y el centro de trabajo 48 son *equivalentes* y son también *tiempo-igual.*
- La situación de trabajando en la llave es *equivalente* a la de puesta en marcha del taladro y es también *tiempo-igual.*
- En los recursos, una situación de disponibilidad es un *estado antagonista* de una situación de trabajando.

3.1.3 Concurrencia, paralelismo y estados compuestos

En el mundo real, actos o cambios en general, pueden ocurrir en paralelo. La representación de habilitación y causalidad puede especificar más de una situación o acto siguiente. Las RdPs se han utilizado extensivamente para modelar procesos concurrentes. En KRON, los conceptos de concurrencia y paralelismo quedan reflejados por la semántica de la red subyacente.

En algunas aproximaciones como [FOX 83], se lleva a cabo la especificación de este paralelismo introduciendo estados compuestos, que son construidos con descripciones de estado simples. En KRON, los estados compuestos están derivados

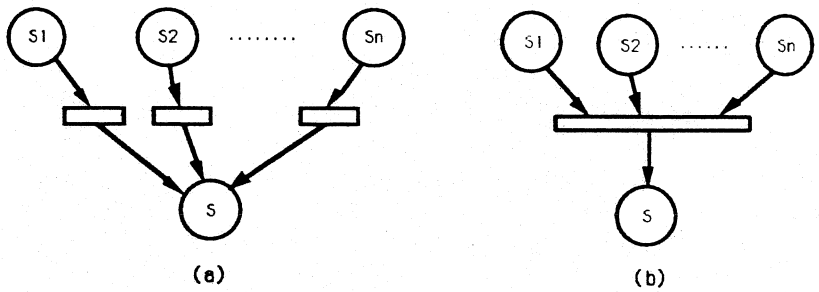

Figura 3.7: Estructuras de red para estados compuestos.

de las conexiones de red. Las composiciones de estados más usuales se pueden modelar sencillamente mediante la composición de relaciones de red como muestra el ejemplo de la figura 3.7:

- **Composición OR**: Se especifica mediante las relaciones de red posteriores que llegan a un atributo de estado. La situación S se verifica cuando cualquiera de los $S1 \ldots Sn$ se ha verificado previamente. Dicho en otras palabras, la situación S está habilitada por cualquiera de las situaciones $S1 \ldots Sn$. Figura 3.7a.
- **Composición AND**: Está especificada por las relaciones de red previas que llegan a un objeto de acción. Para que se verifique la situación S deben haberse verificado previamente todas las situaciones $S1 \ldots Sn$. Figura 3.7b.

3.2 Análisis del modelo

El proceso de modelado y diseño de sistemas de fabricación es una tarea relativamente compleja, fundamentalmente si se trata de sistemas con un comportamiento flexible (Sistemas de Fabricación Flexible). Así antes de pasar a su implementación resulta útil validar algunas etapas durante la construcción del modelo. Por otra parte, el alto coste de implantación de estos sistemas hace necesario el empleo de técnicas cuantitativas para el análisis previo de su dimensionamiento y prestaciones.

Un aspecto destacado de las RdPs es su capacidad para ser analizadas y comprobar de esta forma si el modelo construido verifica algunas propiedades de su especificación. En [SILV 89] se propone la distinción entre dos categorias de análisis de modelos:

1. **Análisis cualitativo** Se concentra en el estudio de propiedades tales como vivacidad, ausencia de bloqueos, limitación, exclusiones mutuas, conflictos,

etc.. Existen tres técnicas básicas de análisis cualitativo para redes de Petri de bajo nivel (ordinarias, generalizadas, etc.), la extensión de estos métodos a las RAN está bajo desarrollo:

- análisis por enumeración [HUBE 84],
- análisis estructural [SIL 85a], y
- análisis por reducción [COLO 87].

2. **Análisis cuantitativo** Su propósito es evaluar las prestaciones del sistema modelado obteniendo información sobre ratios de ocupación de las máquinas y del sistema de transporte; tamaños medios de colas de espera; tiempos medios de espera, etc.. Para el análisis cuantitativo de modelos se han utilizado *métodos formales*, entre los que cabe destacar los basados en modelos de RdP temporizados y estocásticos, y métodos basados en *simulación*.

Las técnicas de análisis basadas en *teoría de sincronía* de redes pueden situarse entre ambos tipos de análisis. La caracterización de propiedades tales como equidad[2] y el cálculo de parámetros tales como distancias sincrónicas, avances sincrónicos[3] o desviaciones en el número de disparos entre dos transiciones o entre dos grupos de transiciones, puede ser utilizado para caracterizar parcialmente el comportamiento dinámico de un modelo [SIL 86a,SIL 86b].

Aunque el desarrollo en detalle de métodos de análisis de redes escapa del alcance de este trabajo, a continuación se va desarrollar un ejemplo, para comprobar la utilidad y viabilidad de su aplicación en sistemas modelados con KRON. Inicialmente se realiza el modelado del sistema utilizando KRON. Con objeto de aprovechar las conocidas técnicas de análisis de RAN, se mostrará un mecanismo, fácilmente automatizable, para el paso de un modelo KRON a una especificación de RAN. A continuación se propondrán algunas posibilidades de análisis, tanto para comprobar ciertas propiedades estructurales del modelo, como para detectar los puntos de decisión, importantes en las subsiguientes etapas de toma de decisiones dentro de la estructura jerárquica de control.

3.2.1 Ejemplo de modelado con KRON

El ejemplo consiste en una celda de fabricación con seis estaciones de trabajo y un sistema de transporte formado por seis tablas que ya fue introducido en el apartado 1.3.2 modelándose en base a RAN (más concretamente con redes coloreadas).

Tablas, estaciones y piezas son objetos físicos cuyo estado va evolucionando en el sistema. Para su representación se definen tres clases de objetos de marcado que recogerán sus peculiaridades comunes:

[2] "Fairness" según su más conocida denominación inglesa.

[3] "Synchronic leads" según su denominación inglesa.

- **Tabla**: Entre las tablas existe una relación de orden, *sig-tabla*, que indica cual es la tabla que está a continuación. Cada tabla tiene asociada una estación (la ordenación de las tablas define la ordenación de las estaciones, aunque se podría haber utilizado la estrategía inversa), *con-estacion*, y puede contener una pieza (*pieza*). El atributo *instance+inv* recoge todas las instancias que han sido creadas de dicha clase.

  {*tabla*
  is-a : objeto-marcado
  instance+inv : T_1 T_2 T_3 T_4 T_5 T_6
  sig-tabla :
  con-estacion :
  pieza : }

- **Estacion**: *Con-tabla* es la relación inversa a *con-estacion* y *piezas-asignadas* indica las piezas que se ha decidido sean realizadas en la estación.

  {*estacion*
  is-a : objeto-marcado
  instance+inv : S_1 S_2 S_3 S_4 S_5 S_6
  con-tabla :
  piezas-asignadas :
  pieza : }

- **Pieza**: *Localizacion* relaciona la pieza con el lugar en que se encuentra (relación inversa a *pieza*). *Estacion-asignada* indica la estación en que ha de procesarse la pieza (inversa de *pieza-asignada*) y *sentido* la dirección de entrada o de salida.

  {*pieza*
  is-a : objeto-marcado
  instance+inv : X Y Z R S T D V W
  sentido :
  estacion-asignada :
  localizacion : }

- **Sentido-piezas**: Es un atributo no físico que se utiliza para indentificar las piezas que están entrando o saliendo.

  {*sentido-piezas*
  is-a : objeto-marcado
  inverse+inv : *ent* *sal*
  sentido-de : }

El subsistema de transporte puede representarse por una red KRON con dos atributos de estado *tablas* y *tablaslibres*) y cinco acciones (*entrada*, *salida*, *sigtabla*, *carga-est* y *descarga-est*). Los valores del atributo de estado *tablaslibres* ($T_1 \ldots T_6$) indican las tablas que se encuentran libres en el instante inicial.

{*transp-tablas-R*
instance : transporte-tablas
estados : tablas tablaslibres
tablas :
type : slot-estado
dato-asociado : (TYPE is-a pieza)
tablaslibres : T_1 T_2 T_3 T_4 T_5 T_6
type : slot-estado
dato-asociado : (TYPE is-a tabla)
acciones : entrada salida sigtabla carga-est descarga-est
nivel-precision : nivel-precision-celda
descripcion : }

Identicamente se puede representar el subsistema de estaciones mediante otra red KRON con seis atributos de estado (*C*, *CC*, *P*, *CP*, *D* y *CD*) y cuatro objetos de acción (*carga-st*, *carga-pu*, *descarga-pu*, *descarga-st*).

{*celda-estaciones-R*
instance : celda-estaciones
estados : C CC P CP D CD
C :
dato-asociado : (TYPE is-a pieza)
CC : S_1 S_2 S_3 S_4 S_5 S_6
dato-asociado : (TYPE is-a estacion)
P :
dato-asociado : (TYPE is-a pieza)
CP : S_1 S_2 S_3 S_4 S_5 S_6
dato-asociado : (TYPE is-a estacion)
D :
dato-asociado : (TYPE is-a pieza)
CD : S_1 S_2 S_3 S_4 S_5 S_6
dato-asociado : (TYPE is-a estacion)
acciones : carga-st carga-pu descarga-pu descarga-st
nivel-precision : nivel-precision-celda
descripcion : }

El listado de los objetos y condiciones de red del modelo se encuentra en el apéndice B. En la figura 3.8 se muestra gráficamente la red KRON resultante del modelado del sistema completo.

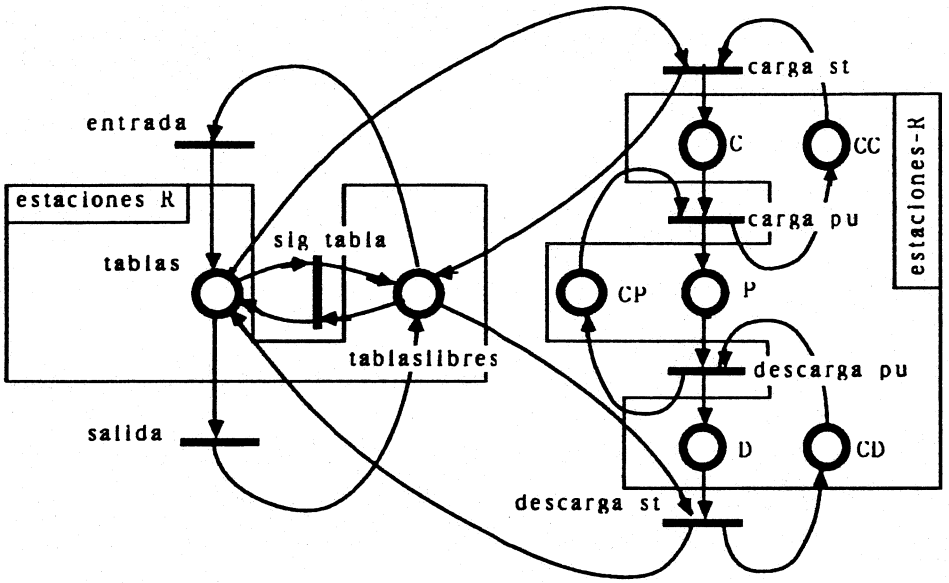

Figura 3.8: Red KRON para el sistema del ejemplo.

3.2.2 Modelo contemplado según la perspectiva de RAN

En este apartado trataremos de identificar la red de Petri subyacente en la red KRON y representarla como una RAN. Comenzemos reconociendo los posibles *conjuntos de colores.*

De las distintas clases de objetos de marcado pueden obtenerse los conjuntos de colores simples que corresponden con todas las instancias realizadas de la clase y que se encuentran en el atributo *instance+inv*:

Conjunto de **estaciones**:

$$Est \equiv \{< S_1 >< S_2 >< S_3 >< S_4 >< S_5 >< S_6 >\}$$

Conjunto de **tablas**:

$$Tab \equiv \{< T_1 >< T_2 >< T_3 >< T_4 >< T_5 >< T_6 >\}$$

Conjunto de **piezas**:

$$Pie \equiv \{< X >< Y >< Z >< R >< S >< T >< D >< V >< W >\}$$

Conjunto de **sentidos**:

$$Sen \equiv \{< ent >< sal >\}$$

Las relaciones de atribución y especificación de los objetos de marcado permiten obtener conjuntos de colores compuestos. Por ejemplo, el color compuesto por pieza y estación asignada se encuentra embebido en la clase de objeto *pieza* que dispone de la relación *estacion-asignada*:

{*estacion-asignada*
is-a : atribucion
domain : (TYPE is-a pieza)
range : (TYPE is-a estacion)
inverse : piezas-asignadas
predicado : t
description : "estacion a la que es asignada una pieza" }

De la propia definición de la relación se puede deducir el conjunto de colores compuestos, puesto que *estacion-asignada* conecta objetos de tipo *pieza* y *estacion*. Los conjuntos de colores compuestos se encuentran también restringidos por el predicado asociado a la relación. El ser una relación de atribución significa que cualquier objeto de *pieza* puede estar relacionado con cualquier objeto de *estacion* (en las relaciones de especificación las conexiones son fijas y los conjuntos de colores hay que buscarlos expresamente en las propias instancias):

$$PieEst = Pie \times Est \equiv \{< X, S_1 > \ldots < X, S_6 > \ldots < W, S_6 >\}$$

Por otra parte, los diferentes subconjuntos de colores (simples o compuestos) asociados a los atributos de estado y objetos de acción, se identifican a partir de los datos asociados a ambos, de las condiciones de red y los predicados asociados a relaciones de atribución y objetos de acción.

En el caso de los atributos de estado (lugares), su dato asociado indica el tipo de objeto de marcado que podrá contener. Si el color es compuesto, los colores simples componentes están indicados por las relaciones de atribución o especificación utilizadas en las condiciones de red que parten o llegan a dicho atributo. Las relaciones de especificación que conectan objetos del mismo tipo no agregan colores simples a la composición puesto que, al ser fijas su consideración resultaría redundante.

- Objeto **transp-tablas-R**

  tablas : El dato asociado es de tipo *pieza* y los conjuntos de objetos de las relaciones de atribución a las que se accede (*sentido, estacion-asignada* y *tabla*) son Sen, Est y Tab. Su conjunto de colores es el dado por:

  $$Pie \times Sen \times Est \times Tab$$

tablaslibres : El dato asociado es de tipo *tabla* y el conjunto de colores está formado por los colores simples Tab (debido a que la relación *sig-tabla* es de especificación).

- Objeto **celda-estaciones-R**

  C : $Pie \times Est$
  CC : Est
  P : $Pie \times Est$
  CP : Est
  D : $Pie \times Est$
  CD : Est

Para el caso de los objetos de acción (transiciones), el conjunto de colores está indicado por las clases de objetos de las variables de su dato asociado, restringidas por el predicado del objeto de acción.

- Objetos de acción relacionados con **transp-tablas-R**

  entrada : $Pie \times Est \times \{T_1\} \times \{< ent >\}$
  salida : $Pie \times Est \times \{T_6\} \times \{< sal >\}$
  sig-tabla : $Pie\times$
  $\{\{< S_1 >\} \times Tab \times \{< sal >\}$
  $\{< S_2, T_1, ent >\}$
  $\{< S_2 >\} \times \{< T_2 > \ldots < T_6 >\} \times \{< sal >\}$
  $\{< S_3 >\} \times \{< T_1 >< T_2 >\} \times \{< ent >\}$
  $\{< S_3 >\} \times \{< T_3 > \ldots < T_6 >\} \times \{< sal >\}$
  $\{< S_4 >\} \times \{< T_1 > \ldots < T_3 >\} \times \{< ent >\}$
  $\{< S_4 >\} \times \{< T_4 > \ldots < T_6 >\} \times \{< sal >\}$
  $\{< S_5 >\} \times \{< T_1 > \ldots < T_4 >\} \times \{< ent >\}$
  $\{< S_5 >\} \times \{< T_5 >< T_6 >\} \times \{< sal >\}$
  $\{< S_6 >\} \times \{< T_2 > \ldots < T_6 >\} \times \{< ent >\}$
  $\{< S_6, T_1, sal >\}\}$

- Objetos de acción relacionadas con **celda-estaciones-R**

  carga-pu : $Pie \times Est$
  descarga-pu : $Pie \times Est$

- Objetos de acción relacionados con **transp-tablas-R** y **celda-estaciones-R** que están sincronizados:

$$\textbf{carga-st} : Pie \times Est \times \{< S_1, T_1 > \ldots < S_6, T_6 >\} \times < ent >$$
$$\textbf{descarga-st} : Pie \times Est \times \{< S_1, T_1 > \ldots < S_6, T_6 >\} \times < sal >$$

Idénticamente a las funciones que etiquetan los arcos de las redes de Petri coloreadas, las condiciones de arco y los predicados utilizados en KRON pueden expresarse matricialmente a modo de funciones lineales. Estas expresiones matriciales se pueden calcular explícitamente considerando cada posible ligadura de las variables asociadas a objetos de acción y atributos de estado (transiciones y lugares según la perspectiva de las RdP).

Considerese, por ejemplo, la condición de arco *fest* que conecta el atributo de estado *CC* con el objeto de acción *carga-pu* para el caso en que únicamente exista la pieza $< X >$. *Carga-pu* tiene como dato asociado las variables $< vpieza >$ y $< vestacion >$, mientras que *CC* dispone de la variable $< vestacion >$. Si suponemos las ligaduras $< vpieza > \longrightarrow X$ y $< vestacion > \longrightarrow E_1$ en *carga-pu*, la variable $< vestacion >$ queda ligada a la estación E_1 y no al resto de estaciones, lo que se puede representar con un 1 para la coordenada $(< X, E_1 >, E_1)$ y un 0 para el resto. Siguiendo un procedimiento idéntico para la representación del resto de posibles ligaduras conduce a la matriz:

$$fest = \begin{array}{c} \\ < X, E_1 > \\ < X, E_2 > \\ < X, E_3 > \\ < X, E_4 > \\ < X, E_5 > \\ < X, E_6 > \end{array} \begin{array}{c} \begin{array}{cccccc} < E_1 > & < E_2 > & < E_3 > & < E_4 > & < E_5 > & < E_6 > \end{array} \\ \left(\begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right) \end{array}$$

Para poder realizar ciertos análisis de la red resulta necesario conocer su matriz de incidencia, generalmente especificada en modo simbólico. La figura 3.9 muestra la transpuesta de la matriz de funciones lineales de la red del ejemplo.

3.2.3 Características estructurales del modelo del sistema

Las especificaciones de la red anterior puden utilizarse para obtener información importante, útil para completar la etapa de diseño. A modo de ilustración se mencionan a continuación algunas caracterizaciones que pueden obtenerse realizando un análisis estructural de la RdPC subyacente, particularmente mediante el método de invariantes.

Realizando el cálculo de invariantes simbólicos en la matriz de la figura 3.9

	tablas	tablas-libres	C	CC	P	CP	D	CD
entrada	fPETS	-ftabla						
salida	-fPETS	ftabla						
sig-tabla	-fPETS	-ftabla						
	fPESS	fsig-tabla						
carga-st	-fPETS	ftabla	fPE	-fest				
carga-pu			-fPE	fest	fPE	-fest		
descarga-pu					-fPE	fest	fPE	-fest
descarga-st	fPETS	-ftabla					-fPE	fest

Figura 3.9: Representación simbólica de la transpuesta de la matriz de incidencia.

aplicando el método presentado en [SIL 85a], se obtienen cuatro invariantes[4]:

1. *(marcado estacion-asignada celda-estaciones-R C)*
   + *(marcado schema-name celda-estaciones-R CC)*
   $= (S_1\ S_2\ S_3\ S_4\ S_5\ S_6)$

2. *(marcado estacion-asignada celda-estaciones-R P)*
   + *(marcado schema-name celda-estaciones-R CP)*
   $= (S_1\ S_2\ S_3\ S_4\ S_5\ S_6)$

3. *(marcado estacion-asignada celda-estaciones-R D)*
   + *(marcado schema-name celda-estaciones-R CD)*
   $= (S_1\ S_2\ S_3\ S_4\ S_5\ S_6)$

4. *(marcado localizacion transp-tablas-R tablas)*
   + *(marcado schema-name transp-tablas-R tablas-libres)*
   $= (T_1\ T_2\ T_3\ T_4\ T_5\ T_6)$

Cada invariante puede ser traducido en una especificación funcional del modelo. El diseñador puede verificar si estas especificaciones corresponden con el comportamiento deseado para el sistema para, en su caso, reconsiderar el diseño. Por ejemplo, el invariante 4 indica que, en cualquier instante, una tabla está libre o es la localización de una única pieza.

Estos invariantes proporcionan también información acerca de cotas superiores del marcado, como queda reflejado en la figura 3.10.

A partir del análisis cualitativo del modelo es posible deducir que el sistema puede **bloquearse** si, en un instante dado, se asignan más de cuatro piezas a una misma estación. Un marcado, que verifique los valores representados en la figura

[4] *Marcado* es una función que devuelve la lista de los valores en el atributo *estacion-asignada*, de los objetos de marcado contenidos en el atributo de estado C del objeto *celda-estaciones-R*.

Objeto estado	Atributo estado	Atributo	Cota
transp-tablas-R	tablas	localizacion	$\leq (T_1 \ldots T_6)$
transp-tablas-R	tablas-libres	schema-name	$\leq (T_1 \ldots T_6)$
celda-estaciones-R	C	estacion-asignada	$\leq (S_1 S_2 S_3 S_4 S_5 S_6)$
celda-estaciones-R	CC	schema-name	$\leq (S_1 S_2 S_3 S_4 S_5 S_6)$
celda-estaciones-R	P	estacion-asignada	$\leq (S_1 S_2 S_3 S_4 S_5 S_6)$
celda-estaciones-R	CP	schema-name	$\leq (S_1 S_2 S_3 S_4 S_5 S_6)$
celda-estaciones-R	D	estacion-asignada	$\leq (S_1 S_2 S_3 S_4 S_5 S_6)$
celda-estaciones-R	CD	schema-name	$\leq (S_1 S_2 S_3 S_4 S_5 S_6)$

Figura 3.10: Tabla de cotas de los atributos de estado.

{*transp-tablas-R*
tablas : $\ldots T_i \ldots$}
{*celda-estaciones-R*
C : $\ldots S_i \ldots$
P : $\ldots S_i \ldots$
D : $\ldots S_i \ldots$}
; con:
{T_i
con-estacion : S_i
pieza : $pieza_j$ }
{$pieza_j$
sentido : *ent*
estacion-asignada : S_i }

Figura 3.11: Marcado para que se verifique la condición de bloqueo.

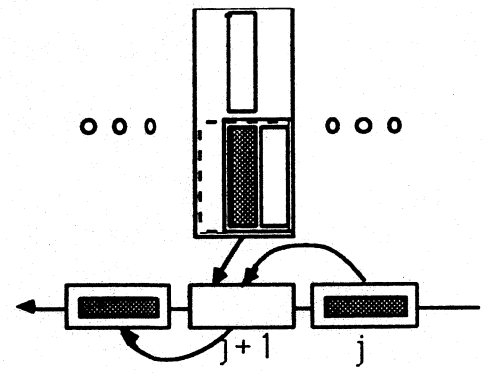

Figura 3.12: Situación de conflicto.

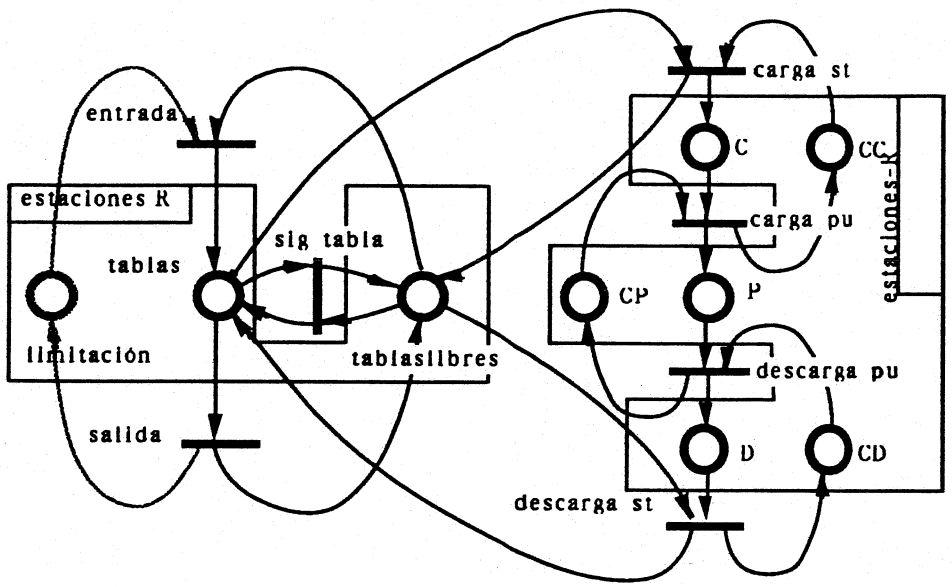

Figura 3.13: Red KRON para el sistema del ejemplo con limitación a 2 del número de piezas asignadas a cada estación.

3.11, conduce a la situación de conflicto mostrada en la figura 3.12a. A partir de la red es posible encontrar otros conflictos en el modelo entre:

1. los objetos de acción *entrada* y *sig-tabla* con respecto al objeto de marcado T_1 del lugar *tablaslibres*;
2. los objetos de marcado *carga-st* y *sig-tabla* con respecto a los objetos de marcado T_2 a T_6 de *tablaslibres*; y
3. los objetos de marcado S_i que deben formar la sustitución con respecto a la cual se disparará el objeto de acción *entrada*.

El mismo análisis muestra que las prestaciones del sistema bajan considerablemente si el número de piezas asignadas simultaneamente a una estación es superior a dos. Este hecho se debe a la generación de bloqueos temporales en el sistema de transporte, como el mostrado en la figura 3.12b, que bloquea la carga y descarga de otras estaciones. La decisión de limitar a dos el número de piezas que pueden asignarse a una estación en un instante dado, puede ser incorporada en el modelo introduciendo un lugar adicional que limita la entrada de piezas (figura 3.13), o puede ser considerada como heurística en el diseño del sistema de decisión. En el primer caso se configura un nuevo invariante dado por:

5. *(marcado estacion-asignada celda-estaciones-R C)*

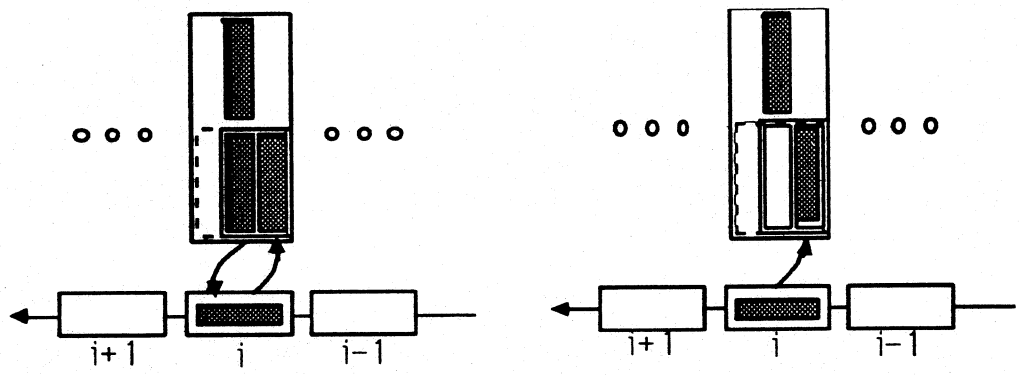

Figura 3.14: Conflicto por la tabla $j+1$.

+ *(marcado estacion-asignada celda-estaciones-R P)*
+ *(marcado estacion-asignada celda-estaciones-R D)*
+ *(marcado estacion-asignada transp-tablas-R tablas-libres)*
$= (S_1\ S_1\ S_2\ S_2\ S_3\ S_3\ S_4\ S_4\ S_5\ S_5\ S_6\ S_6)$

3.2.4 Problemas de decisión en el modelo

Como ya ha sido comentado anteriormente, hay ciertos aspectos que no están especificados en la red. Si se considera la red KRON de la figura 3.8 como ejemplo, algunos de estos aspectos pueden ser resueltos modificando el modelo como se ha mostrado anteriormente. Sin embargo, estas y otras decisiones pueden ser también tomadas a un nivel más alto en la jerarquía del sistema de control integrado del SFF.

Las indeterminaciones en una red KRON se manifiestan esencialmente en la forma de conflictos. La detección de estos mediante el análisis de red puede ser interesante en algunos casos, puesto que ello facilita la localización de los puntos de decisión donde asociar las políticas de control. Adicionalmente del conflicto puede extraerse información relativa al problema de decisión y los objetos de marcado implicados. En el modelo de la figura 3.8 surgen tres tipos de problemas de decisión, que corresponden a cada uno de los conflictos detectados en el análisis de la red:

1. Surge un problema de decisión relativo a la asignación de T_1 cuando es requerida simultáneamente por una pieza que pretende entrar en el sistema y por otra que desea abandonar la primera estación.

2. Se debe tomar otra decisión para asignar la tabla T_{j+1} (con $n > j \geq 1$)

cuando es requerida simultáneamente por una pieza que pretende avanzar por el sistema de transporte, y por otra que desea abandonar la estación *j+1-ésima* (figura 3.14).

3. Cada vez que entra una nueva pieza en el sistema, se debe decidir la estación en la que debe ser procesada. El número de piezas asignadas a la misma estación en cualquier instante de tiempo estará limitado a dos con objeto de limitar bloqueos y caidas de prestaciones, tal como se ha deducido del análisis del modelo.

3.3 Mecanismo de Control de la Red como Motor de Inferencia

Hasta ahora se ha abordado el problema de integración de la RdP centrandonos en los aspectos de representación, en base a la utilización de composiciones estructuradas de datos, y a la aplicación de una metodología de modelado. Para que este tipo de representación pueda ser utilizado como modelo de deducción en sistemas de IA, es importante hablar también de su adecuación inferencial.

La representación tipo frame admite la ejecución de algoritmos de inferencia clásicos en estos lenguajes como *búsqueda por intersección* [BRAC 79], *razonamiento por defecto* e *inferencia basada en espectativas* [MINS 75] o *inferencia por herencia* propia de lenguajes orientados a objetos.

Una aportación más específica desde la perspectiva de RAN consiste en la consideración de las transiciones de la red como reglas precondición/postcondición de sistemas de producción, como ha sido mostrado en varios trabajos tales como [BRU 86a], en el que las transiciones se implementan directamente a modo de reglas de producción de OPS5, o [VALE 87] donde se propone un mecanismo de interpretación de la red que funciona como motor de inferencias y puede trabajar con encadenamientos hacia atrás o hacia adelante. Los útimos trabajos de Valette [VALE 90] profundizan en esta idea de similitud entre redes de Petri y sistemas basados en reglas y proponen una adaptación del mecanismo de compilación de OPS5 con objeto de adecuarlo más específicamente al tratamiento integrado de redes de Petri. Por otra parte, en [MURO 89] se presenta KRON como un modelo de representación útil para abordar determinados problemas de planificación y búsqueda.

KRON dispone de un mecanismo de control específico basado en la interpretación de la red de alto nivel subyacente y comporta un nuevo mecanismo de inferencia. Una implementación del mecanismo de interpretación de una RAN es conocida como el "token player" [MUR 86b]. La aproximación utilizada en KRON sigue esta misma idea, pero hace uso de una técnica que sigue más de cerca la metodología desarrollada

para motores de inferencia en lenguajes de programación basados en reglas, y en concreto la desarrollada para el *lenguaje OPS5* [BROW 86]. El mecanismo de control de la red manejará los objetos de acción a modo de reglas, donde la precondición está constituida por las condiciones de habilitación implicadas por las relaciones de red previas y el predicado asociado al objeto de acción. El mecanismo de control forma parte de los trabajos desarrollados por J.L. Villarroel [VILL 90] como parte de su tesis doctoral. En el mencionado trabajo se adopta la perspectiva centrada en redes de Petri. A continuación se muestra un esquema del mecanismo de control desde una perspectiva más orientada a la IA y nos remitimos a [VILL 90] para un análisis detallado de conceptos más específicos sobre implementación.

3.3.1 Ciclo de reconocimiento-actuación

Como es tradicional en todo lenguaje de sistemas de producción, el procedimiento de control está constituido por un proceso en tres etapas de *reconocimiento*[5], *selección* y *ejecución* referido normalmente como ciclo de reconocimiento-actuación[6].

Durante el proceso de interpretación de la red pueden ocurrir situaciones de conflicto en las que existen varias posibilidades de disparo en que las precondiciones para la sensibilización estan interrelacionadas, es decir, la selección para el disparo de una posibilidad puede modificar las precondiciones que habilitan la sensibilización de otra. Para controlar efectivamente esta dependencia, KRON dispone de un nuevo tipo de objeto llamado **objeto conflicto** que agrupa los objetos de acción que pueden estar en conflicto (en situaciones donde no sea posible la aparición de conflictos el cometido del objeto conflicto se reduce a una labor trivial). Es posible conocer de antemano esta dependencia atendiendo exclusivamente a condiciones estructurales de la red. Así, el ciclo de reconocimiento-actuación estará dirigido por conflictos y su funcionamiento es el que se muestra a continuación:

- **Etapa de reconocimiento.** Examina los objetos conflicto para comprobar la sensibilización, para lo cual se aplica un *algoritmo de cálculo de sensibilización* a los objetos de acción componentes del conflicto. Su objetivo es encontrar aquellas ligaduras consistentes de las variables del objeto de acción que verifiquen el predicado asociado al objeto de acción. Estas ligaduras se calculan a partir de los objetos de marcado presentes en los atributos de estado, relacionados con el objeto de acción a través de una relación de red posterior.

- **Etapa de selección.** Si durante la etapa de reconocimiento se detecta alguna posibilidad de disparo, se aplican una serie de algoritmos y procedimientos de deducción (que serán el objeto de los siguientes capítulos), para encontrar, colectivamente, la resolución del conflicto que dará como resultado la selección

[5] "Matching" según su denominación inglesa.
[6] "Recognize-act cycle" según su denominación inglesa.

de la ligadura/s consistente/s a disparar. En la mayoría de los trabajos sobre RdP esta selección se reduce a la aplicación sistemática de una heurística simple, p.e. la política FIFO adoptada en [BRU 86b]. En [VALE 87] se adopta una solución más flexible en base a la asociación de reglas diferentes a las transiciones y en [MART 87] se propone para la solución de los conflictos el uso de técnicas de sitemas expertos que utilizen reglas de producción modeladas usando la información de la RAN. En el area de programación basada en reglas, las estrategias más reconocidas son LEX y MEA disponibles en OPS5 [BROW 86].

En la solución adoptada en KRON, cada objeto conflicto tiene asociada una **política de control**, especializada en resolver el conflicto. La ejecución de dicha política de control produce una llamada al sistema de toma de decisiones, del siguiente nivel en la jerarquía de control, que debe dar solución al conflicto. En esta llamada se envía la información de los objetos de marcado que han originado el conflicto y las estrategias (bases de conocimiento) útiles a seguir (pueden consistir en simples reglas de "dispatching" o en procedimientos que pueden implicar la ejecución de elaboradas estrategias de decisión).

- **Etapa de ejecución.** Una vez concluido el proceso de selección se procede a la ejecución del disparo para las ligaduras consistentes seleccionadas siguiendo las normas de disparo de objetos de acción, heredadas de la semántica de interpretación de las redes de Petri. La ejecución debe hacerse de forma que se permitan procesos concurrentes, especialmente para actuaciones sobre control directo [VILL 90].

3.4 Definición de las Redes KRON

3.4.1 Definición y terminología

Una red KRON ($RKRON$) es una 8-tupla dada por:

$$RKRON =< OE, SE, OA, OM, R, V, Rpost, Rprev >$$

donde:

- OE: es un conjunto finito de *objetos de estado*
- OA: es un conjunto finito de *objetos de acción*
- OM: es un conjunto finito de *objetos de marcado*

Los elementos de cada conjunto $n : t$ tienen un *nombre*, n, y están agrupados por *tipos jerárquicos*, t, definiendo un polimorfismo [CARD 85]. Se asume que todos los nombres son distintos. En la terminología de programación orientada a objeto los tipos se corresponden con las *clases* y los objetos con las *instancias*.

Objetos de estado

Un *objeto de estado*, $oe \in OE$, se define mediante su nombre, n_{oe}, su tipo, $t_{oe},, TOE = \bigcup_{oe \in OE} t_{oe}$, y un conjunto de atributos de estado $S_{oe},, SE = \bigcup_{oe \in OE \wedge s \in S_{oe}} s$.

Sean $i, j \in OE$, $t_i, t_j \in TOE$ sus tipos y S_i, S_j sus conjuntos de atributos de estado respectivos,

si $t_i = t_j$ entonces $S_i = S_j$

Objetos de marcado

Un *objeto de marcado* $m \in OM$ se define mediante su nombre, n_m, su tipo, $t_m,, TOM = \bigcup_{m \in OM} t_m$.

Relaciones

$\forall m \in OM$ puede establecer vínculos con otros objetos $o \in OM \cup OE$ a través de la definición de *relaciones* r (se pueden establecer relaciones entre objetos de marcado y otros objetos de marcado o de estado).

Sea $m \in OM$, R el conjunto de relaciones definidas sobre el tipo $t_m \in TOM$ y $r_i \in R$, entonces $r_i(m)$ representa el objeto relacionado con m mediante la relación r_i.

Sean $i, j \in OM$, $t_i, t_j \in TOM$ sus tipos y R_i, R_j los conjuntos de sus relaciones respectivas,

si $t_i = t_j$ entonces $R_i = R_j$

Datos asociados y marcado

Cada atributo de estado, $s \in SE$, tiene asociado un conjunto no-vacío de objetos de marcado, $C(s)$, definidos por su tipo, que representan las posibles ocurrencias de OM en s.

Un atributo de estado s podrá contener un número variable de objetos de marcado $m \in t_m$ con $t_m \in TOM$.

Se define el *marcado* de una red KRON como una función $M : SE \longrightarrow OM_{MC}$, tal que aplicada a un atributo de estado s retorna los objetos de marcado que contiene $M(s)$ y donde OM_{MC} representa el conjunto de todos los *multi-conjuntos finitos* sobre el conjunto OM.

Cada atributo de estado s tiene asimismo asociado un marcado inicial $M_0(s) \in$

$C(s)_{MC}$

Variables tipadas

Existe un conjunto V de *variables tipadas*. Cada variable tipada $v \in V$ tiene un nombre v y un tipo TV, siendo $TV \in OM \bigcup OE$.

Objetos de acción

Un objeto de acción $a \in OA$ se define mediante su nombre a, su tipo t_a, un conjunto de variables tipadas $V_a \subseteq V$ con $v_i : tv_i$ con $tv_i \in TV, , \forall v_i \in V_a$ y un predicado p_a que sólo contiene variables asociadas a dicho objeto de acción.

Relaciones de red

Los atributos de estado y objetos de acción pueden ser conectados de forma direccional mediante *relaciones de red*. Las relaciones de red establecidas desde un atributo de estado hacia un objeto de acción se denominan relaciones de red previas $Rprev : SE \longrightarrow OA$. Aquellas definidas en sentido contrario se denominan relaciones de red posteriores $Rpost : OA \longrightarrow SE$.

Las relaciones de red son un tipo de relaciones especiales que se definen como una terna (e, a, f) donde:

- $e \in SE$ es el atributo de estado de la relación y;
- $a \in OA$ es el objeto de acción de la relación;
- f es una *sustitución* sobre las variables $V_{a,f} = \{v_1, ..., v_n\}$ con $V_{a,f} \subseteq V_a$, que denotaremos por $f(m) = \{(r_1(m), v_1); ...; (r_n(m), v_n)\}$ donde:
  - v_i y $r_i(m)$ son del mismo tipo,
  - $m : t_e$ y t_e es el tipo asociado a e y $R = \{r_i\}$ el conjunto de relaciones definidas en t_e y
  - V_a el conjunto de variables asociadas a a

La aplicación de una sustitución, definida por una relación de red, a un objeto de marcado, da como resultado la *ligadura* de un subconjunto de variables del objeto de acción determinado por la relación de red.

Ligadura consistente

Una *ligadura consistente* de las variables de un objeto de acción a es aquella en la que $\forall v_i \in V_a$:

- v_i está ligada y

- la ligadura de v_i es la misma para cualquier sustitución $f \in Rprev(a)$, donde $Rprev(a)$ es el conjunto de sustituciones de todas las relaciones de red previas de a.

Sensibilización

Un objeto de acción $a \in OA$ se dice que está *sensibilizado* respecto de un marcado M si en cada atributo de estado, e, relacionado con a mediante una *Rprev*, existen los objetos de marcado, m_i, adecuados, de tal forma que el resultado de aplicar las sustituciones de *Rprev*, a dichos objetos de marcado, comporte una *ligadura* consistente que verifique el predicado p_a asociado a a.

Dado que, para un cierto marcado y objeto de acción, pueden existir varias ligaduras consistentes que verifiquen el predicado asociado, diremos que un objeto de acción está sensibilizado respecto a una o varias ligaduras.

Disparo

El *disparo* de un objeto de acción $a \in OA$, sensibilizado respecto de una cierta ligadura, consiste en retirar de los atributos de estado e_i, relacionados con a mediante una *Rprev*, los objetos de marcado que han producido dicha ligadura y depositar en los atributos de estado relacionados con a mediante las *Rpost*, los objetos de marcado que produzcan una ligadura consistente como la anterior, mediante las sustituciones definidas en las *Rpost*.

3.5 Características de KRON como lenguaje de representación del conocimiento

A modo de recapitulación, se podrían destacar las siguientes peculiaridades de KRON como lenguaje de representación del conocimiento para sistemas dinámicos discretos:

1. Integra consistentemente características de RAN, representación estructurada del conocimiento y sistemas de producción. Un mismo modelo puede ser interpretado y manipulado desde diferentes perspectivas. Así, un objeto de acción puede ser visto como una transición en una RAN, una estructura de datos para la especificación de una acción, un operador en un espacio de búsqueda o una regla en un sistema de producción.

2. Proporciona unas agradables características gráficas que, no sólo permiten mostrar la estructura jerárquica de las clases de objetos, sino también sus relaciones de conexión y sincronización.

3. En la representación de sistemas físicos, una aproximación orientada a objetos está más cerca del proceso cognitivo humano de percepción. Facilita la manipulación del conocimiento al tratar con objetos coherentes en lugar de con variables dispersas. Y, cuando los objetos se organizan apropiadamente en jerarquías, optimizan el almacenamiento y recuperación de datos.

4. La RAN subyacente en el modelo especifica el flujo de información y manteniene su consistencia, lo que elimina la ambigüedad en la interpretación presente en otros modelos.

5. La expresión de las características dinámicas mediante el modelo formal de RAN facilita la consistencia y precisión y permite utilizar técnicas de análisis de redes de Petri para comprobar ciertas propiedades cualitativas (bloqueos, alcanzabilidad, limitación, ...) o cuantitativas (tasas de utilización, probabilidad de disparo, ...) del modelo.

6. Puede soportar diversos mecanismos de inferencia: razonamiento por defecto, búsqueda por intersección, razonamiento por espectativas e inferencia basada en reglas.

7. Proporciona una metodología de modelado que facilita el diseño de los objetos y su interconexión mediante la sincronización de sus acciones. Esta característica resulta tanto más útil cuanto mayor es la complejidad y concurrencia del sistema a modelar.

8. Soporta la ejecución y simulación del modelo.

9. Un único modelo puede ser utilizado con diversos objetivos en la manipulación de sistemas dinámicos discretos como: coordinación, simulación, monitorización, recuperación de errores, scheduling y planificación.

3.6 Conclusiones

KRON facilita la representación de conceptos temporales y causales según una doble perspectiva relacional y estructural. Esta facilidad se debe a las características gráficas de la representación y a los desarrollos teóricos disponibles sobre propiedades estructurales de redes de Petri.

Se ha apuntado un método para extraer, de un modelo KRON definido en base a objetos y relaciones, una red de alto nivel, RAN. El análisis de la red mediante técnicas específicas, proporciona la posibilidad de encontrar ciertas propiedades del modelo como especificaciones funcionales relativas a vivacidad, cotas en los lugares o conflictos.

El mecanismo de control definido para la red permite la interpretación del comportamiento dinámico del sistema modelado. La extricta semántica de esta

interpretación (derivada de la semántica de la red de Petri subyacente en el modelo) dificulta la aparición de inconsistencias relativamente fáciles en otros modelos de representación utilizados en inteligencia artificial.

Por último, se aborda la presentación de los objetos y relaciones utilizados en KRON desde una perspectiva formal.

LEON Y CRONOPIO

Un cronopio que anda por el desierto se encuentra con un león, y tiene lugar el diálogo siguiente:

León. -Te como.

Cronopio (afligidísimo pero con dignidad). -Y bueno.

León. -Ah, eso no. Nada de mártires conmigo. Echate a llorar, o lucha, una de dos. Así no te puedo comer. Vamos, estoy esperando. ¿No dices nada?

El cronopio no dice nada, y el león está perplejo, hasta que le viene una idea.

Leòn.- Menos mal que tengo una espina en la mano izquierda que me fastidia mucho. Sácamela y te perdonaré.

El cronopio le saca la espina y el león se va, gruñendo de mala gana:

-Gracias, Androcles.

Historias de cronopios y famas
JULIO CORTAZAR

Capítulo 4

Representación a Nivel del Dominio: Sistemas de Fabricación

En el presente capítulo se va a abordar la aplicación de la herramienta de representación KRON en un dominio de aplicación específico: los **sistemas de fabricación**. En este nivel de representación se define la terminología y la semántica referida a sistemas de fabricación en términos de los conceptos, objetos, funciones y métodos definidos por los niveles de representación inferiores de KRON (implementación, lógico, epistemológico y conceptual).

Los objetos modelados en este nivel deben representar conceptos comunes en la planta de fabricación y deben poder ser tratados, con objeto de facilitar la creación y el mantenimiento de los modelos, en un lenguaje comprensible por el personal de planta. El sistema de representación incorpora primitivas que definen objetos prototipos de recursos, operaciones, planes de trabajo, órdenes de fabricación, etc. Estos pueden ser instanciados y sincronizados a múltiples niveles de abstracción. Pueden tener propiedades y disponen de una especificación de su comportamiento dinámico definida por una red KRON. Los objetos predefinidos están diseñados formando una jerarquía de especialización en la que también se hereda la dinámica especificada por la red. Esta jerarquía puede ser extendida o modificada según las necesidades de forma interactiva. La metodología de representación comporta una economía en el modelado y resulta especialmente útil para aplicaciones de prototipado rápido. Al mismo tiempo la representación pretende ser independiente de una planta concreta para posibilitar su aplicación a diferentes problemas.

Por otra parte, siempre es posible extraer la red de alto nivel que subyace en el modelo y analizar ciertas propiedades de comportamiento a partir, fundamentalmente, del análisis de invariantes.

En este trabajo estamos interesados en el modelado de entidades de fabricación

desde el punto de vista de su control de planta. su planificación y su simulación, por tanto la representación se enfocará desde estos puntos de vista. La consideración adicional de otras perspectivas de modelado (p.e. diseño del producto, diseño del proceso, contabilidad) conducirían obviamente a representaciones más extensas.

4.1 Modelado de Sistemas de Fabricación

Un aplicación "software" para el control de sistemas de fabricación integrada por computador (sistemas CIM[1] en terminología anglosajona) precisa de una herramienta de representación que capture el conocimiento requerido, relativo a la planta y sus restricciones de funcionamiento. La cantidad de información requerida para el funcionamiento del sistema se incrementa en gran medida al aumentar la variedad de productos que han de ser manipulados, la complejidad de las operaciones y de la maquinaria a utilizar, así como con la flexibilidad pretendida por el sistema, característica típica en sistemas de fabricación flexible. Esta gran variedad de entidades y restricciones indican la necesidad de un rico modelo para el dominio de sistemas de fabricación. Si se desea construir una arquitectura "inteligente" de control, el sistema debe tener acceso al conocimiento detallado sobre las características del dominio de aplicación, incluyendo operaciones, rutas de proceso, máquinas, áreas de trabajo, herramientas, materiales, personal, órdenes, etc..

En el campo de la Inteligencia Artificial se han desarrollado diversas aproximaciones para el modelado de sistemas de fabricación [STEF 86]. La aproximáción más mencionada en la literatura se basa en la utilización de sistemas basados en reglas, donde se considera el conocimiento del experto como factor clave [HALE 83,DESC 84,ACOC 86,BENS 86,BRUN 86,LAWR 86]. Otra aproximación sigue la perspectiva de sistemas basados en el conocimiento donde la representación juega un papel más importante. Generalmente se han utilizado técnicas estructuradas de representación basadas en frames [FOX 83,LEPA 85, SMIT 86].

El modelado y análisis de sistemas de fabricación han constituido una aplicación de especial interés en el campo de las redes de Petri [MART 86,ALJA 87,SILV 89]. Estos trabajos abordan fundamentalmente la construcción de modelos de sistemas de fabricación que verifiquen ciertas propiedades de tipo cualitativo, como ausencia de bloqueos, limitación de capacidades, exclusión mútua en el uso de máquinas, etc.

Inicialmente se utilizaron modelos con redes de Petri de bajo nivel (quizás por su mayor facilidad de tratamiento y la mayor disponibilidad de resultados teóricos) tanto en aproximaciones de modelado ascendente [DUBO 83,NARA 85], como descendente [VALE 82]. Existen trabajos que han utilizado restricciones

[1] "Computer Integrated Manufacturing".

o extensiones de estas redes respecto a su definición tradicional: redes de Petri modificadas [BECK 86], redes de Petri de libre elección [KROG 87]. Cuando el objetivo del modelo es la evaluación de prestaciones o planificación, se utilizan redes de Petri con una interpretación temporizada [KAMA 86] o estocástica.

Para tratar con modelos más complejos se han utilizado formalismos más abstractos en base a redes de alto nivel, donde la composición modular resulta ser la metodología de diseño imperante. [VALE 82] hacen uso extensivo de refinamientos en sus modelos de redes Predicado/Transición y [ALLA 82,MART 85,MART 86, MART 87] utilizan redes de Petri coloreadas y compactan los modelos a través de la integración de información en los colores. En los trabajos de Castelain y col. [CAST 89], basados en la creación previa de una descripción funcional del modelo materializada en su *pregrafo*, y en [GENT 87] que utilizan redes de Petri coloreadas adaptativas y estructuradas.

Se han desarrollado también aproximaciones que utilizan abstracciones más orientadas al dominio en el sentido de utilizar lenguajes con primitivas más cercanas a la jerga propia de sistemas de fabricación. Generalmente estas aproximaciones utilizan interfases gráficas con iconos especiales para las distintas entidades. Dentro de este grupo se encuentra el sistema GRAMAN [VILL 88]. GRAMAN utiliza una descripción separada de planes de trabajo y recursos disponibles en la planta que, mediante una metodología de interconexión, produce un modelo (transparente al usuario), que puede ser utilizado posteriormente para evaluación de prestaciones, control o planificación.

También se han desarrollado soluciones conjuntas que hacen uso de ambos tipos de técnicas. La opción más generalizada ha sido la combinación de redes de Petri de alto nivel y programación basada en reglas [ATAB 87,FLEI 89]

4.1.1 Modelado de Sistemas de Fabricación con KRON

La peculiaridad de KRON de integrar RdP y técnicas de IA permite abordar la representación de sistemas de fabricación utilizando las ventajas que proporcionan ambas perspectivas. En la metodología que se propone, el sistema de fabricación se define en base a un conjunto de bloques constructivos, constituidos por redes KRON, y una fase posterior de interconexión. Cada uno de estos bloques modelará un componente del sistema de producción. Dependiendo del nivel de abstracción utilizado, estos componentes pueden modelar una entidad individual (máquina, robot, almacén, etc.) o un subsistema (celda de trabajo, sector, etc.).

Cada bloque constructivo, modelado a cierto nivel de abstracción, puede refinarse haciendo uso de elementos constructivos del nivel de representación inferior. La interconexión entre bloques se realiza según dos direcciones:

1. en sentido *horizontal* para conectar entidades al mismo nivel de abstracción y

2. en sentido *vertical* conectando entidades modeladas a diferentes niveles de abstracción.

Esta descripción del sistema de producción a varios niveles de abstracción permite utilizar metodologías de diseño ascendente o descendente indistintamente.

Las *primitivas de representación* correspondientes al dominio han sido divididas en tres categorias de entidades en atención a la homogeneidad de los conceptos involucrados:

- **Recursos:** Describen elementos o agrupaciones físicas y lógicas de entidades de la planta, así como características importantes acerca de su naturaleza física (p.e. capacidad, restricciones, productividad) y características operativas (p.e. requerimientos, disponibilidad, estado, acciones, etc.).

- **Materiales, productos y operaciones:** Proporcionan información sobre los materiales, productos, operaciones, por quien han sido requeridos y datos acerca de su importancia y restricciones. Las operaciones son conceptos básicos en la tarea de control y planificación, y describen los requerimientos específicos de cada actividad

- **Planes de proceso:** Definen los procesos necesarios para generar los productos por medio de las posibles conectividades entre las operaciones.

En los siguientes apartados se introducirán los objetos y atributos especiales, relativos al dominio de aplicación. Paralelamente, se irá presentando la metodología de modelado propuesta, cuyo esquema general se muestra en la figura 4.1. Recursos y planes de trabajo se modelarán con redes KRON que describirán el sistema de producción desde dos perspectivas (física y operacional), mientras que operaciones y materiales constituirán los objetos de marcado que fluyen por la red. Mediante la sincronización de los objetos representativos de los recursos se constituye la estructura de la planta de producción. Una sincronización especial entre estructura de planta y planes de proceso forma la estructura del sistema de producción que, integrando los objetos de marcado, formará el modelo del sistema de producción completo.

4.2 Modelado de los Recursos de Fabricación

Dentro del contexto de sistemas de fabricación, un **recurso de fabricación** es una entidad que realiza alguna operación conducente a producir el producto final. Están incluidos dentro de este concepto todas las máquinas, almacenes, herramientas, operadores, etc. que son necesarios para fabricar un producto.

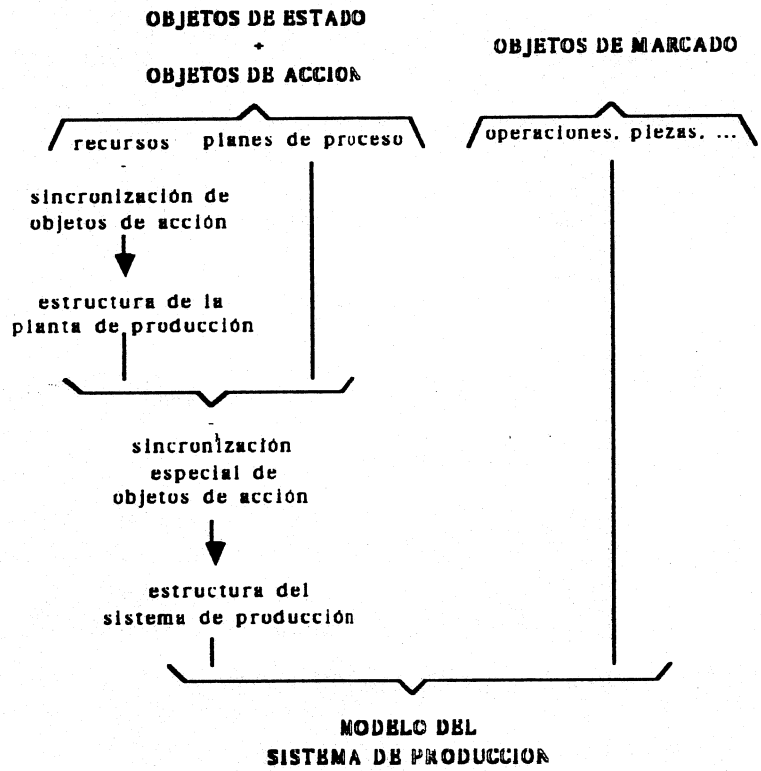

Figura 4.1: Esquema de la metodología de modelado.

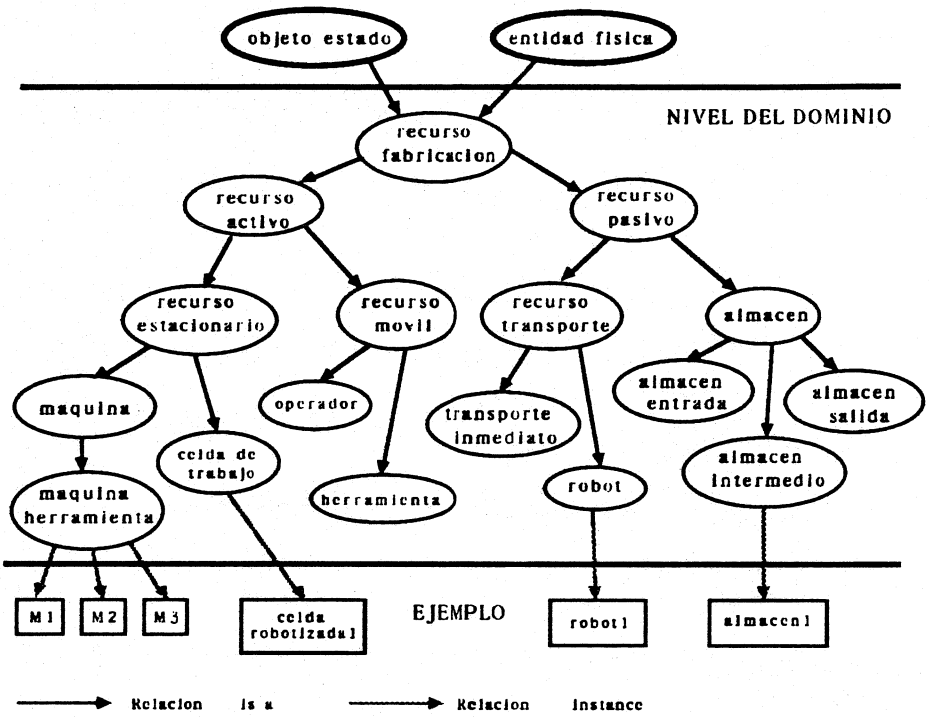

Figura 4.2: Ejemplo de jerarquía en la representación de recursos.

Para el modelado completo de un recurso, es necesaria la información de sus características físicas así como la descripción de su comportamiento dinámico (en el marco de aplicaciones de simulación y planificación, es imprescindible también, la posibilidad de recopilar información histórica o predictiva). En general, los recursos de fabricación poseen un comportamiento dinámico que puede ser modelado usando una red KRON. Por tanto, un recurso estará descrito por *un objeto de estado*, que recoja la información referente a sus características y propiedades, incluido su estado, y *distintos objetos de acción* que modelarán sus posibles acciones.

El modelado de recursos se realiza en un marco en el que se considera una especificación jerárquica de los procesos de fabricación, motivado por un deseo de permitir el diseño del control y el razonamiento sobre asignación de recursos a diferentes niveles de precisión.

La figura 4.2 muestra un ejemplo de jerarquía de especialización de recursos. En esta jerarquía, los diferentes objetos heredan las características de los otros objetos que se encuentran a un nivel superior en el árbol de especialización, mecanismo típico de las jerarquías de clases en los lenguajes orientados a objeto. KRON soporta además la herencia de las características que definen el comportamiento dinámico de los recursos, de forma que un objeto pueda heredar la RAN que subyace en la definición de los objetos superiores en la jerarquía.

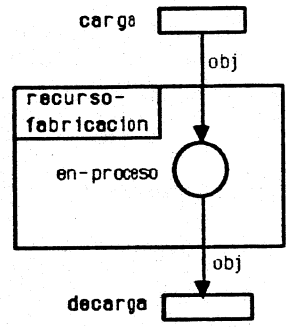

Figura 4.3: Redes KRON que representa el objeto *recurso*.

{*recurso-fabricacion*
is-a : objeto-estado entidad-fisica
en-proceso :
type : atributo-estado
dato-asociado : producto
pre-relaciones-red-con : &descarga
post-relaciones-red-con : &carga
acciones : carga descarga
carga : &carga
descarga : &descarga
nivel-precision :
range : (TYPE instance niveles-de-precision)
descripcion : }

{*carga*
is-a : objeto-accion
post-relaciones-red : (&recurso-fabricacion en-proceso Obj)
dato-asociado : < *vop* >
predicado : t }

{*descarga*
is-a : objeto-accion
pre-relaciones-red : (&recurso-fabricacion en-proceso Obj)
dato-asociado : < *vop* >
predicado : t }

donde: Obj ≡ {**schema-name** < *vop* > }

Figura 4.4: Objetos representantes de la entidad *recurso-fabricacion*.

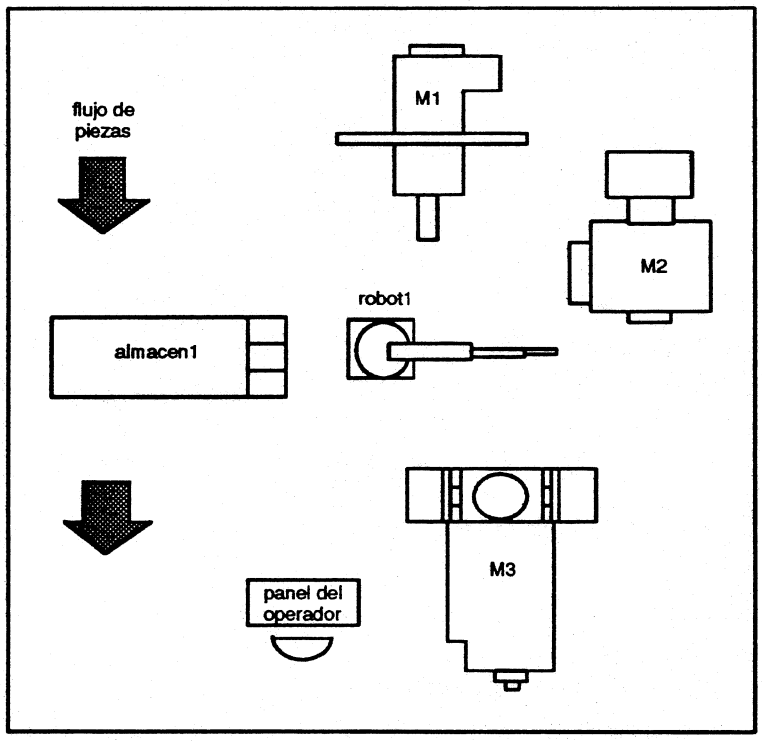

Figura 4.5: Ejemplo de celda robotizada.

Para construir esta jerarquía de recursos, comencemos representando el concepto de recurso al más alto nivel de abstracción, a partir del cual se irán definiendo subsiguientes objetos que representen otras entidades recurso más especializadas. La entidad **recurso de fabricación** representa un concepto muy general que recoge cualidades de entidades muy diversas. El objeto *recurso-fabricacion* debe recoger aquellos atributos que son comunes a todas ellas. En general, se podría considerar que todos los recursos pueden utilizarse para realizar alguna operación con un comienzo y un final de ejecución. Este comportamiento dinámico queda representado mediante la red de la figura 4.3, con un atributo de estado *en-proceso* y dos objetos de acción para la *carga* y *descarga* del recurso. En la figura 4.4 se muestran sus "schemas". Los atributos *carga* y *descarga* del objeto de estado *recurso-fabricacion*, referencian los prototipos de los objetos de acción (indicados con el símbolo "&")[2].

A partir del objeto genérico *recurso-fabricacion*, se pueden definir recursos más específicos hasta componer el árbol de la figura 4.2. En los siguientes apartados se irán desarrollando algunos aspectos de esta jerarquía que se ilustrarán por medio del ejemplo mostrado en la figura 4.5. El dibujo muestra esquemáticamente una celda de fabricación, que denotaremos por *celda-robotizada1*, compuesta por tres

[2]La necesidad de estos atributos, que permiten enlazar indirectamente con los prototipos de las acciones, se debe a razones de implementación del mecanismo de herencia e instanciación de la red.

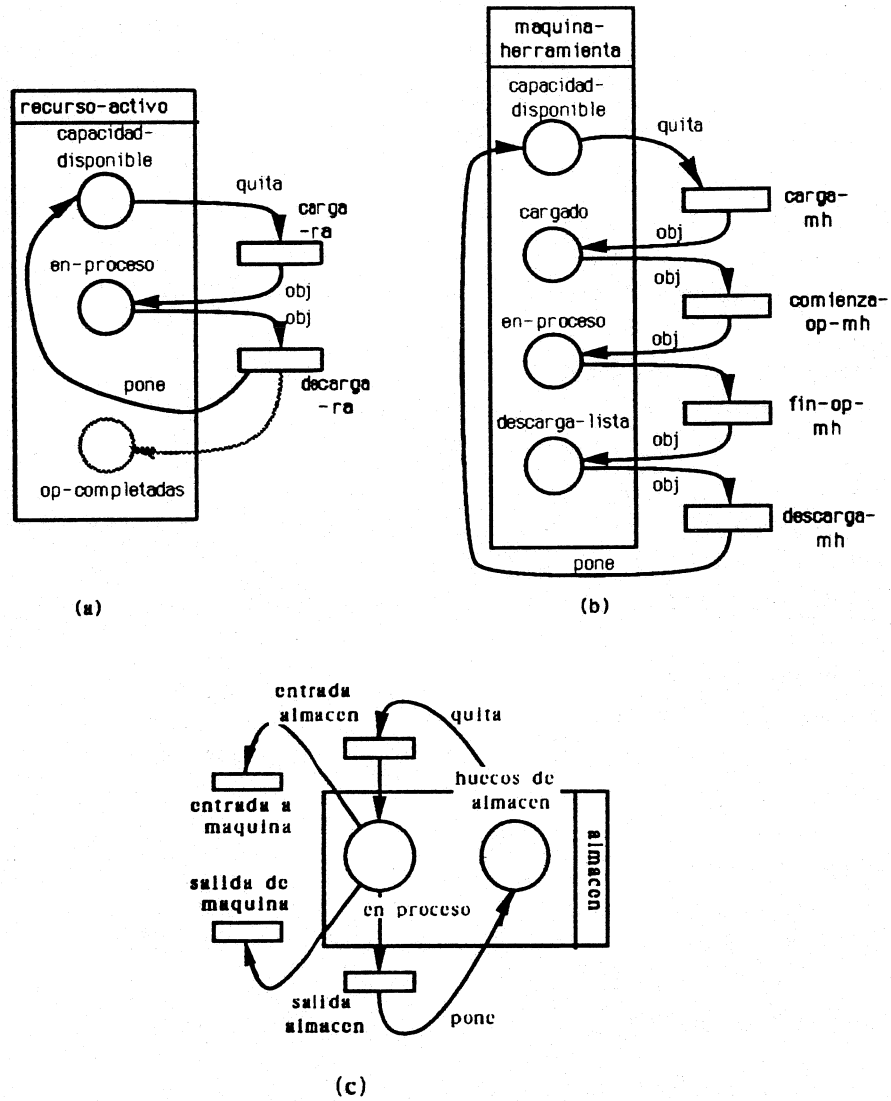

Figura 4.6: Prototipos de *recurso-activo*, *maquina-herramienta* y *almacen*.

máquinas *M1*, *M2* y *M3*, un almacén de entrada/salida de acceso aleatorio *almacen1*, y un robot, *robot1*, que realiza el movimiento de material en dicha celda (carga y descarga de cualquier recurso).

4.2.1 Recursos estacionarios

Los **recursos estacionarios** son aquellos recursos de fabricación que tienen una localización fija en el sistema de producción. A nivel de detalle, incluyen las máquinas individuales y estaciones de trabajo. A nivel abstracto, son áreas funcionales de trabajo compuestas por conjuntos de máquinas y/o estaciones de trabajo.

{*recurso-activo*
is-a : recurso-estacionario
en-proceso :
capacidad-disponible :
op-completadas :
type : atributo-registro-estado
carga : &carga-ra
descarga : &descarga-ra
op-pendientes :
tiene-recursos :
recurso-de :
capacidad-nominal :
capacidad-disponible-prevista :
range : (SET (TYPE instance *intervalo-capacidad*))
espec-averia :
range : (TYPE instance *espec-averia*)
estadisticas :
range : (TYPE instance *informe-estad-recurso*)
nivel-precision :
range : (TYPE instance *niveles-de-precision*)
puesta-en-marcha :
range : (TYPE instance *puesta-en-marcha-restr*)
turnos-trabajo :
range : (TYPE instance *turnos-trabajo-restr*) }

{*carga-ra*
is-a : carga
pre-relaciones-red :
(&recurso-activo capacidad-disponible Obj)
post-relaciones-red :
(&recurso-activo en-proceso Obj)
dato-asociado : < *vobj* > }

Figura 4.7: Objetos *recurso-activo* y *carga-ra*.

La definición de *recurso-activo* (figura 4.6) incluye la red subyacente en el modelo de su anterior en la jerarquía, a la que añade una nueva restricción dinámica identificada por un atributo de estado (*capacidad-disponible*) que introduce la limitación de capacidad. En la figura 4.7 se representa el objeto representativo del *recurso-activo*. En este objeto destacan también otra serie de atributos característicos que lo muestran como un objeto más especializado con respecto a sus predecesores:

- Los prototipos de objetos de acción son diferentes (*carga-ar* y *descarga-ar*), al haber cambiado también las relaciones de red respecto al predecesor.
- El atributo *estadisticas* recoge la información histórica relativa a la utilización del recurso. El atributo *capacidad-disponible-prevista* incluye información acerca de previsiones futuras (por ejemplo en forma de reservas del recurso realizadas por las órdenes), evaluada por el sistema de decisión.
- Las relaciones *nivel-precision*, *recurso-de* y *tiene-recursos* proporcionan la perspectiva jerárquica de definición del modelo de la planta, e indican el nivel de precisión del recurso en la jerarquía de abstracción y su relación con el recurso superior e inferiores en dicha jerarquía.
- Se asocian restricciones relativas a limitaciones de tiempo implicadas por los turnos de trabajo (atributo *turnos-trabajo*), y restricciones debidas a intervalos de puesta en marcha (atributo *puesta-en-marcha*).
- El atributo *op-pendientes* sirve al sistema de decisión para almacenar información precalculada sobre asignación de operaciones al recurso.
- *op-completadas* es un atributo de estado especial (véase figura 4.6), que es utilizado para recopilar información sobre datos del proceso. Sin embargo, por construcción, no afecta para nada al resto de la red, por lo que no se considera en la lista de estados. A este tipo de atributo de estado de la red se les denomina **atributos de registro de estado** (en adelante no será detallado como atributo de estado conectado a la red).

Mediante un proceso de especialización análogo se construye la jerarquía completa de definición de recursos estacionarios. Por ejemplo, se puede definir el objeto prototipo *maquina-herramienta* refinando el atributo *en-proceso* con dos estados adicionales representados como atributos de estado:

- *cargado* para indicar que una pieza ha sido cargada pero no ha comenzado el proceso y
- *descarga-lista* para representar la finalización del proceso y que la pieza está lista para ser descargada.

{*M1*
instance : maquina-herramienta
cargado :
en-proceso :
descarga-lista :
capacidad-disponible :
recurso-de : celda-robotizada1
nivel-precision : nivel-precision-maquina
capacidad-nominal : 1 }

{*carga-M1*
instance : carga-mh
pre-relaciones-red :
(M1 capacidad-disponible quita)
post-relaciones-red :
(M1 cargado Obj) }

{*descarga-M1*
instance : descarga-mh
pre-relaciones-red :
(M1 descarga-lista Obj)
post-relaciones-red :
(M1 capacidad-disponible pone) }

Figura 4.8: Objetos *M1*, *carga-M1* y *descarga-M1*.

En este proceso de especialización de las redes KRON, es necesario crear nuevos objetos (de estado y de acción) conectados mediante relaciones *is-a* con sus superiores. Adicionalmente, es necesario en algunos casos eliminar o crear específicamente nuevas relaciones de red (en el presente caso los correspondientes a los objetos de acción *carga-mh* y *descarga-mh*).

El usuario puede expandir la jerarquía de clases de recursos definiendo nuevos prototipos más especializados. Una vez definidos los prototipos, el modelo del sistema se construye creando instancias de dichos objetos. Así, las máquinas **M1, M2** y **M3**, utilizadas en el ejemplo, se construyen creando instancias del prototipo *maquina-herramienta*, y particularizandolas con sus respectivas características. El prototipo está representado por el objeto de estado *maquina-herramienta*, su instanciación compuesta crea el objeto de estado *M1* y los objetos de acción *carga-M1, comienza-op-M1, fin-op-M1* y *descarga-M1*, algunos de estos objetos se muestran en la figura 4.8 y su representación gráfica en la figura 4.9a.

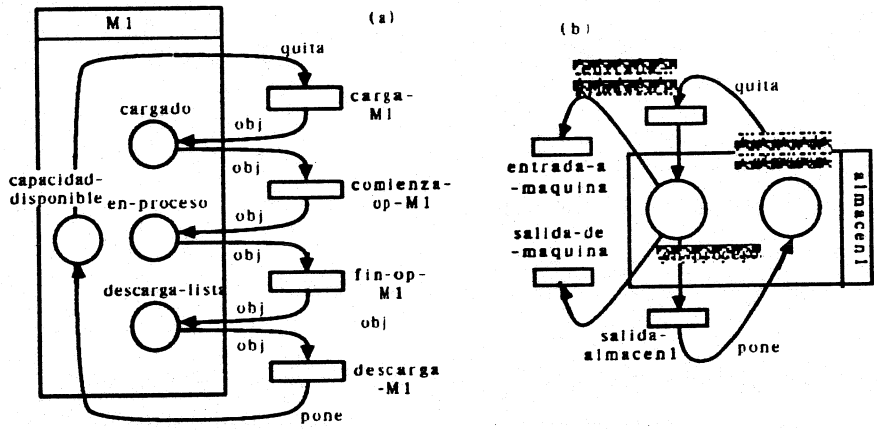

Figura 4.9: Red KRON reprepresentativa de los objetos *M1* y *almacen1* del ejemplo.

Los almacenes y estaciones de paletizado son considerados **recursos pasivos.** Modelos de almacén con acceso aleatorio son fácilmente definibles con un único atributo de estado y las transiciones apropiadas para entrada y salida. La figura 4.9b muestra un ejemplo de red para un almacén de múltiple entrada/salida, capacidad limitada y acceso aleatorio.

4.2.2 Recursos móviles

Los recursos móviles no tienen una localización fija en el sistema de fabricación y se trasladan (o son trasladados) entre distintas localizaciones a lo largo del tiempo. Dentro de este tipo cabe encuadrar herramientas, recursos humanos, palets, etc.

Los recursos móviles presentan algunas diferencias respecto a los anteriores porque no están restringidos a un atributo de estado concreto. Generalmente no representan los recursos más restrictivos para realizar operaciones (**recursos secundarios**), por lo que normalmente se combinan con otros recursos primarios asumiendo únicamente un papel habilitador. En su modelo más simple, pueden representarse mediante un atributo de estado con arcos habilitadores a los diversos **recursos primarios.** Cuando estos recursos son limitados (no existen suficientes operadores o herramientas, o no todos están entrenados para cualquier máquina) se añade un atributo de estado adicional para contemplar la capacidad disponible. Pueden modelarse también configuraciones más complicadas, por ejemplo para representar almacenes de herramientas incluyendo o no la política de servicio del almacén.

{*recurso-transporte*
is-a : recurso
capacidad :
en-proceso :
carga : &carga-transporte
descarga : &descarga-transporte
tiene-localizacion :
recurso-transporte-de : }

Figura 4.10: Descripción del objeto *recurso-transporte*.

4.2.3 Recursos de transporte

Los recursos de transporte (*dispositivo-transporte*) se refieren a los dispositivos mecánicos que se utilizan para el transporte de unidades de producción de una localización a otra. Un *dispositivo-transporte* solo podrá ser requerido en operaciones de transporte hacia recursos primarios.

Ejemplos de este tipo de recursos son carretillas, elevadores, vehículos de transporte, filoguiados, robots, etc. Las características generales de los recursos de transporte están recogidas en el objeto *recurso-transporte*, figura 4.10 (representación gráfica en la figura 4.11a):

- *En-proceso* es un atributo de estado heredado de *recurso*. *Capacidad* es utilizado para limitar la disponibilidad del transporte.
- Las acciones *carga-transporte* y *descarga-transporte* son especializaciones de *carga* y *descarga* para el transporte.
- *Tiene-localizacion* indica la localización del recurso de transporte y *recurso-transporte-de* es una relación jerárquica que conecta el objeto con su nivel de abstracción superior.

En la celda de la figura 4.5, el robot se comporta como un recurso de transporte. Si consideramos que el tiempo necesitado por el robot para realizar un movimiento es despreciable con respecto a los tiempos del resto de operaciones, entonces cabe aplicar un mayor grado de abstracción. Así, se puede representar el comportamiento del robot como un caso restringido de un recurso de transporte, con una única acción a considerar, cuya función consiste exclusivamente en limitar la sensibilización de otras acciones que necesiten la participación del robot. La figura 4.11b muestra la red KRON del robot del ejemplo.

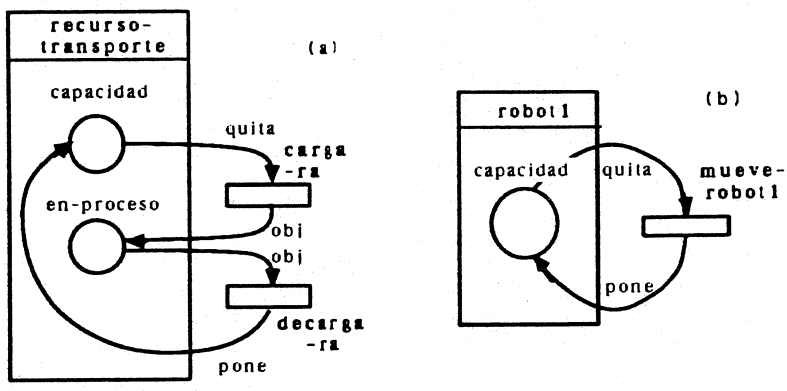

Figura 4.11: Red KRON reprepresentativa de los objetos *recurso-transporte* y *robot1* del ejemplo.

4.2.4 Interconexión de recursos en el modelado del sistema

La descripción estructural de una instalación de producción se completa estableciendo las interfases entre recursos, lo que hará posible la comunicación de un flujo de objetos de marcado entre ellos.

En el caso presentado en el ejemplo, el robot coge/deja piezas de/en el almacén y las deja/coje en/de las máquinas. Para modelar este hecho, será necesario establecer una sincronización múltiple entre la acción *mover* del robot y las acciones *entrada-a-maquina* y *salida-de-maquina* del almacén. En el modelado de la llegada/salida de piezas a las máquinas, es necesario establecer otra sincronización múltiple entre cada una de las acciones resultantes de la sincronización múltiple anterior, con las respectivas acciones de entrada/salida de las máquinas. La figura 4.12a muestra el modelo completo resultado de dichas sincronizaciones.

El método de modelado aplicado permite conservar las características de comportamiento desde la prespectiva particular de cada recurso. El conjunto de relaciones causales y de habilitación del modelo final se han construido de forma incremental a partir de las relaciones de los objetos componentes.

Si se considera ahora un nivel superior de abstracción, se puede pensar en una red KRON que represente el comportamiento de la celda (*celda-robotizada1*) de forma abstracta. En el nivel de precisión de celda, *celda-robotizada1* está compuesta por un atributo de estado que representa el estado de piezas en proceso y dos acciones para la entrada y salida de piezas. También pueden especificarse limitaciones de capacidad añadiendo un atributo de estado adicional para la capacidad disponible, como se muestra en la figura 4.12b. En este caso, es necesario otro tipo de sincronización (ahora en sentido vertical) que acomode el flujo de información entre ambos niveles de precisión. Con este objeto se define un nuevo tipo de relación denominada **relación de sincronización internivel.**

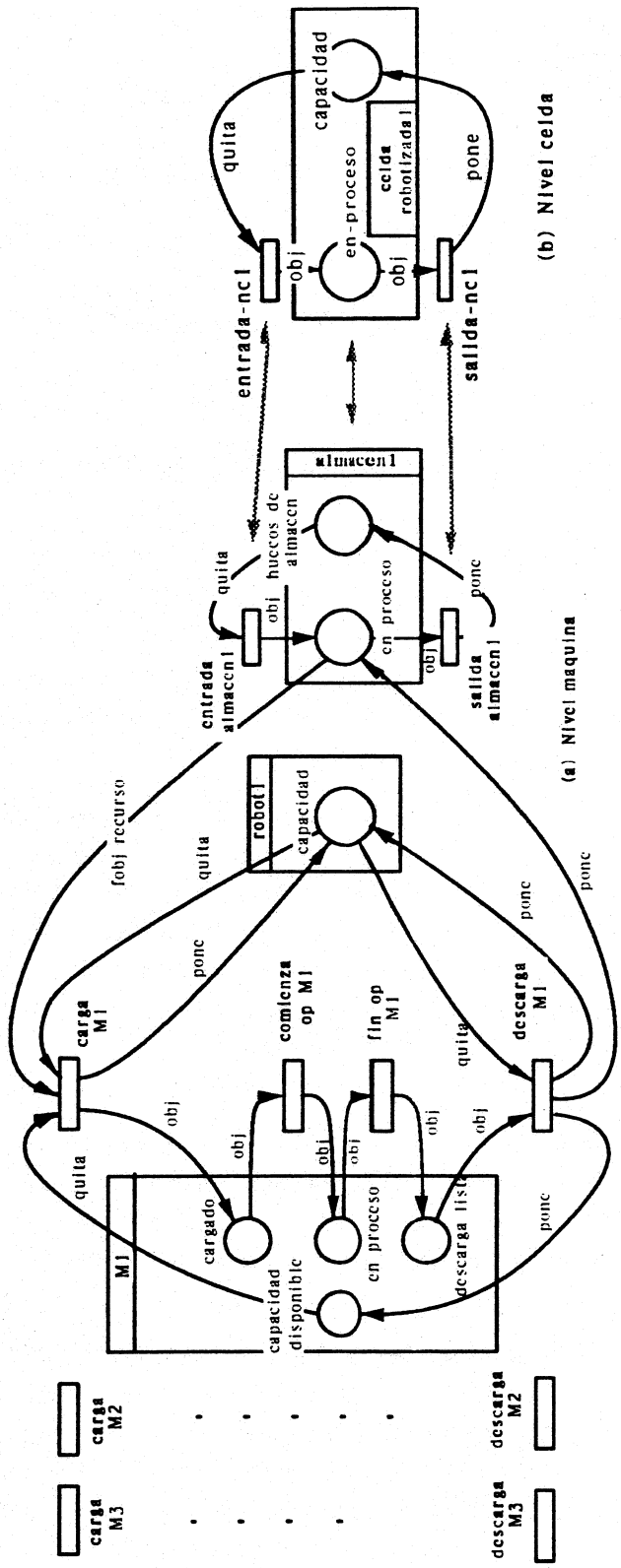

Figura 4.12: Red KRON que representa la *celda-robotizada1* del ejemplo.

{*celda-robotizada1*
instance : celda-de-trabajo
en-proceso :
capacidad : 3
carga : entrada-nc1
descarga : salida-nc1
tiene-recursos : almacen1 robot1 M1 M2 M3
recurso-de :
nivel-precision : nivel-precision-celda
descripcion : }

{*entrada-nc1*
is-a : entrada-celda
pre-relaciones-red :
(celda-robotizada1 capacidad quita)
post-relaciones-red :
(celda-robotizada1 en-proceso Obj)
sincronizacion-nivel-inferior : entrada-almacen1
dato-asociado : < *vobj* >
predicado : t }

Figura 4.13: Objetos a nivel de precisión de celda.

{*producto*
is-a : objeto-marcado
prototipo-plan-proceso :
rango : (TYPE is-a *plan-proceso*)
plan-proceso :
estado :
rango : (OR no-planeado planeado en-proceso completado)
satisface-demanda :
rango : (TYPE instance *demanda*)
tiempo-medio-proceso :
rango : (TYPE instance *espec-duracion*)
intervalo-produccion-planeado :
rango : (TYPE instance *intervalo-tiempo-calendario*)
descripcion : }

Figura 4.14: Descripción del objeto *producto*.

La figura 4.12 muestra gráficamente las relaciones entre ambos niveles de precisión y en la figura 4.13 se ilustran algunos objetos representativos al nivel de celda. Cabe destacar que la relación *tiene-recursos* (su inversa es *recurso-de*) referencia los objetos de estado al nivel de precisión de máquina. Esta relación tiene el sentido de una relación de *agregación* según se define en [SATH 85].

4.3 Modelado de Materiales, Productos y Operaciones

Los materiales y productos constituyen las entidades que recorren las redes que modelan los recursos, mientras que las operaciones recorren las redes que modelan los planes de trabajo. Así pues, parece lógico su modelado mediante la especialización del *objeto de marcado*.

Los materiales son objetos utilizados y consumidos durante el proceso de fabricación, en general se comportarán a modo de precondiciones o habilitaciones para el disparo de acciones. Los productos (figura 4.14), en cambio, precisan acarrear mayor cantidad de información relativa a la demanda que satisface (información que relacione el producto a fabricar con el cliente así como especificaciones tales como pedido completo, fecha debida, prioridad, etc.), el plan de proceso que se precisa para fabricar el producto, estado de fabricación y otras características fundamentalmente temporales.

{*operacion-fabricacion*
is-a : operacion
recurso-primario :
capacidad-requerida :
recursos-secundarios :
siguiente-operacion :
previa-operacion :
producto :
estado :
rango : (OR no-planeada planeada en-proceso completada)
tiene-duracion :
rango : (TYPE instance *espec-duracion*)
estadisticas :
rango : (TYPE instance *informe-estad-operacion*)
descripcion : }

Figura 4.15: Descripción funcional del objeto *operacion-fabricacion.*

4.3.1 Descripción de las operaciones de fabricación

Las operaciones constituyen las actividades que pueden ser realizadas en el sistema de fabricación. Estas incluyen la especificación de las distintas operaciones de fabricación (así como otras no directamente implicadas en la producción, como operaciones de mantenimiento) y su composición para formar los planes de proceso.

Las **operaciones** son unidades de trabajo realizados sobre algún material como parte de la creación de un producto final y pueden existir en varios niveles de abstracción o detalle. Las operaciones especifican restricciones sobre cada paso del proceso de producción (p.e. requerimientos de recursos, materiales, herramientas, condiciones de proceso, etc.).

Los prototipos de las operaciones se definen de forma incremental construyendo un árbol jerárquico de manera similar a los recursos, en este caso teniendo como progenitor el *objeto de marcado.* La figura 4.15 muestra un ejemplo de prototipo para el modelado de operaciones de fabricación. En el objeto *operacion-fabricacion* pueden distinguirse varios atributos y relaciones importantes:

- La relación *is-a* constituye el mecanismo de herencia a través de la jerarquía de especialización.
- Los atributos *recurso-primario, recursos-secundarios* y *capacidad-requerida* indican los requerimientos de recursos primarios y secundarios, así como la

{*espec-duracion*
is-a : objeto-conceptual
calcula-duracion-esperada :
calcula-duracion-puesta-en-marcha-esperada :
calcula-duracion-ejecucion-esperada :
calcula-duracion-actual :
espec-duracion-de :
distribucion-probabilidad :
range : (TYPE instance objeto-distribucion)
media :
desviacion-estandard :
...}

Figura 4.16: Objeto para la especificación de duración de operaciones de fabricación.

capacidad de recurso requerida.

- La traza de fabricación de un producto está formada por el encadenamiento, extrictamente secuencial, de las diversas operaciones concretas que han sido ejecutadas sobre dicho producto. Las relaciones *siguiente-operacion* y *previa-operacion* definen conexiones causales de la instancia de la operación con sus adyacentes.

- El estado de la operación desde el punto de vista del sistema de decisión (*no-planeada, planeada, en-proceso* ó *completada*) se almacena en el atributo *estado*.

- El atributo *tiene-duración* recoge la especificación de restricciones de tiempo de ejecución de las operaciones. El objeto de especificación de la duración, *espec-duracion* (figura 4.16), tiene por objeto encapsular todos los parámetros (media, desviación estandard, distribución de probabilidad de la duración) y métodos relevantes para la determinación de la duración de la operación (tiempo de puesta en marcha, tiempo de ejecución, tiempo total esperado y tiempo real).

- El atributo *estadisticas* es utilizado para la recolección de datos históricos relacionados con la operación.

4.4 Modelado de los Planes de Proceso

Los **planes de proceso** representan las distintas posibilidades de conexión de las operaciones y constituyen por tanto, procesos de fabricación genéricos. El plan de proceso indica la estructura causal de todas las posibles operaciones útiles para la creación de un producto (final o intermedio). Según esta definición, el plan de proceso representa una descripción prototípica de todas las secuencias posibles de operaciones. La obtención de un producto determinado, implica el seguimiento de una secuencia concreta de operaciones del plan de proceso.

En el presente trabajo se propone un esquema especial de representación, para operaciones y planes de proceso, que está influenciado por el ámbito de aplicaciones que ha de soportar el modelo (control, simulación, planificación, ...).

En cada nivel de abstracción, el plan de proceso estará representado por un objeto de estado y diversos objetos de acción que marcarán el comienzo y finalización de todas sus posibles operaciones. Las relaciones de red constituyen el mecanismo para definir las posibles conexiones a nivel prototípico, así se podrán formar composiciones secuenciales, alternativas, ensamblajes y desensamblajes, etc. Las operaciones serán objetos de marcado enlazados por relaciones causales del tipo *siguiente-operación* o *previa-operación* (véase la sección 3.1.2 y las referencias [FOX 83,SATH 85]). De esta forma los enlaces causales entre los objetos representantes de las operaciones indicarán la traza concreta que ha seguido cada producto por la red prototípica de su plan de proceso.

Para clarificar la exposición de estos nuevos conceptos, continuaremos con el ejemplo introducido en el punto 4.2. Supongase que el sistema *celda-robotizada1* puede fabricar dos diferentes tipos de productos: *P1* y *P2*. La secuencia de operaciones para llevar a cabo procesos del tipo P1 es *OP1* , *OP2*. La operación *OP1* puede llevarse a cabo indistintamente en las máquinas *M1* o *M3*, mientras que la operación *OP2* solo puede ser ejecutada por la máquina *M2*. El proceso *P2* requiere la secuencia de operaciones *OP3, OP4*, que son llevadas a cabo por las máquinas *M2* y *M3*, respectivamente.

La figura 4.17a muestra el plan de proceso para los productos del tipo *P1*, detallado a un único nivel de abstracción. La operación *OP1* se ha desglosado en dos operaciones *OP11* y *OP13*, que representan la diferente ejecución en las máquinas *M1* y *M3*, respectivamente. La estructura causal definida por la red indica la disyunción de las operaciones *OP11* y *OP13*. El modelado según dos niveles de abstracción implicaría la definición en secuencia de las operaciones *OP1* y *OP2* al nivel más alto de abstracción y un siguiente nivel con el refinamiento de la operación *OP1* en una disyunción con las operaciones *OP11* y *OP13*. El proceso de *P2* se representa con una simple secuencia de sus operaciones como se muestra en la figura 4.17b.

Para describir la semántica de la representación elegida, nos centraremos ahora

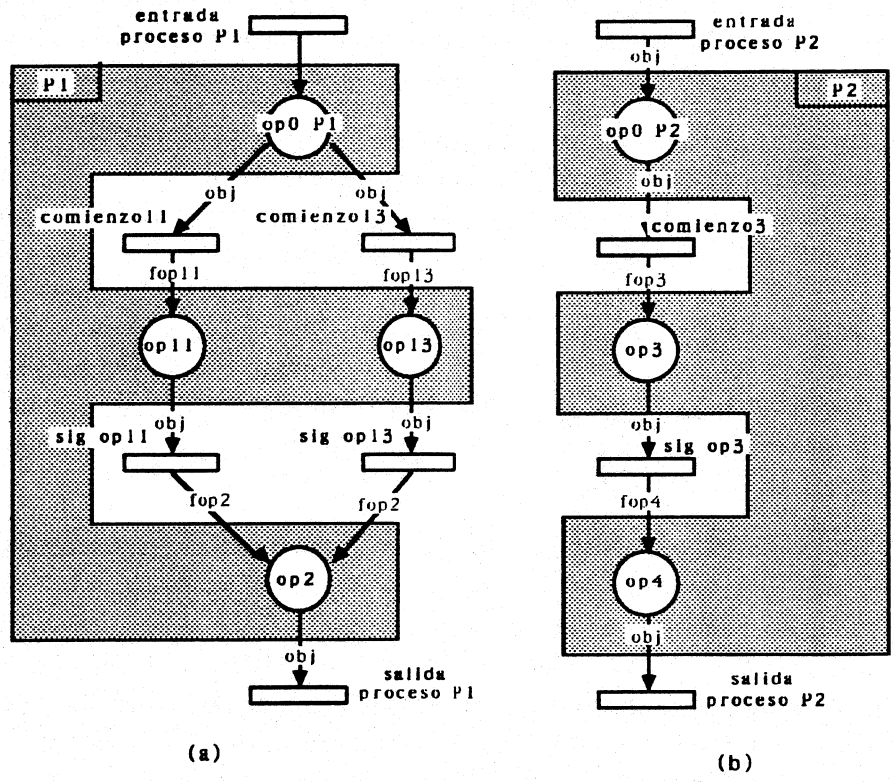

Figura 4.17: Red subyacente en el modelado del los procesos *P1* y *P2*.

{*P1*
instance : plan-de-proceso
estados : OP0-P1 OP11 OP13 OP2
...
OP2 :
type : atributo-estado
range : (TYPE instance op2)
pre-relacion-red-con : sig-op11 sig-op13
post-relacion-red-con : salida-proceso-P1
objeto-asociado :
{ **instance** : op2
recurso-primario : M2
estado : no-planeada
tiene-plan-proceso :
en-plan-proceso : P1
tiene-duracion : }
...
acciones :
comienzo11 comienzo13 entrada-proceso-P1 sig-op11
sig-op13 salida-proceso-P1
nivel-precision : nivel-maquina
plan-proceso-de : }

Figura 4.18: Objeto de estado que representa el plan de proceso *P1*.

en el objeto de estado mostrado en la figura 4.18 que representa el proceso *P1*. Los atributos *op0, op11, op13* y *op2* indican los posibles estados que puede alcanzar una pieza que lleve a cabo dicho plan de proceso (por ejemplo, el conjunto de piezas realizando la operación *op2* está identificado por los objetos de marcado en el atributo *OP2*). Cada atributo de estado dispone en su metaconocimiento de un dato asociado consistente en un objeto prototipo que define las características generales de la operación a realizar en la etapa del proceso que representa (por ejemplo, el atributo *OP2* tiene como dato asociado la operación *op2*).

Cada objeto de acción representa el comienzo de una operación de proceso y/o el final de otra. Las relaciones causales genéricas entre los prototipos de operaciones están determinados por las relaciones de red de su/s plan/es de proceso. Esto proporciona gran claridad sobre el modelo del proceso y evita la necesidad de introducir relaciones especiales, como *posibles-sucesores* y *posibles-predecesores*[3], que proponen otros modelos [SMIT 89].

4.4.1 Jerarquía de abstracción de operaciones y procesos

En general resulta difícil manipular la información directamente al más alto nivel de detalle. En el contexto de una simulación, por ejemplo, puede ser deseable considerar selectivamente distintos niveles de detalle en el modelado las diferentes áreas de la factoría. Similarmente, en la resolución de problemas de planificación, puede ser deseable diferentes niveles de abstracción dependiendo, por ejemplo, del grado de impredecibilidad inherente en el proceso que se modela, o del horizonte temporal de las decisiones de planificación a realizar.

Por ello es importante disponer de la posibilidad de definir agregaciones de operaciones, cuyo modelo conjunto refleje las características de las operaciones a más bajo nivel. Utilizando estas operaciones agregadas, será posible generar planes aproximados, sobre un horizonte más largo, con un cierto nivel de certeza de alcanzar los objetivos de producción. Esta aproximación jerárquica a problemas de modelado y planificación resulta ya clásica en la literatura (véase por ejemplo [SACE 74,FOX 83]). En el área de redes de Petri han aparecido también trabajos atendiendo al concepto de jerarquía fundamentalmente con propósitos de modelado y análisis, cabría mencionar a este respecto el trabajo [HUBE 89] sobre jerarquías en redes de Petri coloreadas.

Una operación particular de un plan de proceso, a cierto nivel de abstracción, a su vez puede describirse como otro plan de proceso en el nivel de abstracción inferior. En la figura 4.19 se destacan varios atributos del objeto *operacion-fabricacion* relacionados con esta composición jerárquica que se añaden a los reseñados anteriormente:

[3] "Possible-successors" y "possible-predecessors" según se denota en [SMIT 89].

{*operacion-fabricacion*
is-a : operacion
tiene-plan-proceso :
plan-proceso-de :
suboperaciones :
suboperacion-de :
primera-suboperacion :
primera-suboperacion-de :
ultima-suboperacion :
ultima-suboperacion-de :
nivel-precision :
range : (TYPE instance *niveles-de-precision*)
tipo : }

Figura 4.19: Descripción jerárquica del objeto *operacion-fabricacion*.

- La relación *tiene-plan-proceso* del prototipo de la operación indica el plan de proceso del nivel inferior (su inversa es *plan-proceso-de*).
- La conexión se produce a través de los objetos de acción de comienzo y final, de la operación y del plan de proceso. Esta conexión precisa el establecimiento de la relación *sincronizacion-nivel-inferior* y su inversa *sincronizacion-nivel-superior*.
- Una vez conocida la trayectoria de una pieza por el plan de proceso, puede resultar más interesante (fundamentalmente en aplicaciones de planificación y toma de decisiones) trabajar con las operaciones instanciadas. Las relaciones *primera-suboperacion*, *ultima-suboperacion* y *suboperaciones* permiten determinar las instancias del nivel inferior (sus inversas son respectivamente *primera-suboperacion-de*, *ultima-suboperacion-de* y *suboperacion-de*).
- Finalmente *nivel-precision* indica la localización de operación en la jerarquía de abstracción.

La descripción jerárquica de procesos de fabricación permite regular el nivel de detalle sobre el que razonar en las decisiones de asignación de recursos. La determinación de un apropiado conjunto de niveles de abstracción de procesos, es obviamente una función del sistema de fabricación particular que se modele. Concretamente, las características y organización de los recursos que deben ser asignados en el sistema de fabricación dictarán los niveles de interés en la toma de decisiones.

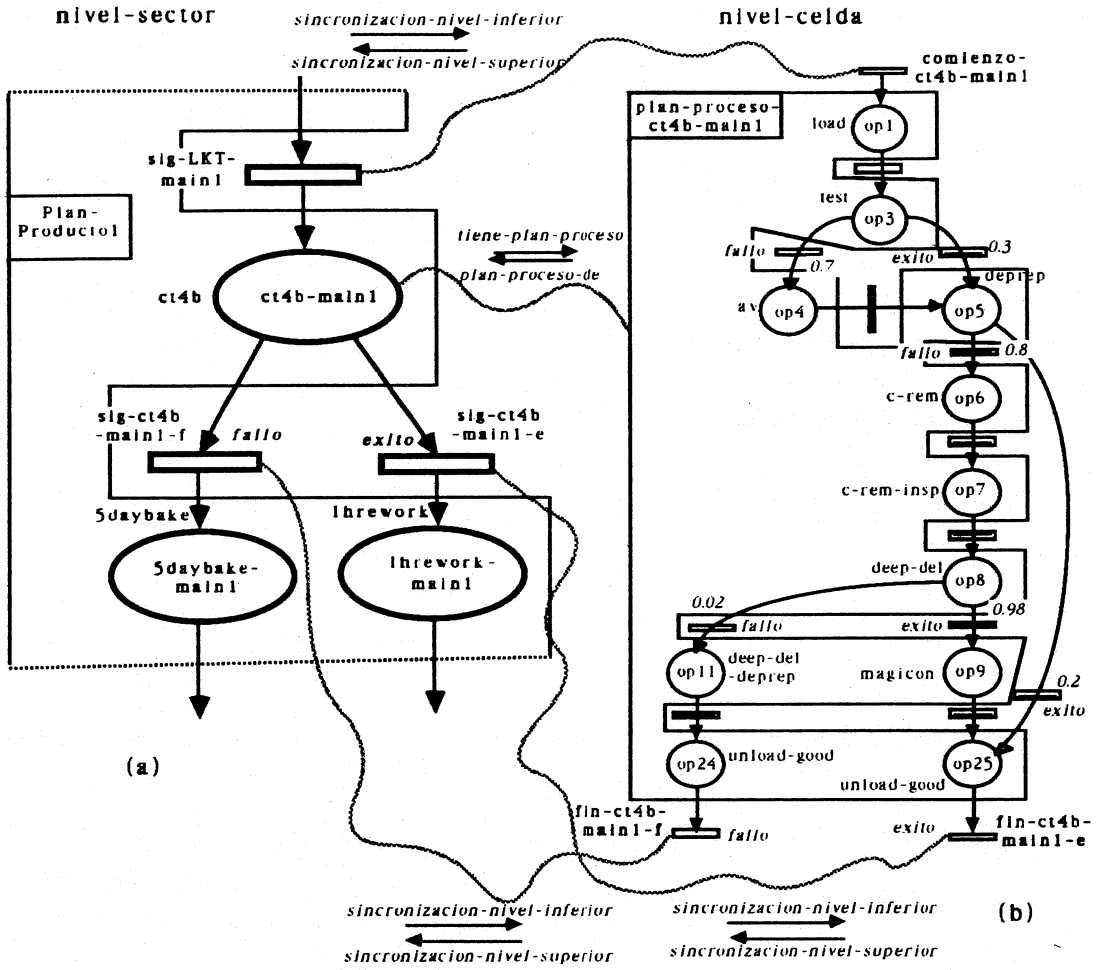

Figura 4.20: Red del plan de proceso *ct4b* a niveles de (a) sector y (b) celda.

4.4.2 Ejemplo de operaciones y procesos

Para ilustrar el modelado jerárquico de operaciones y procesos, consideraremos un plan de proceso relativamente más complicado que los anteriores. El ejemplo, correspondiente a un sistema real, corresponde a una línea de ensamblado de placas de computador, que será abordada posteriormente de manera más extensa. La figura 4.20 muestra la red del prototipo de proceso para un sector de la factoría denominado *ct4b* que se encuentra representado a dos niveles de abstracción [SMIT 89]: formando parte del proceso *plan-producto1* a nivel sector (figura 4.20a) y como plan de proceso a nivel de celda (figura 4.20b).

De forma análoga a lo sucedido en las jerarquías de abstracción de recursos, la conexión de unos niveles de definición de operaciones y procesos con otros se lleva a cabo mediante el estableciendo de relaciones. Así, la relación *tiene-plan-proceso* conecta la operación *ct4b-main1*, representada a nivel de sector, con su refinamiento a nivel celda representado por el plan de proceso *plan-proceso-ct4b-*

{*sig-op3-f*
instance : objeto-accion-plan-proceso
pre-relacion-red :
(ct4b-main1 op3 obj)
post-relacion-red :
(ct4b-main1 op4 next-op3f)
condicion : fallo
probabilidad : 0.7
nivel-precision : nivel-celda
sincronizacion-nivel-inferior :
dato-asociado : < *vop*, *vsigop* >
predicado : (eq < *vsigop* > *op4*) }
; donde:
obj ≡ {**schema-name** < *vop* >}
next-op3f ≡ {**instance** < *vsigop* >
previa-operacion < *vop* >}

Figura 4.21: Objeto de acción *sig-op3-f* del plan de proceso *plan-proceso-ct4b-main1*.

main1. La conexión se produce entre los objetos de acción *sig-lkt-main1*, *comienzo-ct4b-main1*; *sig-ct4b-main1-f*, *fin-ct4b-main1-f* y *sig-ct4b-main1-e*, *fin-ct4b-main1-e* mediante relaciones *sincronizacion-nivel-inferior*. Esta posibilidad de modelado a distintos niveles de abstracción resulta realmente útil en aplicaciones de tipo jerárquico como es la planificación de operaciones. En este caso, la planificación a nivel de sector impone restricciones temporales a la instancia de la operación *ct4b-main1*, al propagarse estas restricciones al nivel de precisión de celda imponen los limites temporales al plan de proceso completo, resultado de la instanciación de *plan-proceso-ct4b-main1*.

Los objetos de acción permiten incluir otro tipo de informaciones no incluidas anteriormente, pero que resultan útiles para el modelado completo de un plan de proceso (véase como ejemplo el objeto de acción *sig-op3-f* mostrado en la figura 4.21):

- *Probabilidad*: probabilidad de la acción, utilizada, por ejemplo, en el sistema de planificación para realizar instanciaciones en consonancia con estas probabilidades.
- *Condicion*: condición de terminación en operaciones de control de calidad.

Estos ejemplos ponen de manifiesto que el método propuesto para la definición de los procesos de fabricación a modo de red, permite un modelado relativamente

sencillo y gráfico de representación de composiciones de operaciones como secuencias, procesos alternativos, operaciones de ensamblaje y desensamblaje, control de calidad y concurrencia de operaciones.

4.5 Conexión de Perspectivas de Recursos y Operaciones

Hasta el momento se ha contemplado el modelado del sistema de fabricación desde una doble perspectiva según los puntos de vista de los recursos y de las operaciones a partir de su consideración como modelos independientes. Sin embargo, es obvio que, durante la emulación del funcionamiento del sistema, los modelos de ambas perspectivas evolucionan acompasadamente y resulta necesario, por tanto, establezer un mecanismo de sincronización que acomode la evolución del flujo de información según las perspectivas de ambos modelos parciales, pues el sistema completo deberá soportar sus restricciones conjuntas.

Este mecanismo de sincronización se construye mediante un nuevo tipo de relación denominado **relación de sincronización parcial**. Las relaciones de sincronización parcial se establecen entre objetos de acción pertenecientes a los planes de proceso y los objetos de acción del modelo de los recursos, que representen hechos o situaciones de decisión similares. Las relaciones anteriores sincronizan el objeto de acción del plan de proceso, y un modo de disparo del objeto de acción del recurso (disparo con respecto a una operación particular), que modelan la misma situación (p.e. fin o principio de operación)

Las relaciones de sincronización parcial son un concepto más complejo que las relaciones de sincronización. Desde el punto de vista de las RAN, estas relacciones implican el despliegue del objeto de acción del recurso en todos sus posibles modos de disparo. Sin embargo, para hacer la descripción de todo este proceso transparente al usuario, el sistema de representación dispone de un objeto de acción especial, denominado **objeto de sincronización condicional** que embebe este mecanismo de sincronización.

4.5.1 Ejemplo de coordinación de perspectivas de recursos y operaciones

Para aclarar los conceptos introducidos en este punto, retomemos nuevamente el ejemplo de la celda robotizada que se viene desarrollando a lo largo de este capítulo. El modelo parcial desde la perspectiva de los recursos está representado por la red de la figura 4.12a y la perspectiva de las operaciones está representada por las redes de la figura 4.17. El siguiente paso, en esta metodología de modelado, es

coordinar dichas perspectivas parciales estableciendo las mencionadas relaciones de sincronización parcial. El método más sencillo consiste en determinar, para cada objeto de acción de los planes de proceso, cuál es el punto de la red que modela los recursos en que debe ejecutarse la misma acción, así para el proceso *P1*:

- El comienzo del proceso según la perspectiva de las operaciones coincide con la entrada de la pieza en la celda de trabajo según la perspectiva del recurso. Esto indica que habrá de sincronizarse la acción *entrada-proceso-P1* con la entrada en la celda *entrada-almacen1*.
- La elección entre las operaciones *op11* y *op13* se realizará cuando se encuentre la pieza en el almacén y deba optarse por uno de los recursos *M1* ó *M3*. Por tanto se habrá de sincronizar la acción *comienzo1* con la acción *carga-M1* del recurso *M1* y la acción *comienzo3* con *carga-M3* de *M3*. Por el mismo razonamiento se deben sincronizar las acciones *sig-op11* y *sig-op13* con *carga-M2*.
- El final del procesamiento corresponderá a la salida de la celda, lo que se consigue sincronizando *salida-proceso-P1* con *salida-almacen1*.

Un razonamiento análogo permite encontrar las sincronizaciones del proceso *P2*. La figura 4.22 muestra la estructura completa del sistema con conexión entre el modelo de los recursos y los planes de proceso.

4.6 Conclusión

En este capítulo se ha presentado la parte de la herramienta de modelado especializada en el dominio de sistemas de fabricación. El objetivo es incrementar la potencia representativa de KRON con un nivel de representación adicional que facilite la tarea de modelado en aplicaciones específicas de fabricación. Operativamente se han desarrollado dos aspectos: por una parte las primitivas relativas a objetos, relaciones y métodos que representan prototipos de conceptos utilizados en las aplicaciones de fabricación; por otra parte una metodología de modelado útil, tanto para la construcción de las jerarquías de especialización de conceptos, como para su conexión formando subsistemas.

La descripción del comportamiento dinámico de los objetos está sustentado por la red KRON y soporta los requerimientos de herencia. Las características dinámicas, definidas por la RAN subyacente, pueden ser heredadas de unos objetos a otros, más especializados, descendientes en la jerarquía. El modelado se puede realizar siguiendo diferentes perspectivas (perspectiva basada en los recursos y perspectiva basada en las operaciones), y es posible establecer conexiones entre redes KRON según diferentes criterios, en una dirección horizontal entre redes pertenecientes al

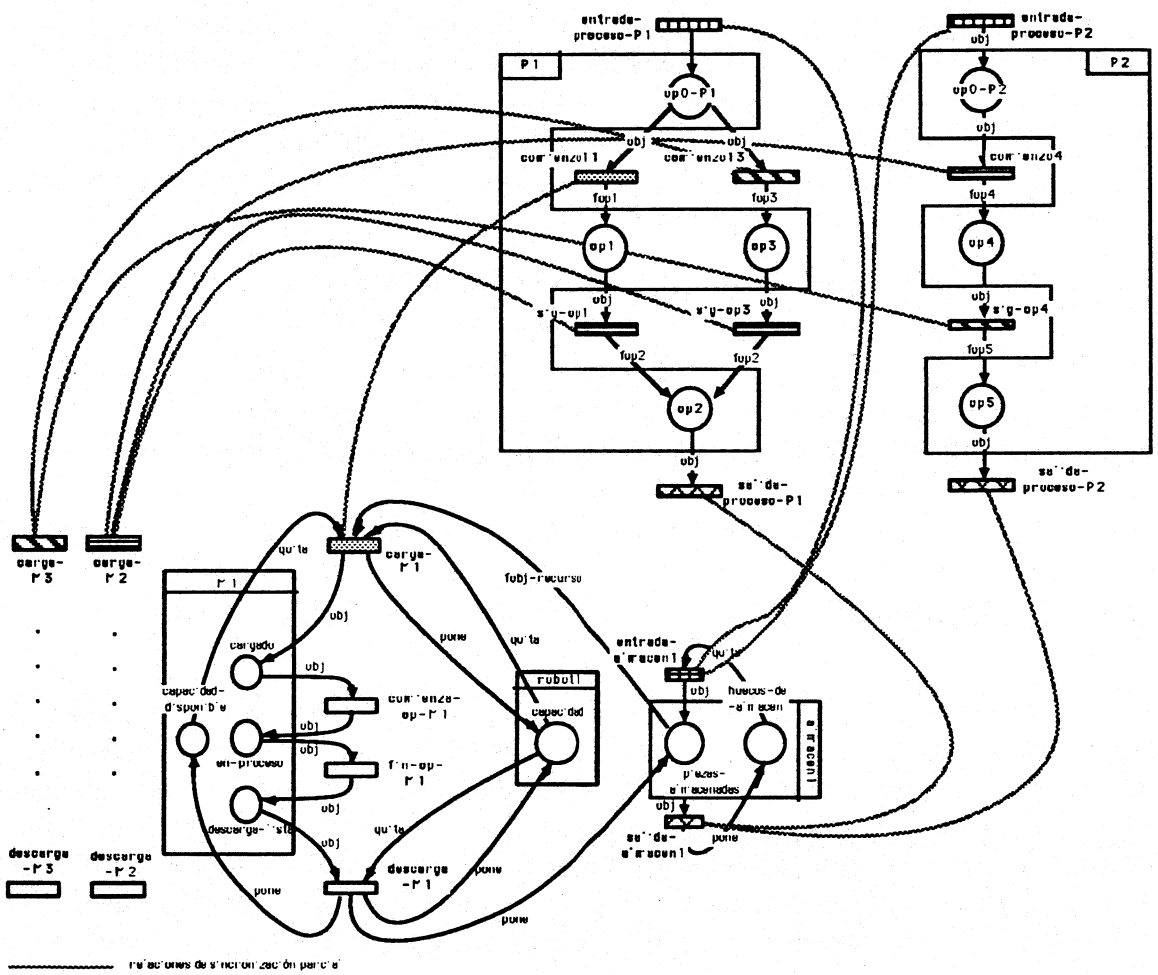

Figura 4.22: Red KRON completa con recursos y plan de proceso.

mismo nivel de abstracción, y en una dirección vertical conectando diferentes niveles de abstracción.

KRON proporciona al usuario la posibilidad de adaptar la semántica de la herramienta a sus necesidades individuales sin forzar perspectivas particulares, por lo que puede ser utilizado de manera similar en aplicaciones de otros dominios. Por otra parte, la representación puede ser extendida en el propio dominio incrementando los objetos de la jerarquía de especialización y las relaciones definibles por el usuario.

Por último, habría que destacar que las posibilidades de análisis del modelo debidas a las propiedades analizables en una RAN, que fueron mencionadas en el capítulo anterior, son extrapolables a las aplicaciones desarrolladas a nivel del dominio.

HISTORIA

Un cronopio pequeñito buscaba la llave de la puerta de calle en la mesa de luz, la mesa de luz en el dormitorio, el dormitorio en la casa, la casa en la calle. Aquí se detenía el cronopio, pues para salir a la calle precisaba la llave de la puerta.

Historias de cronopios y famas
JULIO CORTAZAR

Capítulo 5

Planificador de Operaciones

En los capítulos previos se ha abordado la aplicación de técnicas de IA a la representación y metodología de construcción de modelos de comportamiento de sistemas de fabricación. En los siguientes capítulos se abordará la utilización de técnicas de IA en la toma de decisiones para el control de sistemas de fabricación. La estrategia de toma de decisiones que se propone consta de dos niveles: un **distribuidor de operaciones**, que está en contacto directo con un sistema coordinador y da respuesta a problemas de decisión a corto plazo, y un **planificador de operaciones** que trabaja haciendo previsiones de planes de producción a medio y largo plazo. Este último nivel constituye el objeto del presente capítulo. Las características propuestas para el planificador de operaciones, forman parte de los trabajos realizados por el autor de la memoria durante su participación en el desarrollo del sistema OPIS en el Intelligent Systems Laboratory y el Center for Integrated Manufacturing Decision Systems del Robotics Institute de la Universidad de Carnegie Mellon.

La primera parte se dedica a establecer el problema y se hace una revisión de los métodos utilizados para planificación de operaciones[1].

A continuación se propone un planificador basado en las ideas de búsqueda dirigida por restricciones, razonamiento oportunista con una visión común sobre planificación predictiva y reactiva. Su construcción está basada en los principios de arquitecturas de pizarra estandard. En esta arquitectura, el control está dirigido por eventos y existen una serie de fuentes de conocimiento utilizados para planificación y análisis en respuesta a dichos eventos.

Para abordar el problema de elegir la fuente de conocimiento más adecuada a cada problema, se propone una metodología en la que, a partir de una caracterización

[1]Se ha adoptado la traducción *planificación de operaciones* para el conocido término "*scheduling*".

de los conflictos, se generan las heurísticas de meta-conocimiento de control, que guiarán a la fuente de conocimiento de gestión de alto nivel en su proceso de búsqueda de la mejor acción a aplicar.

5.1 Métodos de Planificación de las Operaciones

Uno de los problemas más complejos en el control de un sistema de fabricación es la organización del flujo de las tareas que deben ser realizadas por los elementos operativos del sistema. Este problema, conocido como problema de planificación de la producción, puede establecerse en términos generales como el problema de realizar una asignación de recursos (máquinas, herramientas, robots, transportes, instalaciones de almacenamiento, etc.) a operaciones a lo largo del tiempo, de forma que conduzca a la realización de las órdenes de trabajo en el instante oportuno y a un coste eficiente. Las operaciones deben se planeadas de forma que satisfagan las restricciones tecnológicas asociadas a los procesos de producción y las limitaciones de capacidad del conjunto de recursos compartidos. Por otra parte, el último requerimiento involucra típicamente el compromiso entre un conjunto de objetivos en conflicto (p.e. minimización del trabajo en proceso, maximización de la utilización de recursos, encontrar fechas debidas, etc.). La dificultad del problema de planificación de operaciones se pone de manifiesto por las siguientes características:

- El problema del cálculo de un plan óptimo de operaciones resulta ser, excepto para casos triviales, NP-completo, incluso bajo hipótesis académicas (que reduzcan la complejidad del problema) que contemplen la naturaleza de las restricciones tecnológicas y los criterios de optimización [GARE 79].

- No existen planes de operaciones para los que todas las restricciones puedan ser satisfechas.

- El comportamiento del entorno de un sistema de fabricación resulta impredecible.

El tratamiento de este problema se ha abordado fundamentalmente desde dos perspectivas diferentes. La primera y más tradicional está basada en la utilización de métodos de investigación de operaciones. La segunda se centra en la aplicación de métodos de inteligencia artificial.

Dentro de los **métodos de investigación de operaciones** se han desarrollado dos tipos de técnicas radicalmente distintas. Por una parte se han utilizado técnicas de optimización basadas en programación matemática [JETE 86], aplicadas fundamentalmente a problemas particulares (en su mayoría de tipo teórico); mientras que las aplicaciones prácticas se han basado en el uso de heurísticas de

despache[2] [GRAV 81].

A lo largo del presente capítulo se van a utilizar una serie de términos cuyo significado se introduce ahora con objeto de evitar posibles interpretaciones diferentes. Un sistema de fabricación debe satisfacer una serie de *órdenes* de producción. La satisfación de una orden tendrá asociada la ejecución o realización de un *trabajo* o *tarea*. Estos se realizarán de acuerdo con *planes de trabajo* o *planes de proceso* que constan de una serie de operaciones, que deben ser asimismo realizadas siguiendo un cierto orden, sobre unos *recursos* determinados.

5.1.1 Métodos de programación matemática

La metodología general de resolución con *programación matemática* [JETE 86] parte de un modelo cuantitativo del problema y se trata de transformar un problema complejo en otro con estructura más simple. Una vez seleccionado un método de optimización se determina la solución del nuevo problema. Si la solución obtenida no es de calidad suficiente se procede a una nueva reformulación del problema. Cabe destacar cuatro métodos de optimización: programación lineal (fundamentalmente en sus versiones de programación lineal entera y 0-1), método de separación y evaluación progresiva[3], relajación lagragiana y programación dinámica [MURO 87]. Estos métodos utilizan la técnica de modelado cuantitativa en distintas versiones lineales, enteras y no lineales. La técnica primaria de resolución de problemas es la satisfación de restricciones y la aproximación dominante es la reformulación del problema como un problema de programación lineal. Generalmente utilizan el método simplex o el de Karmarkar. La ventaja característica es la consecución de soluciones óptimas o cuasióptimas. La desventaja gravísima de estos métodos se deriva de la imposibilidad de manejar la complejidad de la mayor parte de los problemas de planificación de operaciones.

Las *reglas de despache* trabajan en base a prioridades definidas relativas a criterios particulares. Funcionan a modo de reglas de decisión que se aplican en el momento en que es necesaria una decisón de planificación particular [PANW 77]. En planificación predictiva, el método de resolución consiste en una simulación de la factoría dirigida por eventos. Cuando llega una trabajo o un recurso queda libre, se utiliza una regla de despache local para seleccionar el siguiente trabajo a asignar al recurso. La elección de la regla local depende de los criterios utilizados. Estos pueden referirse a la cantidad de trabajo que queda, la proximidad a la fecha debida, los tiempos de proceso de los trabajos, etc. Los resultados de esta simulación conforman el plan de operaciones. Generalmente se realizan varias simulaciones en las que los resultados de cada simulación son utilizados para mejorar los de la siguiente. Como aspecto positivo, las reglas de despache son sencillas de aplicar y

[2] "Dispatch heuristics" según su denominación inglesa.
[3] "Branch and Bound" según su denominación inglesa.

proporcionan una base directa para realizar un control en tiempo real. Sin embargo, la confianza en estas técnicas es limitada. Unicamente se han conseguido éxitos en la planificación de sistemas de fabricación relativamente simples fundamentalmente debido a una visión miope de la factoría que proporciona una satisfación local en lugar de la optimización global pretendida. El intento de optimizar únicamente uno o dos objetivos y las dificultades para tratar con restricciones constituyen otras limitaciones adicionales.

Cabría mencionar también algunos trabajos relativos a la planificación de operaciones con métodos basados en redes de Petri [SILV 89], que si bien no han producido aportaciones especialmente novedosas, dejan translucir nuevas posibilidades en el contexto de la arquitectura de control de producción que aquí se propone. Destacan a este respecto las aproximaciones con redes temporizadas entre las que merecen especial atención los trabajos de Chretienne [CHRE 88].

5.1.2 Métodos de inteligencia artificial

Las aproximaciones a la planificación de la producción basadas en técnicas de inteligencia artificial han surgido recientemente como alternativas a las aproximaciones de despache local. Estas aproximaciones tratan la complejidad del problema de planificación de operaciones explotando el conocimiento disponible sobre el entorno de producción y sus restricciones. Revisiones de tales aproximaciones pueden encontrarse en [STEF 86,KUSI 88]. Habría que destacar también, como apunta Steffen, la limitada implantación operativa de sistemas de planificación de operaciones basados en técnicas de IA pues la mayoría de los trabajos han sido realizados exclusivamente a nivel de prototipo.

En terminología de IA, la planificación de operaciones en sistemas de fabricación es un problema con las siguientes características [FOX 86]:

- Se trata de un problema de planificación *basado en el tiempo* en el que las actividades deben ser seleccionadas y secuenciadas y debe asignarseles recursos y tiempos de ejecución.
- Se trata de un problema de planificación *multi-agente* donde cada orden de trabajo y/o cada producto representa un agente separado para el que es necesario crear un plan. El número de agentes puede ser muy elevado.
- Los agentes son *no-cooperativos*, es decir, cada uno sólo intenta maximizar sus propios objetivos.
- Existe una *alta disputa* por los recursos, lo que provoca decisiones acopladas.
- La búsqueda es *combinatoriamente explosiva*, 85 órdenes de trabajo, cada una con 10 operaciones sin caminos alternativos, con una única máquina posible

por operación y sin considerar tiempos de inactividad, admite 10^{880} planes posibles.

Cuatro han sido los métodos utilizados: aproximaciones basadas en reglas, etiquetado de restricciones[4], negociación distribuida[5] y búsqueda dirigida por restricciones.

Las **aproximaciones basadas en reglas** se basan en la codificación de la experiencia del experto en forma de reglas y adoptan una representación de la factoría orientada a objetos. Se han aplicado para la distribución de trabajos y la planificación de factorías pequeñas. Los inconvenientes provienen de la dificultad para extraer el conocimiento así como en la desconfianza en el propio conocimiento del experto. Destacan trabajos como [ORCI 84], [ACOC 86] o [BRUN 86].

El método de resolución mediante **etiquetado de restricciones** representa el problema mediante un conjunto de variables con restricciones (usualmente binarias) entre ellas y se fundamenta en una aproximación con encadenamiento hacia atrás para la asignación de valores a dichas variables: a) se elige una variable para asignar un valor, b) se le asigna un valor que satisface las restricciones conocidas y c) si el valor no resulta admisible, se vuelve hacia atrás a la última asignación y se elige otro valor. Para reducir la complejidad de la búsqueda existen multitud de métodos para elegir la variable y el valor a asignar. En estos métodos la planificación reactiva se consigue v'ia la retracción y adición de restricciones. Constituye un método interesante para planificación reactiva, sin embargo, el tamaño de los problemas que puede manejar es relativamente pequeño y limitado en el número de restricciones, por lo que sólo resulta útil para situaciones simples.

En el método de **negociación distribuida** se representa el problema mediante agentes separados y distribuidos a través de la planta. Cada agente gestiona sus propias tareas y las relaciones entre agentes se definen mediante contratos. El método de resolución se focaliza en los protocolos de la negociación que actúan basicamente de la siguiente forma:

- el agente supervisor de la planta hace una llamada de requerimiento de ofrecimientos[6] para realizar una operación (la llamada define la duración, tiempo de finalización y recursos necesarios);
- los agentes de los centros de trabajo responden con ofrecimientos (el ofrecimiento define el instante de finalización[7] y el coste);
- el agente supervisor adjudica un contrato al ofrecimiento que cumple la fecha debida y los requerimientos de coste y hace una nueva llamada para la siguiente

[4] "Constraint labeling" según su denominación inglesa.
[5] "Distributed negociation" según su denominación inglesa.
[6] "Call" y "bid" en sus denominaciones inglesas.
[7] "Date to deliver" según su denominación inglesa.

operación y sigue el proceso;

- los agentes también deben reconocer la adjudicación del contrato.

Este método sólo resulta interesante para entornos de fabricación muy flexibles pues en otro caso ocurren comportamientos monopolistas. Otra limitación reside en el hecho de que la búsqueda para contratar a un agente es una simple política del tipo "el mejor primero"[8] que no resuelve la complejidad de dicha tarea.

5.1.3 Búsqueda dirigida por restricciones

H.A. Simon apuntó la existencia de tres formas distintas para representar y pensar acerca de las tareas de resolución de problemas: *búsqueda* a través de un espacio de nodos y arcos, *razonamiento* que permita deducir nuevas aserciones a partir de un conjunto de axiomas en un lenguaje formal y *satisfacción de restricciones* donde la adición incremental de restricciones reduce el conjunto de posibilidades a un subconjunto que satisface también las nuevas. Estas formas no sólo no son mutuamente excluyentes, sino que incluso pueden complementarse fructíferamente. Así, la combinación de búsqueda, heurísticas y restricciones para guiar en el espacio de búsqueda ha sido la técnica de más éxito para resolver problemas reales combinatoriamente complejos. Entre los ejemplos más representativos destacan los sistemas Hersay II para reconocimiento de la palabra[ERMA 80], Molgen para planificación de experimentos moleculares [STEa 81,STEb 81], R1/XCON para proponer configuraciones de computadores[McDE 82], ISIS para planificación de operaciones en sistemas de fabricación [FOX 83] o GARI utilizado para planificación de procesos de producción [DESC 85].

La metodología aplicada en estos sistemas tiene dos particularidades[9]:

- *Refinamiento jerárquico.* Se disponen varios niveles de abstracción y se consideran tipos específicos de restricciones en cada paso de refinamiento. Las soluciones de cada nivel restringen la búsqueda realizada en los niveles inferiores.

- *Búsqueda heurística basada en restricciones.* Las restricciones definen los operadores a aplicar y las funciones heurísticas de evaluación y limitan el espacio de búsqueda.

Se puede decir, a modo de conclusión, que el método de búsqueda dirigida por restricciones resulta apropiado para realizar la planificación de operaciones a varios niveles de la planta pudiendo manejar su complejidad. Permite asímismo satisfacer múltiples restricciones simultaneamente e integrar planificación predictiva y reactiva.

[8] "Best first" según su denominación inglesa.

[9] Estas particularidades están desarrolladas muy diferentemente en cada sistema pero todas responden a los mismos principios básicos.

5.2 Arquitectura del Planificador de Operaciones

La aproximación elegida se construye sobre la metodología para la planificación de operaciones desarrollada en el sistema OPIS [OW 88a]. Esta metodología está sustentada en la visión de la planificación como una actividad dirigida por restricciones que puede soportar la generación de planes de operaciones que reflejen de manera fiable las características y objetivos del entorno de producción, y la revisión incremental de estos planes en respuesta a cambios no anticipados del estado de la factoría (p.e. fallos de máquinas, rechazos del control de calidad, etc.). El plan de operaciones puede, de esta forma, proporcionar una guía en tiempo de ejecución no disponible en las aproximaciones basadas en reglas de despache.

La arquitectura del planificador de operaciones, componente del sistema de toma de decisiones, está motivada por el deseo de enfocar dinámicamente el esfuerzo de planificación de acuerdo con las restricciones relevantes del problema a tratar y con las posibilidades de trabajo *predictivo* y *reactivo*, haciendo uso además, de una metodología de resolución de problemas de tipo *jerárquico* y *oportunista*. La búsqueda realizada está dirigida por restricciones en el espacio de todos los planes posibles. Los diferentes niveles de búsqueda proporcionan múltiples abstracciones del problema de planificación de operaciones. Cada una es función de los tipos específicos de restricciones que son considerados a cada nivel.

El término **razonamiento oportunista**[10] se ha venido utilizando para caracterizar los procesos de resolución de problemas donde la actividad de razonamiento se dirige hacia aquellas acciones que aparecen como más prometedoras en términos del estado de resolución del problema considerado. Esta noción de oportunismo ha sido extensamente tratada en aplicaciones de planificación [OW 88a] aunque, por otra parte, el campo de aplicación ha sido mucho más amplio, siendo la idea germen de la aparición de arquitecturas genéricas especialmente dedicadas [ERMA 80].

Adicionalmente, la metodología de planificación que subyace en el sistema de decisión presentado en el presente capítulo aboga por una visión de **planificación de operaciones predictiva y reactiva** común. En este marco de trabajo, el sistema reconcilia incrementalmente las discrepancias que surgen entre el plan de operaciones previsto y el comportamiento real del entorno de producción, interpretando dicha predicción como un conjunto de restricciones que han sido impuestas sobre la operación de la factoría [SMIT 87].

La aproximación que se va a desarrollar constituye una extensión de la arquitectura presentada en [FOX 84,SMIT 86,SMIT 87]. En relación con dicha arquitectura, se ha modificado sensiblemente el ciclo de control del planificador

[10] "Opportunistic reasoning" según su denominación inglesa.

para potenciar de forma espec'ifica posibilidades de análisis de conflictos surgidos en el plan de operaciones. Se han ampliado y mejorado las fuentes de conocimiento existentes y se ha incluido una fuente de análisis nueva. La caracterización de conflictos y la manipulación de dicha información a modo de heurísticas de meta-conocimiento de control es así mismo nueva.

5.2.1 Arquitectura de pizarra

La arquitectura del módulo de planificación de operaciones que se propone (mostrada en la figura 5.1), está basada en los principios de **arquitecturas de pizarra estandard**[11] [ERMA 80,NII 86a,NII 86b].

En una arquitectura de pizarra, una colección de métodos, llamados *fuentes de conocimiento*, pueden ser aplicados selectivamente para generar, revisar y analizar componentes específicos del plan de operaciones completo. El planificador hace uso de dos tipos de fuentes de conocimiento con objetivos diferenciadas: métodos de planificación y métodos de análisis. Los *métodos de planificación* varían según el punto local de enfoque que asumen y los tipos de restricciones sobre las que hacen énfasis, y definen estrategias alternativas para resolver una tarea de planificación dada. Los *métodos de análisis* proporcionan información adicional relativa a interacciones entre restricciones y parámetros de caracterización local del plan de operaciones. La responsabilidad del uso y coordinación de estos métodos está asumida por una fuente de conocimiento especial de control, denominda *gestor de alto nivel*, que combina mecanismos de *propagación de restricciones* y *mantenimiento de consistencia*, con *conocimiento heurístico de alto nivel* que permite relacionar las restricciones y parámetros del plan, con estrategias específicas de planificación predictiva o reactiva.

La *base de conocimiento* del sistema de planificación contiene información relativa a la hipótesis del plan considerado hasta el momento, con las restricciones temporales de las operaciones y de la capacidad de los recursos. Asimismo dispone de conocimiento más general relativo al modelo, como restricciones físicas, causales y de preferencia, así como de conocimiento heurístico de alto nivel utilizado por el gestor.

El gestor de alto nivel es informado de los resultados de la actividad de las fuentes de conocimiento, de los cambios en el plan de operaciones y de los cambios que tienen lugar en la factoría por medio del envío de *eventos de control*. Los eventos contienen información relativa a las características de las restricciones de la solución (p.e. la contención parece ser alta en un recurso particular), al progreso hecho hacia la solución final (p.e. se ha generado o revisado otro componente del plan), o permiten identificar problemas específicos de la solución considerada (p.e. violaciones de restricciones no ralajables) [SMIT 87].

[11] "Blackboard architectures" según su denominación inglesa.

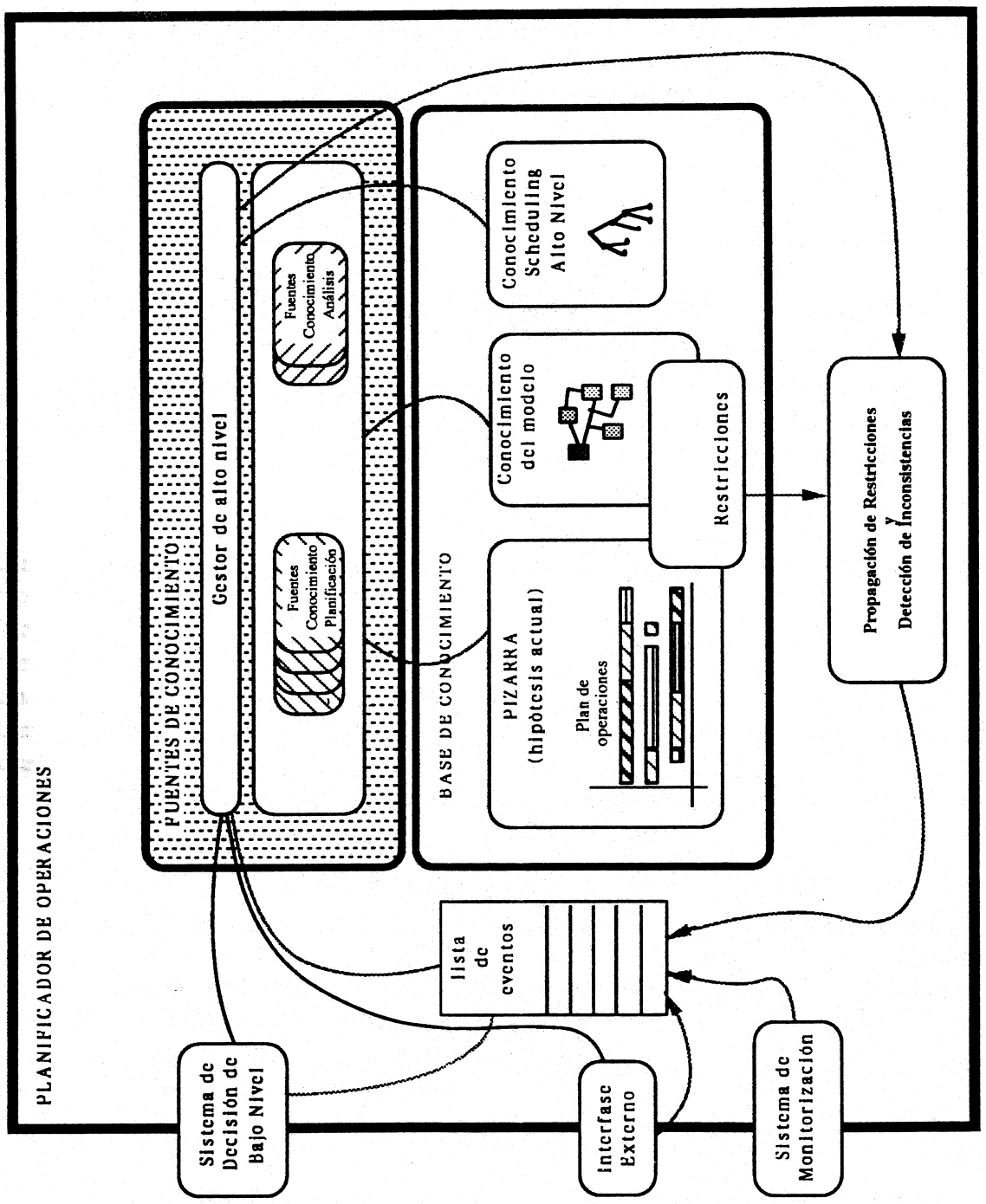

Figura 5.1: Arquitectura del planificador.

Como se ha argumentado anteriormente, la arquitectura hace uso extensivo de la idea de similitud entre las actividades de generación y de revisión reactiva del plan de operaciones, por lo que la gestión de ambas estrategias se realiza con una visión integrada. La invocación del módulo de planificación se realiza creando una tarea que active el *gestor-alto-nivel* que, dependiendo de la estrategia elegida, puede hacerse de tres formas:

1. *Modo generativo*: provoca la generación de un nuevo plan de operaciones completo.
2. *Modo reactivo*: modifica el plan de operaciones previo para obtener uno nuevo donde se encuentren satisfechas las restricciones no relajables.
3. *Modo reactivo local*: focaliza el esfuerzo de reacción en una zona local del plan de operaciones de forma que únicamente resulten necesariamente satisfechas las restricciones no relajables relativas a dicha zona.

5.3 Representación del Plan de Operaciones: la Pizarra

El conocimiento acerca del entorno de producción está modelado sobre las representaciones estructuradas del conocimiento, especializadas en el dominio de sistemas de fabricación, que fueron presentadas en capítulos anteriores. Este tipo de representación proporciona una organización relacional y estructural del conocimiento sobre actividades de fabricación, recursos, productos y demandas.

Dado un modelo de factoría, las operaciones componentes de las ordenes de trabajo se hacen explícitas "instanciando" el apropiado plan de proceso para cada orden que va a ser producida. Las instancias de las suboperaciones componentes de una orden/operación concreta, se generan al hacer evolucionar el objeto de marcado correspondiente a la orden/operación en el plan de proceso considerado de forma independiente. Este proceso de evolución identifica una secuencia de marcado en el plan de proceso que definirá una secuencia de instancias de operaciones. Dicha secuencia se hace, a su vez, explícita por medio de las relaciones casusales *operacion-previa* y *operacion-siguiente*.

En el proceso de evolución de la red apuntado anteriormente, pueden surgir situaciones en las que sea preciso la elección entre varios caminos alternativos del plan de proceso. La resolución de estos conflictos puede hacerse adoptando una estrategia según las probabilidades asociadas a los objetos de acción (*instanciación probabilista*), esto se consigue asociando a los planes de proceso simples políticas de control en base a estas probabilidades. En este caso, las secuencias obtenidas reflejarán las probabilidades de disparo asociadas a los objetos de acción. Esto

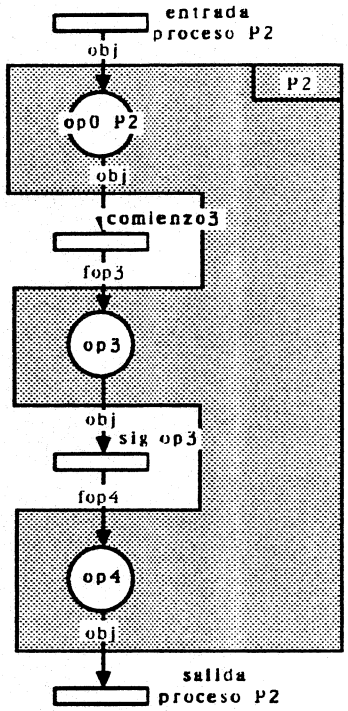

Figura 5.2: Red para el plan de proceso *P2*.

resulta, de alguna manera, lógico puesto que dichas probabilidades son la única información predictiva disponible. Este método proporporciona una representación de posibles secuencias de operaciones que pueden ser realizadas en la factoría al nivel de precisión que se considere[12].

Para ilustrar este proceso, considerese la descripción del plan de proceso *P2* presentado en el capítulo anterior y mostrado en la figura 5.2. La instanciación de una secuencia de operaciones para producir una unidad de producción se genera considerando el plan de proceso *P2* aislado y disponiendo el objeto de marcado correspondiente a la unidad de producción en el atributo de estado *op0-P2*. El disparo de *comienzo3* conectará una instancia de la operación *op3* a través de la función de la condición de red *fop3*. Identicamente ocurrirá en el disparo del objeto *sig-op3* para conectar una instancia de *op4* por medio de *fop4*. Las relaciones de precedencia de la secuencia (*previa-operacion* y *siguiente-operacion*) se forman a través de las ligaduras de la variable $< vobj >$ en el disparo de cada objeto de acción (vease figura 5.3).

[12]Experimentos realizados han demostrado que esta estrategia de instanciación proporciona secuencias representativas cuando se trabaja con conjuntos de tamaño medio y grande (> 25).

ob ≡ {**schema-name** : < *vobj* > }
fop3 ≡ {**instance** < *vobj*1 >
previa-operacion : < *vobj* > }
; *el predicado del objeto de acción* **sig-op3** *impone la ligadura:*
< *vobj* > — *op4*
fop4 ≡ {**instance** < *vobj*2 >
previa-operacion : < *vobj* > }

Figura 5.3: Funciones *obj*, *fop3* y *fop4* de las relaciones de red del plan de proceso *P2*.

5.3.1 Representación de las restricciones impuestas por la solución

Cada operación de fabricación instanciada puede interpretarse como una especificación de restricciones sobre la ejecución. El estado del plan de operaciones se especifica manteniendo de forma incremental una representación de dichas restricciones, siguiendo un doble enfoque según la perspectiva de la orden y del recurso.

Así pues, un aspecto básico en la representación de las operaciones es la especificación de los recursos que han de ser utilizados durante su ejecución y sus restricciones temporales de operación. La descripción de una operación indica el conjunto de decisiones de asignación que son compatibles con las restricciones operativas de la factoría, así como otro tipo de decisiones de planificación que hayan sido adoptadas para otras operaciones.

En la operación descrita en la figura 5.4 se pueden observar atributos típicos utilizados en tareas de planificación. La interpretación de dicho objeto es inmediata. La operación *orden6-op4* es una instancia del prototipo *op4* y pertenece a la orden *orden6* debiendo realizarse en la máquina *M2*. La operación que le precede en *orden6* es *orden6-op0-P2* y la que le sigue *orden6-op5*. Su intervalo de ejecución estará restringido por los intervalos de las operaciones anteriores y en concreto, su tiempo de comienzo debe estar comprendido entre las 10:30 y las 15:30 del 26 de Noviembre y su tiempo de finalización entre las 12:00 y las 17:00 del mismo día. También se indica que la restricción de tiempo de finalización más tardío es una consecuencia de la decisión de planificación que se hizo concerniente a la operación *orden6-op5*, que se ejecutará a continuación, y la restricción temporal del tiempo de comienzo es una consecuencia del tiempo de liberación de la orden y de la no disponibilidad de *M2* debido a la asignación previa de otras operaciones. El estado de la operación es *no-planeada* por lo que no tiene asignado ningún intervalo temporal en el plan.

{*orden6-op4*
instance : op4
suboperacion-de : orden6-produce-P2
recurso-primario : M2
capacidad-requerida : 1
orden : orden6
siguiente-operacion : orden6-op5
previa-operacion : orden6-op0-P2
estado : noplaneado
intervalo-limite-tiempo :
{**instance** : intervalo-tiempo-calendario
tiempo-comienzo-mas-temprano : Nov 26 10:30
origenes : (fecha-liberacion-orden orden6)
(restriccion-capacidad M2)
tiempo-comienzo-mas-tardio : Nov 26 15:30
tiempo-finalizacion-mas-temprano : Nov 26 12:00
tiempo-finalizacion-mas-tardio : Nov 26 17:00
origenes : (tiempo-comienzo-planeado orden6-op5) }
tiempo-planeado :
{**instance** : intervalo-tiempo-calendario
tiempo-comienzo-planeado :
tiempo-finalizacion-planeado : }}

Figura 5.4: Operación todavía no planeada, con sus restricciones de límites de tiempo.

Si se considera ahora el punto de vista de los *recursos primarios*. Un atributo clave de cada recurso para propósitos de planificación es la especificación de su *disponibilidad* para su asignación a operaciones. La disponibilidad caracteriza intervalos de tiempo durante los cuales los recursos se encuentran disponibles para su asignación a operaciones, y en el caso de recursos agregados, describen su capacidad remanente.

Estas representaciones de las restricciones de límites de tiempo y capacidad componen la hipótesis del plan de operaciones y constituyen la pizarra de la arquitectura del sistema de planificación. Dichas restricciones han de reflejar consistentemente el estado del plan, para lo cual existen un conjunto de procesos de propagación que combinan las restricciones adicionales impuestas por la definición del modelo con las resultantes de anteriores decisiones.

5.4 Ciclo de Control del Planificador

El control de las actividades para la resolución del problema se realiza por medio de una fuente de conocimiento especializada denominada *gestor-alto-nivel* (figura 5.1). El gestor de alto nivel trabaja a modo de *fuente de conocimiento de control* (según la idea desarrollada en [HAYE 85]) y tiene dos objetivos fundamentales:

- Ejecuta el ciclo de control básico que es independiente del dominio.
- Gestiona el conocimiento de alto nivel específico del dominio que está referido al conocimiento acerca de las características de aplicabilidad de las fuentes de conocimiento.

La figura 5.5 muestra el ciclo de control básico del planificador. El ciclo de control puede verse como un bucle cerrado compuesto por varias etapas. Cuando el planificador es invocado, el ciclo de control se activa en caso de que exista algún evento de control en la lista de eventos pendientes. La selección de un evento implica la elección del punto donde se focalizará la actividad del planificador. Esto se materializa en la selección y ejecución de una fuente de conocimiento, que resolverá al menos parte del problema de planificación, pero que puede conducir a la creación de más eventos internos. El bucle finaliza cuando el conjunto de eventos, relevantes a la tarea del planificador, quede vacío. En los siguientes apartados se presenta un sumario más detallado los métodos componentes de este ciclo de control.

5.4.1 Control Dirigido por Eventos

Como ya se ha adelantado, la coordinación del esfuerzo del gestor de alto nivel se consigue a través de un proceso dirigido por eventos. Los eventos de control se

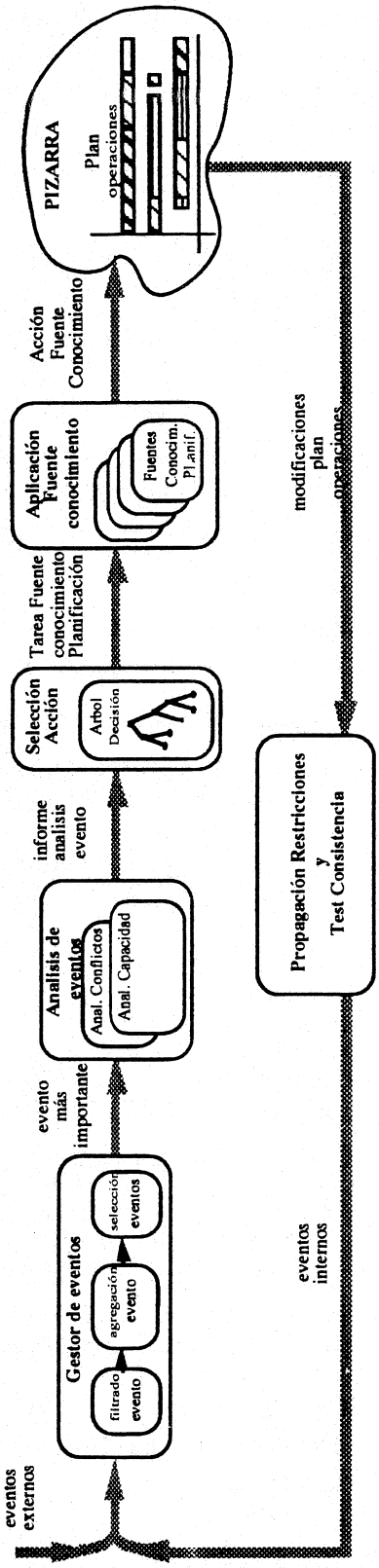

Figura 5.5: Ciclo de Control del Planificador.

envían para informar de cambios en el estado del plan de operaciones, introducidos internamente por la aplicación de fuentes de conocimiento, o generados externamente debido a actualizaciones del estado de la factoría reseñados por el módulo de monitorización (p.e. rotura de máquinas o retrasos en la finalización de operaciones). La coordinación de la actividad de planificación procede reactivamente en respuesta a los eventos de control enviados.

Los **eventos** se definen para representar aquellas consecuencias de acciones específicas de resolución del problema, relevantes a las decisiones de control, que deben ser realizadas en el ciclo de control del planificador. Los eventos se pueden generar por varias razones:

1. Como resultado de la ejecución de una tarea por parte de alguna fuente de conocimiento. Su objeto es que la fuente de conocimiento de control pueda conocer la traza de tareas realizadas en el actual proceso de resolución del problema.
2. Como resultado de violaciones de restricciones (temporales o de capacidad) detectadas por el subsistema de mantenimiento del plan de operaciones.
3. Como resultado de la adaptación del estado de la factoría a partir de la información recibida por el sistema de monitorización (p.e. fin de una operación de fabricación más tarde de lo previsto o rotura de una máquina).
4. Como resultado de cambios en las especificaciones introducidas a nivel superior de planificación (p.e. introducción de una nueva orden de trabajo).

Tipos de eventos

Se han definido tres grandes clases de eventos (en la figura 5.6 se muestra el árbol de especialización de eventos de control) atendiendo a la característica que plantean desde el punto de vista del plan de operaciones:

- **conflictos**: Describen situaciones relacionadas con conflictos entre restricciones. Se pueden especializar en tres clases atendiendo al tipo de restricciones involucradas, y a que el conflicto se refiera a un problema actual o a un problema previsto del plan de operaciones:
  - *inconsistencia*: Son conflictos entre restricciones no negociables (p.e. violación de precedencia entre dos operaciones de la misma orden) que indican que la solución corriente es inconsitente en algún aspecto y su modificación es imprescindible.
  - *compromiso*: Indican la violación o relajación de restricciones de preferencia. El problema no radica en la inconsistencia del plan de operaciones pero apunta una posiblilidad para aumentar su calidad.

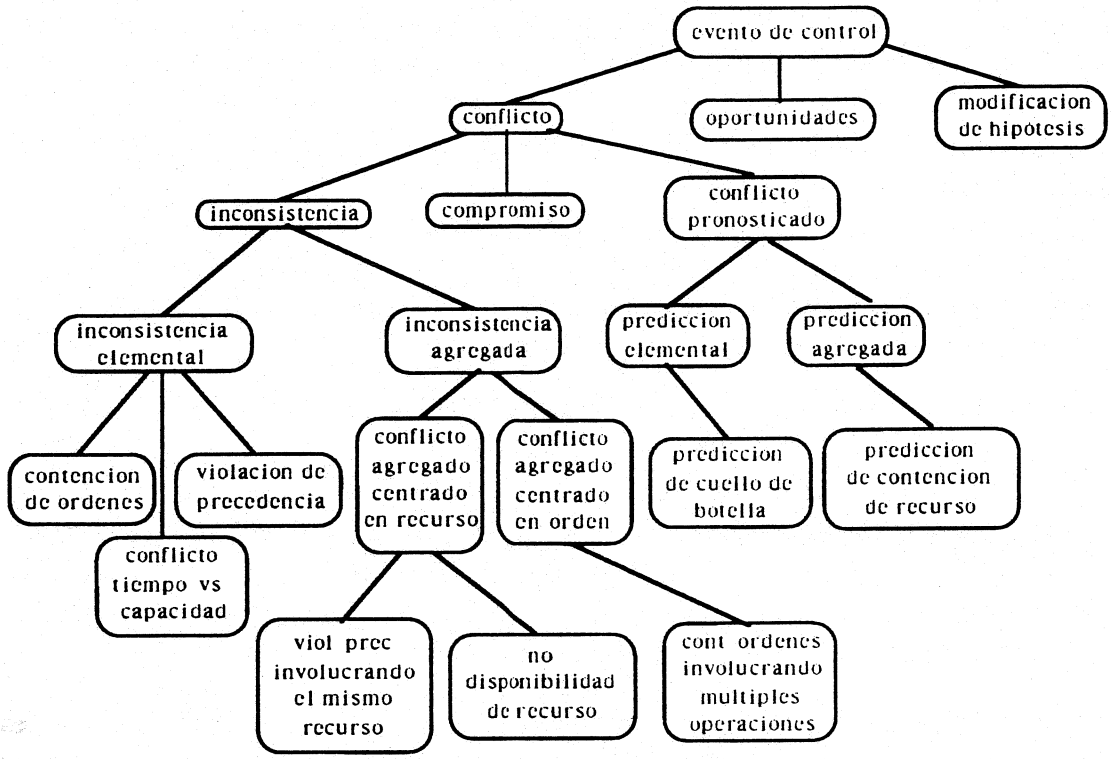

Figura 5.6: Jerarquía de especialización de eventos de control.

- *conflicto-pronosticado*: Se generan como resultado de análisis explícitos de porciones en desarrollo del espacio de la solución, e identifican componentes de la solución donde será necesario establecer compromisos entre restricciones. Proporcionan información para las decisiones de control estratégicas que afectan a la descomposición del problema de planificación y a la elección de la zona sobre la que dirigir la actividad del planificador.

- **oportunidades**: Describen situaciones donde han sido eliminadas una o más restricciones impuestas previamente, que posibilitan la existencia de oportunidades para mejorar el plan de operaciones (p.e. si una máquina rota pasa a encontrarse disponible antes de lo previsto, el planificador puede aprovechar la ventaja de la capacidad adicional disponible para mejorar el plan de operaciones).

- **modificación de hipótesis**: Indica modificaciones de hipótesis particulares (p.e. introducción de una nueva orden).

Aunque aquí se proponen una diversidad de tipos de eventos de control, sólo los de tipo *inconsistencia* indican violaciones de restricciones no ralajables, que han de ser resueltas para obtener un plan de operaciones consistente. Así, en los siguientes

apartados se hará énfasis en esta clase de eventos (conflictos que llevan asociados inconsistencias), relegando fuera del alcance de este trabajo el tratamiento específico del resto de eventos de control.

5.4.2 Selección del evento

El módulo de selección del evento (véase la figura 5.5) trata de identificar el evento de control más prometedor, que proporcione la dirección más oportuna sobre la que orientar la actividad de resolución del problema.

Este proceso de selección se puede descomponer funcionalmente en tres etapas:

- *Filtrado de eventos*: En ocasiones en las que el planificador ha sido invocado para una reacción limitada a consideraciones locales (p.e. un recurso particular durante un intervalo de tiempo limitado o una orden concreta), esta fase produce un efecto de filtrado, seleccionando únicamente los eventos directamente relacionados con dicho foco de interés.

- *Agregación de eventos*: Las inconsistencias surgidas en el plan de operaciones que han sido enviadas como eventos individuales, pueden estar relacionadas de alguna forma. Normalmente resulta más beneficioso abordar estos problemas de forma agregada (p.e. agregación de conflictos de capacidad individuales que involucran el mismo recurso). Para ello se dispone de unas heurísticas de agregación de eventos cuyo objetivo es la detección de tales situaciones y la transformación de dichos subconjuntos de eventos en eventos agregados simples.

- *Priorización de eventos*: Se aplican heurísticas de prioridad para seleccionar el evento más importante (evento que permitirá llegar a una solución "mejor"), sobre el cual se focalizará el esfuerzo del planificador en el corriente ciclo de control.

  Esta prioridad es, en primer lugar, una función del tipo de evento. Cada tipo de evento dispone de un valor por defecto (atributo *importancia-evento*) cuya magnitud indica su prioridad de tratamiento. Esto implica la siguiente política de control: con respecto a una hipótesis dada (o plan de operaciones parcial), los eventos enviados que posean diferentes valores de *importancia-evento* serán tratados según el orden de dichos valores. Según esta política, los eventos de tipo *inconsistencia* serán tratados antes que los eventos de tipo *compromiso* y éstos antes que los de tipo *conflicto-previsto* y a su vez éstos antes que los eventos de *modificacion-de-hipotesis*. En los casos apropiados puede imponerse un secuenciamiento de tipos de eventos más fino (p.e. inconsistencias agregadas se consideran más importantes que inconsistencias elementales).

Existe todavía un segundo nivel de diferenciación para aquellos eventos que resulten con la misma importancia base. En estos casos se sigue una política que atiende a las características específicas de cada evento. Para el cálculo de esta medida de importancia específica se dispone de un método (*calcula-importancia-especifica-evento*) que forma parte de la definición del objeto genérico *evento-control*.

5.4.3 Análisis del evento

El evento de control porta información acerca del tipo de problema puntual que identifica, así como de las entidades (recursos y operaciones) involucradas. Sin embargo, con objeto de facilitar la elección de las fuentes de conocimiento más adecuadas, resulta útil disponer de otro tipo de información adicional, que proporcione información acerca de la magnitud e implicaciones del problema originado por el evento. Esta información se consigue realizando un análisis que examine, con más detalle, la porción de plan de operaciones que corresponde al entorno del punto de interés y al conflicto mismo. El contexto del evento es pues analizado y caracterizado, y sus parámetros registrados en un **informe de análisis.**

Los parámetros de caracterización de los eventos de control dependen del tipo de evento y se examinarán detalladamente en el punto 5.6.

5.4.4 Selección de la acción

Una vez que el evento de control ha sido caracterizado, el gestor de alto nivel inicia un proceso de reconocimiento para encontrar la fuente de conocimiento más adecuada a aplicar así como sus parámetros de aplicación. El conocimiento heurístico de alto nivel a utilizar en este procedimiento decisorio se encuentra expresado a modo de **árbol de decisión**, de forma que los parámetros de caracterización de conflictos imponen la forma de navegación por las ramas del árbol (más adelante se examinará este extremo con más detalle).

La caracterización del evento de control, que ha sido proporcionada por el informe de análisis, es contrastada con los parámetros del árbol de decisión que representa la recopilación de las **heurísticas de control** de alto nivel. Estas heurísticas combinan información acerca del evento y su entorno con conocimiento relativo a la intensidad y fragilidad de las fuentes de conocimiento del planificador.

El propósito final de la fase de selección de acciones es la generación de la **tarea** apropiada, que habrá de ser enviada a la fuente de conocimiento elegida.

{*PREC-tarea-4582*
instance : tarea-planificador-recursos
receptor : planificador-recursos
asignador : gestor-alto-nivel
hipotesis : hip-5682
objetivo : refinar-decisiones
punto-de-enfoque : ct4b-test-cells
evento-de-disparo : prediccion-de-cuello-de-botella-57
operaciones : ct4b-main1-op33364 ct4b-main1-op33528
tiempo-de-ejecucion : 2556508
espacio-adicional-consumido : 87599}

Figura 5.7: Ejemplo de descripción de tarea.

5.4.5 Ejecución de la acción

Cada tarea designa una fuente de conocimiento particular, a aplicar a una hipótesis particular, de una forma particular. Cada fuente de conocimiento dispone de un método especializado con el que puede ejecutar sus tareas.

La aplicación de esta acción oportunista conduce a cambios en el plan de operaciones, que pueden introducir inconsistencias con otros componentes de dicho plan (debido a la no consideración de las otras perspectivas componentes del plan de operaciones completo).

La figura 5.7 muestra un caso particular de tarea enviada al *planificador de recursos*. Los atributos identifican información útil para la tarea, por ejemplo los parámetros utilizados por la fuente de conocimiento tales como el recurso sobre el que se centrará el esfuerzo de planificación (*ct4b-test-cells*) y las operaciones involucradas (en este caso *ct4b-main1-op33364* y *ct4b-main1-op33528*). También portan información sobre el problema (*hipótesis* y *evento* provocador de la tarea) y el *objetivo*. Finalmente existe otro tipo de datos útiles para las tareas de evaluación de experimentos como son el *tiempo-de-ejecucion* y el *espacio-adicional-consumido* de memoria.

5.4.6 Mantenimiento del plan de operaciones

El bucle de control se completa con un *sistema de mantenimiento de consistencia* del plan de operaciones, que explota las restricciones temporales y las limitaciones de capacidad especificadas en el modelo de la factoría para determinar las violaciones de restricciones (eventos de control de tipo conflicto) adicionales implicadas por cada nueva aplicación de una base de conocimiento del planificador.

El sistema de mantenimiento actualiza las descripciones sobre límites de tiempo asignados a operaciones y capacidad disponible de los recursos a lo largo del horizonte de tiempo. El proceso de mantenimiento es activado siempre que se realiza algún cambio y su propagación es llevada a cabo de un modo orientado a objetos. Con cada cambio, las descripciones de los objetos se actualizan para indicar los recursos elegidos y los intervalos de ejecución. Las restricciones son entonces comunicadas a los objetos de los recursos y de las operaciones involucrados. Las restricciones asociadas a dichos objetos son actualizadas y la propagación continua. La propagación ha de realizarse en dos dimensiones:

- *lateralmente* siguiendo el sentido de las relaciones de precedencia de las operaciones, y
- *verticalmente* a través de las jerarquías definidas de operaciones y recursos.

Esta propagación de restricciones puede conducir a la detección de dos tipos de conflictos (la figura 5.6 muestra su posición dentro de la jerarquía de especialización de eventos):

- **conflictos de tiempo**: describen situaciones en que los límites de tiempo, de dos operaciones que pertenecen a la misma orden, son inconsistentes (p.e. cuando una operación se prevé que comienze antes de que su anterior acabe).
- **conflictos de capacidad**: describen situaciones en que las reservas de operaciones hechas en un recurso exceden su capacidad sobre cierto intervalo de tiempo.

Para una descripción más completa acerca de la gestión de restricciones temporales y el mantenimiento del plan de operaciones, puede consultarse la referencia [LEPA 87].

5.5 Fuentes de Conocimiento del Planificador

Los planes de operaciones son analizados, generados y revisados incrementalmente a través del uso combinado de una colección de diferentes métodos de planificación o procedimientos heurísticos denominados fuentes de conocimiento. El objetivo de cada fuente de conocimiento es contribuir con nueva información para tratar de conducir a la solución del problema. Una fuente de conocimiento toma un conjunto de información de la pizarra y la actualiza según el conocimiento especializado que tiene codificado, generalmente en forma de reglas o procedimientos [ELGE 88].

Una característica bastante extendida en las arquitecturas de pizarra es que las propias fuentes de conocimiento sean responsables de conocer las condiciones bajo

las cuales han de ser aplicadas. Con este objetivo, cada fuente de conocimiento posee precondiciones que indican las condiciones que han de existir en la pizarra antes de que la fuente pueda ser activada. En la aproximación considerada en este trabajo se sigue un método ligeramente distinto, según el cual las precondiciones de todas las fuentes están agregadas y son controladas por el gestor de alto nivel, que es el responsable de decidir la fuente de conocimiento a aplicar.

Considerando su aspecto funcional, las fuentes de conocimiento del planificador de operaciones pueden ser agrupadas según dos tipos:

- **Acciones de planificación** - son fuentes de conocimiento cuya ejecución conlleva cambios en la hipótesis del plan de operaciones considerada.

- **Acciones de análisis** - son fuentes de conocimiento que implican la evaluación, recopilación o abstracción de la hipótesis considerada.

En los siguientes apartados se muestran algunos ejemplos concretos de fuentes de conocimiento cuya utilización ha resultado ser, en gran medida, genérica y satisfactoría.

5.5.1 Acciones de Planificación

Cada fuente de conocimiento de planificación varía en relación con los *tipos de problemas* que es capaz de resolver, por medio de sus diferentes capacidades relativas a la solución de conflictos de restricciones particulares, y con los *objetivos de optimización* sobre los que puede hacer énfasis.

Es bien conocida la dificultad de los sistemas de planificación para establecer métodos de resolución genéricos. Teniendo presente esta limitación, las acciones de planificación mencionadas más adelante tratan de conseguir cierto grado de generalidad, en el sentido de que abordan perspectivas de planificación de una forma abstracta. Su implementación podrá variar en general con la aplicación pero no así las situaciones adecuadas para su aplicación. Con esto se trata de conseguir que el conocimiento de alto nivel, que define su aplicabilidad, sea relativamente invariante del sistema específico de aplicación.

En el planificador, como en toda herramienta de desarrollo, resulta necesaria la disponibilidad de una buena interfase con el diseñador. Es muy importante que el usuario tenga información del proceso de deducción que llevan a cabo las fuentes de conocimiento. La figura 5.8 muestra un ejemplo de esta interfase, la pantalla muestra información sobre la fuente de conocimiento aplicada, tarea que creó la activación de la fuente, información sobre el proceso de deducción y gráfico Gantt que ilustra la situación general del plan de operaciones.

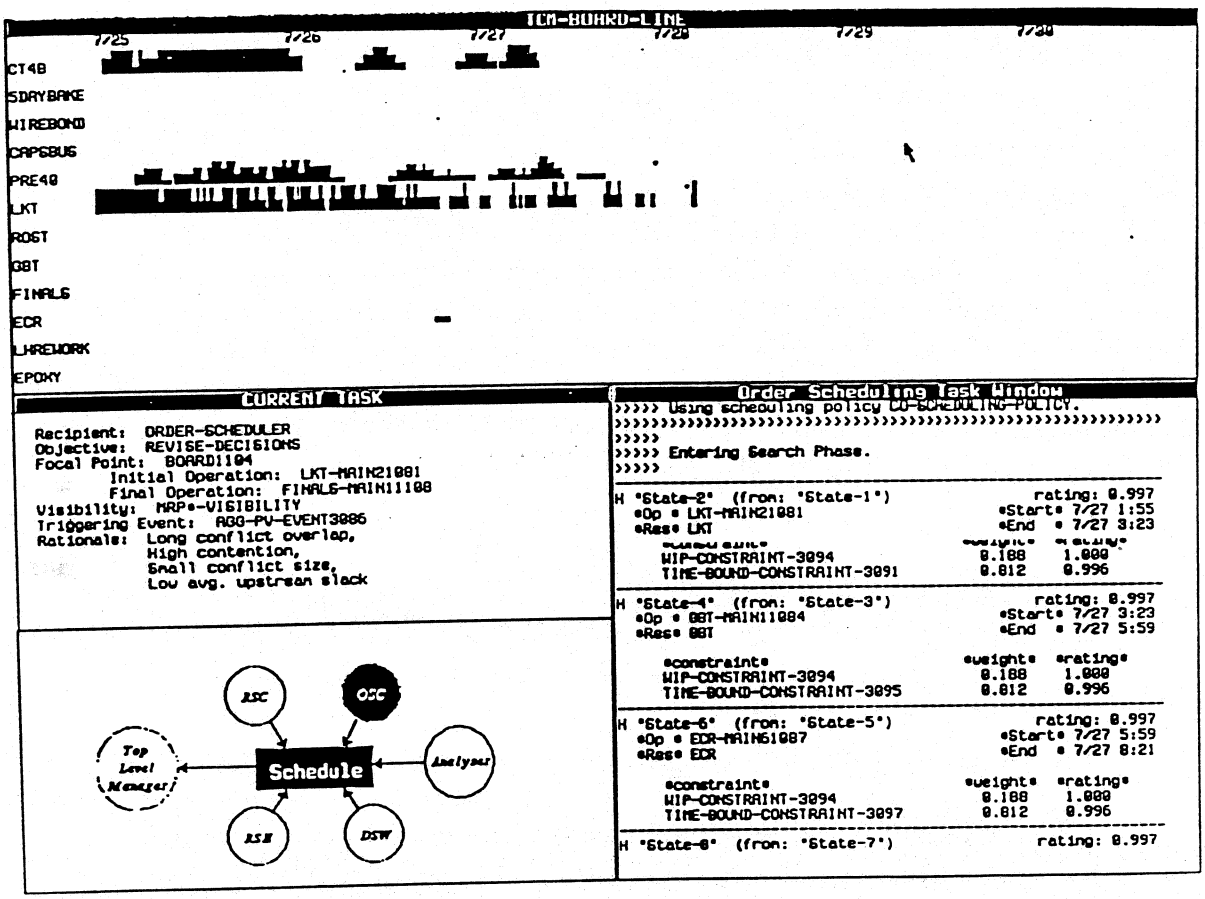

Figura 5.8: Ejmplo de interfase ded usuario del planificador de operaciones.

Fuente de conocimiento con perspectiva basada en las órdenes

El **planificador de órdenes**, PORD, proporciona un método para generar o revisar las decisiones de planificación relacionadas con alguna porción contigua del plan de producción correspondiente a una orden de trabajo específica.

Dado este enfoque, el PORD es adecuado para resolver conflictos centrados en las órdenes, fundamentalmente violaciones de precedencia entre operaciones correspondientes a una misma orden (que surgen, por ejemplo, al introducir operaciones de control de calidad imprevistas). Por la misma razón, este método resulta adecuado para satisfacer objetivos de producción centrados en las órdenes, tales como minimizar el trabajo en proceso, la tardanza media o ponderada, el número de órdenes tardias, o el coste de almacén, etc.

El PORD emplea técnicas de búsqueda dirigida por restricciones desarrolladas originariamente en el sistema ISIS[13]. Este método se caracteriza por el uso de una *búsqueda en rayo*[14] para explorar distintas alternativas con respecto a la bondad con la que las decisiones satisfacen las restricciones relevantes como puede ser el trabajo en proceso, la fecha debida (restricción de límite temporal), el coste almacén por acabado temprano de productos, preferencias máquinas, etc. Cada restricción tiene asociado conocimiento heurístico en forma de parámetros que ponderan su importancia con respecto a otras restricciones y su utilidad relativa con respecto a otras alternativas. El ciclo de ejecución del PORD consta de tres fases:

1. *Inicialización del proceso*: Establece las condiciones iniciales para la búsqueda para lo cual determina las operaciones involucradas y actualiza sus restricciones temporales retractandose de decisiones anteriores (en caso necesario). Finalmente define el estado inicial de la búsqueda.

2. *Proceso de búsqueda en rayo*: Hace uso de tres operadores de búsqueda para pasar a la siguiente operación, para expandir en subrecursos en caso de ser un recurso agregado, y para explotar las distintas posiciones de la operación en la cola de espera del recurso. La evaluación de cada estado de búsqueda está basada en restricciones, generalmente preferencias de tiempo de finalización y coste de almacén (evaluadas a partir de las restricciones temporales de las operaciones), preferencias relativas al trabajo en proceso y preferencias locales. La anchura del rayo de búsqueda aporta un grado adicional de flexibilidad[15].

3. *Asignación de recursos*: Las posibilidades resultantes se refinan para minimizar el trabajo en proceso.

---

[13]Más especificamente, los niveles de análisis y asignación de recurso de la arquitectura de búsqueda de ISIS.

[14]"Beam search" según su denominación inglesa.

[15]En [OW 84] se proponen 9 estados como anchura ideal para equilibrar fiabilidad y prestaciones.

Resulta útil la posibilidad de graduar el grado de visibilidad con que el PORD puede detectar las restricciones de disponibilidad de los recursos. Es decir, no tener en consideración las asignaciones del recurso reservadas para órdenes con inferior prioridad a la de la órden considerada, si bien en este caso se debe admitir la posibilidad de introducir conflictos adicionales al permitir una sobreasignación por encima del límite de disponibilidad de los recursos.

La estrategia de trabajo en modo reactivo consiste inicialmente en la retracción de las decisiones previas de la porción de orden indicada en la tarea, para pasar posteriormente a aplicar la estrategia de generación.

Fuente de conocimiento con perspectiva basada en los recursos

El **planificador de recursos**, PREC, proporciona un método para generar o revisar el plan de operaciones del recurso asigando en la tarea (típicamente un area de trabajo). Así, en contraste con el PORD, el PREC es adecuado para la solución de conflictos centrados en los recursos (p.e. violaciones de capacidad en un recurso).

Este método está basado en la asunción de que la contención del recurso en cuestión es alta y, por tanto, no hay necesidad de introducir tiempo débil[16] entre operaciones. Las decisiones de planificación son generadas mediante un proceso iterativo haciendo uso de una aproximanción de tipo depache (utiliza una colección de heurísticas de despache para proporcionar la sensibilidad a las distintas preferencias). Estas heurísticas son aplicadas en dos fases, en la primera se asignan órdenes a cada máquina para lo que se utilizan las reglas de minimización de puesta en marcha[17] y minimización de tardanza[18]. En la segunda fase se desarrollan las secuencias de órdenes para cada máquina para lo que se utilizan las reglas de fecha debida más temprana[19], mismo tiempo de puesta en marcha[20] y tiempo de espera[21].

En modo reactivo la estrategia utilizada trata de retraer el mínimo número posible de decisiones. Así se reconstruye incrementalmente el plan de operaciones del recurso hasta que puede fusionarse con el existente o hasta que los conflictos que originaron la tarea hayan sido resueltos.

Otro parámetro de trabajo en modo reactivo es el horizonte temporal a rehacer (fundamentalmente por razones de velocidad), lo que permite que el PREC pueda utilizarse también para ejecutar reacciones locales con un corto tiempo de respuesta.

---

[16] "Slack time" según su denominación inglesa.
[17] "Minimize setup rule".
[18] "Minimize tardiness rule".
[19] "Earliest due date rule".
[20] "Same setup rule".
[21] "Idle time rule".

Fuente de conocimiento para el desplazamiento de operaciones a la derecha

El **desplazador de operaciones a la derecha**, DOP, implementa un método reactivo mucho menos sofisticado que resuelve los conflictos "empujando" los tiempos de ejecución, previstos en el plan de operaciones, de las operaciones en conflicto, hacia adelante en el tiempo, en una cantidad que sea suficiente para evitar el conflicto. Estos desplazamientos pueden introducir, consecuentemente, nuevos conflictos de tiempo y capacidad, que son resueltos mediante nuevos desplazamientos.

El DOP parte de la asunción de que las restricciones de secuenciamiento de las operaciones permanecen todavía válidas.

Fuente de conocimiento de conmutador de demandas

El **conmutador de demandas**, CDEM, es un método reactivo especializado, aplicable en situaciones donde una operación ha resultado inesperadamente y signicativamente retardada (p.e. como resultado de una acción de reproceso por no superar un control de calidad). Implementa un intercambio de la porción de la órden afectada con la correspondiente porción de plan de operaciones de otra orden del mismo tipo con objeto de que se minimize la tardanza combinada de ambas. Esto tiene el efecto de redireccionar las dos órdenes de forma que cada una satisfaga la demanda respectiva de la otra.

La idea subyacente esta motivada por el deseo de explotar la flexibilidad de las restricciones temporales de otra orden para el común beneficio de las dos.

5.5.2 Acciones de Análisis

Las fuentes de conocimiento de análisis proporcionan un mecanismo para completar la información relativa a los eventos de control y el plan de operaciones mismo, que es utilizada para caracterizar los conflictos y los problemas en el plan de operaciones, lo que proporciona una base para la diferenciación en la aplicación potencial de las distintas acciones de planificación.

Se distinguen a continuación dos tipos de fuentes de conocimiento de análisis, especializadas en diferentes objetivos:

Fuente de conocimiento de análisis de capacidad

El **analizador de capacidad**, CAN, proporciona información acerca de la carga de trabajo del sistema de fabricación, identificando las áreas en las que previsiblemente los recursos tendrán una alta contención. En funcionamiento en modo generativo, esto proporciona una base para estructurar dinámicamente la tarea de planificación.

La estrategia utilizada es la de construir un plan de operaciones grosero que satisfaga los límites de tiempo iniciales de las operaciones. En situaciones en las que se precisa la elección entre varios recursos, se utiliza una *heurística de balanceo de l'inea* ("line balancing heuristic"). La demanda de capacidad reflejada por este plan de operaciones se compara con la capacidad real de los recursos agregados (computando las relaciones demanda/capacidad) y de esta forma se identifican los posibles *cuellos de botella.*

En contraste con las anteriores fuentes de conocimiento, el CAN opera siempre a un nivel agregado del modelo de la factoría.

Fuente de conocimiento de análisis de conflictos

El *analizador de conflictos*, CONAN, calcula una serie de medidas que caracterizan el conflicto mismo y otro conjunto de parámetros que caracterizan distintos grados de flexibilidad relativa a las restricciones de tiempo y capacidad en la porción de plan de operaciones que rodea el conflicto.

En el apartado 5.6 se presentan detalladamente los conceptos relativos a estos parámetros y se propone un método para su cálculo.

5.6 Caracterización de los conflictos de planificación

Los eventos proporcionan información relativa al tipo de problema detectado, pero no proveen datos cuantitativos relativos al alcance del problema. Para poder tomar decisiones a alto-nivel, relativas al tipo de acción de planificación a aplicar, es importante explotar el conocimiento concerniente con la flexibilidad de las restricciones de tiempo y capacidad involucradas, tanto localmente como en el entorno rodeando el área del plan de operaciones en conflicto. En [OW 88b] se apunta ya un primer intento en este sentido, donde se trata de identificar un conjunto de características más globales de los conflictos.

Se introduce en primer lugar la terminología que se utilizará a continuación para describir las medidas de caracterización de conflictos, para pasar seguidamente a

realizar la presentación de los distintos parámetros ya en base a la terminología introducida. Ahora bien, los parámetros que se presentan se limitarán a dos tipos de conflictos, de violación de precedencia y de contención de órdenes así como sus versiones agregadas, que si bien obviamente, no cubren el espectro completo de tipos de conflicto, si son los de más importancia y mayor frecuencia de aparición.

5.6.1 Terminología

Los conflictos, **C**, podrán pertenecer a uno de los siguientes cuatro tipos:

- **VP** : Violación de precedencia.
- **CO** : Contención de órdenes. Una *CO* simple representa cualquier sobreasignación de capacidad de un recurso sobre una porción contigua de intevalo temporal.
- **VP.Agr** : Violación de precedencia agregada.
- **CO.Agr** : Contención de órdenes agregada.

$$C \in \{VP, CO, VP.Agr, CO.Agr\}$$

Parámetros relativos a órdenes, operaciones y recursos:

- $\mathbf{R_C}$: capacidad del reucurso R
- $\mathbf{Cap_R}$: recurso de enfoque del conflicto C
- $\mathbf{El_C}$: conjunto de conflictos elementales componentes de los conflictos agregados, $C \in \{VP.Agr, CO.Agr\}$
- $\mathbf{TC_{CO}}$: tiempo de comienzo del conflicto CO
- $\mathbf{TF_{CO}}$: tiempo de finalización del conflicto CO
- $\mathbf{TCP_{Op}}$: tiempo de comienzo planeado para la operación Op
- $\mathbf{TFP_{Op}}$: tiempo de finalización planeado para la operación Op
- $\mathbf{TP_{Op}}$: tiempo de proceso de la operación Op
- $\mathbf{Plan_R}$: conjunto de operaciones planeados para el recurso R
- $\mathbf{OpDelante_{VP}}$: operación delantera en la violación de precedencia
- $\mathbf{OpDetras_{VP}}$: operación trasera en la violación de precedencia

- $\mathbf{Sig_{Op}}$: conjunto de operaciones que siguen a la operación Op
- $\mathbf{Confl_C}$: conjunto de operaciones directamente relacionadas con el conflicto C

  Para cada uno de los cuatro tipos de conflicto, este conjunto se calcula de la siguiente forma:

$$\mathbf{Confl_{VP}} = \{OpDelante_{VP}\}$$

$$\mathbf{Confl_{VP.Agr}} = \bigcup_{VP \in EL_{VP.Agr}} \{OpDelante_{VP}\} = \bigcup_{VP \in EL_{VP.Agr}} Confl_{VP}$$

$$\mathbf{Confl_{CO}} = \left\{Op \mid Op \in Plan_{Rec_{CO}} \wedge ([TCP_{Op}, TFP_{Op}] \cap [TC_{Op}, TF_{Op}] \neq 0)\right\}$$

$$\mathbf{Confl_{CO.Agr}} = \bigcup_{CO \in EL_{CO.Agr}} Confl_{CO}$$

- $\mathbf{TPMed_R}$: tiempo medio de proceso de las operaciones planeadas para el recurso R

$$= \frac{\sum_{Op \in Plan_R} TP_{Op}}{|Plan_R|}$$

- $\mathbf{Cont_{R,t}}$: asignación de unidades de capacidad del recurso R en el instante t

5.6.2 Caracterización local del conflicto

El primer conjunto de medidas se refieren a caracterizaciones locales del plan de operaciones del conflicto mismo:

- **Duración del conflicto**: Representa la extensión del conflicto y es una medida de la validez de mantener como punto de enfoque la parte de plan de operaciones del recurso involucrado.

  Se mide calculando la duración del conflicto relativa al tiempo de proceso de las operaciones involucradas.

  $\mathbf{Dur_C}$: duración del conflicto C

$$\mathbf{Dur_{VP}} = \frac{[TFP_{OpDetras_{VP}} - TCP_{OpDelante_{VP}}]}{TPMed_{Res_{VP}}}$$

$$\mathbf{Dur_{VP.Agr}} = \max_{VP \in El_{VP.Agr}} Dur_{VP}$$

$$\mathbf{Dur_{CO}} = \frac{[TC_{CO} - TF_{CO}]}{TPMed_{Res_{CO}}}$$

$$\mathbf{Dur_{CO.Agr}} = \max_{CO \in El_{CO.Agr}} Dur_{CO}$$

- **Tamaño del conflicto**: Está representado por el número de operaciones en conflicto, es un indicador de la validez de las decisiones en el punto de enfoque.

  $\mathbf{Tam_C}$: tamaño del conflicto C

  $\mathbf{OpsAReplan_C}$: operaciones a replanificar del conflicto C

$$\mathbf{Tam_{VP}} = \left|OpsAReplan_{VP}\right| = \left|OpDelante_{VP}\right| = 1$$

$$\mathbf{Tam_{VP.Agr}} = \left| \bigcup_{VP \in El_{VP.Agr}} OpsAReplan_{VP} \right| = \left|El_{VP.Agr}\right|$$

$$\mathbf{Tam_{CO}} = \left|OpsAReplan_{CO}\right| = \left|Confl_{CO}\right|$$

$$\mathbf{Tam_{CO.Agr}} = \left| \bigcup_{CO \in El_{CO.Agr}} OpsAReplan_{CO} \right| = \sum_{CO \in El_{CO.Agr}} Tam_{CO}$$

- **Contención del conflicto**: Representa la contención relativa del recurso en el punto de enfoque, indica las posibilidades de solución restringiendose a consideraciones locales del recurso.

  Se calcula estableciendo la ratio entre la máxima capacidad requerida del recurso durante el tiempo del conflicto y su capacidad.

  $\mathbf{Cont_C}$: contención del conflicto C

$$\mathbf{Cont_{VP}} = \frac{\max_{t \in [TFP_{OpDelante_{VP}}, TFP_{OpDetras_{VP}} + TPMed_{Rec_{VP}}]} \{Cont_{Rec_{VP},t}\}}{Cap_{Rec_{VP}}}$$

$$\mathbf{Cont_{CO}} = \frac{\max_{t \in [TC_{CO}, TF_{CO}]} \{Cont_{Rec_{CO},t}\}}{Cap_{Rec_{CO}}}$$

$$\mathbf{Cont_{VP.Agr}} = \max_{VP \in VP.Agr} Cont_{VP}$$

$$\mathbf{Cont_{CO.Agr}} = \max_{CO \in El_{CO.Agr}} Cont_{CO}$$

5.6.3 Caracterización del área de plan de operaciones que rodea el conflicto

El segundo conjunto de medidas se refieren a caracterizaciones que involucran a la flexibilidad desde el punto de vista de la orden y las restricciones de capacidad de los recursos en las areas que rodean la zona del plan de operaciones en conflicto.

Para la definición de los parámetros que se enuncian a continuación, considerese una estructura jerárquica de operaciones donde SOp es una operación a cierto nivel alto de abstracción y $Op_1, Op_2, \ldots Op_n$ una secuencia esperada, ordenada y completa de suboperaciones de SOp al siguiente nivel inferior de abstracción.

- **Espacio hacia delante**: Trata de identificar espacios vacíos hacia delante en el plan de operaciones. Es un indicador de la flexibilidad para modificar los tiempos de terminación de las órdenes del recurso que constituye el punto de enfoque. Esta medida asume que en un buen plan de operaciones sólo hay tiempos de cola antes de los recursos que son, durante algún tiempo, cuello de botella.

  Este parámetro se mide calculando la media de las duraciones del primer tiempo de espera debido a una indisponibilidad del recurso, para todas las operaciones a efectuar en dicho recurso durante el horizonte temporal que se prolongue el conflicto.

  El *espacio hacia delante local*, $EDelL_{OP_i}$, de una operación Op_i, se define con respecto a su siguiente operación en la secuencia de la forma:

$$EDelL_{OP_i} = TCP_{Op_{i+1}} - TFP_{Op_i}$$

  y se define el *espacio hacia delante completo*, $EDel_{Op_j}$, de una operación Op_j, como el primer $EDelL_{Op_i} > 0$ siguiendo la secuencia de operaciones a continuación de Op_j:

$$EDel_{Op_j} = \begin{cases} EDelL_{Op_i}, & \text{si } \exists i \in [j \ldots n-1] \text{ para } \min i \text{ con } EDelL_{Op_i} > 0 \\ 0, & \text{en otro caso} \end{cases}$$

  EspacioDelante$_C$: Espacio hacia adelante en el conflicto C

$$EspacioDelante_C = \sum_{Op \in Confl_C} \frac{EDel_{Op}}{|Confl_C| * TPMed_{Rec_C}}$$

- **Retraso previsto**[22]: Es una medida del retraso que llevan ya las órdenes. Indica la flexibilidad de las restricciones de tiempo impuestas por la propia orden.

[22] "Projected lateness" según su denominación inglesa.

Se mide por la diferencia entre el tiempo más temprano a que la orden puede llegar al recurso cuello de botella más próximo (tiempo de finalización de la órden si no hay recurso que resulte cuello de botella) y el tiempo de comienzo planeado (fecha debida en caso de no haber cuello de botella).

El *retraso previsto local*, $RPrevL_{OP_i}$, de una operación Op_i, se define con respecto a su siguiente operación en la secuencia de la forma:

$$RPrevL_{OP_i} = \begin{cases} -(TCP_{Op_{i+1}} - TFP_{Op_i}), & \text{para } i \in [1 \ldots n-1] \\ -(TFPNS_{Opi} - TFP_{Op_n}), & \text{siendo } Op_n \text{ la última operación} \\ & \text{de la secuencia} \end{cases}$$

siendo $TFPNS_{Opi}$ el tiempo de finalización planeado en el nivel de abstracción superior, que se define como:

$$TFPNS_{Opi} = \begin{cases} FD_{SOp}, & \text{si } SOp \text{ corresponde con el máximo nivel de} \\ & \text{abstracción} \\ TFP_{SOp}, & \text{en otro caso} \end{cases}$$

donde FD_C identifica la fecha debida de la orden de trabajo correspondiente a FD_{SOp}.

Se define el *retraso previsto completo*, $RPrev_{Op_j}$, de una operación Op_j, como el primer $RPrevL_{Op_i} < 0$ siguiendo la secuencia de operaciones a continuación de Op_j:

$$RPrev_{Op_j} = \begin{cases} RPrev_{Op_i}, & \text{si } \exists i \in [j \ldots n-1] \text{ para } \min i \text{ con } RPrev_{Op_i} < 0 \\ 0, & \text{en otro caso} \end{cases}$$

RetrasoPrevisto$_C$: Retraso previsto en el conflicto C

$$RetrasoPrevisto_C = \sum_{Op \in Confl_C} \frac{RPrev_{Op}}{|Confl_C| * TPMed_{Rec_C}}$$

- **Espacio detrás**: Es una medida de la flexibilidad para desplazar hacia atrás los tiempos de comienzo de las operaciones.

  EspacioDetras$_C$: Espacio hacia atrás en el conflicto C

$$EspacioDetras_C = \sum_{Op \in Confl_C} \frac{EDetr_{Op}}{|Plan_{Rec_C}| * TPMed_{Rec_C}}$$

donde $EDetr_{Op_i}$ es el espacio detrás de la operación Op_i que se define como:

$$EDetr_{Op_i} = \begin{cases} \max(TCP_{Op_i} - TFP_{Op_{i-1}}, 0), & \text{si } Op_i \text{ ha sido planeada} \\ 0, & \text{en otro caso} \end{cases}$$

- **Cuello de botella hacia adelante**: Indica la existencia de un recurso cuello de botella para alguna de las órdenes en conflicto.

  CuelloBotAbajo$_C$: Recurso cuello de botella hacia abajo del conflicto C

$$CuelloBotAbajo_C = \begin{cases} si, & \text{si } \exists Op \in Confl_C \wedge EDel_{Op} > 0; \\ no, & \text{en otro caso} \end{cases}$$

  siendo $EDel_{Op}$ el espacio hacia delante completo para la operación Op definido anteriormente.

5.7 Meta-Conocimiento del Planificador

La especificación de un plan de resolución en la arquitectura aquí presentada, requiere la definición de las relaciones entre el conjunto de conflictos que pueden originarse durante el proceso de resolución del problema, y el conjunto de métodos de planificación, dictando qué acción de planificación (fuente de conocimiento) deberá elegirse, para cualquier estado dado en el proceso de resolución del problema.

Con el objeto de obtener y validar este conocimiento, se han realizado experiencias con un modelo de sistema de fabricación de componentes de computador, que la compañía IBM tiene en Poughkeepsie, NY (USA). El modelo y la validación del conocimiento serán tratados con más detalle posteriormente.

En el siguiente apartado se explica el método de implementación elegido, para pasar a continuación a detallar el conocimiento heurístico encontrado como resultado de las experimentaciones, y para finalizar se realizarán algunas matizaciones aclarativos acerca del árbol de decisión propuesto.

5.7.1 Implentación del conocimiento de alto nivel del planificador

Durante la experimentación con el modelo se han identificado un conjunto de meta-reglas empíricas de control, que han mostrado experimentalmente un "buen" comportamiento en respuesta a los conflictos simulados en diferentes contextos. Dicho meta-conocimiento se ha integrado en el módulo de planificación a modo de reglas precondición/postcondición.

Para la implementación se ha hecho uso de la representación de reglas de producción disponibles en el lenguaje *OPS5* (provisto en el entorno *KnowledgeCraft*). Dada la sencillez de las reglas no resulta estrictamente necesario utilizar una implementación en un lenguaje de la potencia de OPS5. Sin embargo, se ha hecho uso de las facilidades para prototipado rápido, proporcionadas por los entornos de desarrollo de IA, entre las que destaca la disponibilidad de lenguajes basados

```
( p selecciona-POR-VP-baja-contencion :context :context
    ( goal ^ instance > ()
           ^ schema-name selection-goal
           ^ status active )
    ( informe-analisis-conflicto
           ^ instance <> ()
           ^ schema-name < ar >
           ^ status current
           ^ duracion-conflicto >= 1
           ^ contencion < 1
           ^ tardanza-proyectada-corr > 0
           ^ demandas-intercambiables () )
— >
    ( devuelve-fc-elegida $< ar > POR-VP
          "Gran sobreposicion en conflicto, % 14T
          Baja contencion, % 14T
          Retraso proyectado positivo, % 14T
          Sin oportunidades para IND" ) )
```

Figura 5.9: Expresión de una meta-regla del árbol de decisión en OPS5.

en reglas. Entre las ventajas que proporciona la utilización de OPS5 habría que destacar la facilidad de diseño y ampliación de la base de meta-conocimiento de control (característica importante en toda herramienta de desarrollo). En segundo lugar, la disponibilidad de un mecanismo de control de un lenguaje basado en reglas evita la necesidad de programación de los métodos y algoritmos para realizar las funciones de reconocimiento y ligadura necesarias en el proceso de sensibilización y disparo de reglas. Por otra parte, la eficiencia del mecanismo de control de OPS5 está ampliamente reconocida.

La figura 5.9 muestra la implementación de una meta-regla típica, en este caso correspondiente a la selección del módulo POR-VP (planificador de órdenes con visibilidad parcial), para el caso de baja contención. Como puede observarse en la figura, la regla hace uso de los parámetros de caracterización del conflicto proporcionados por el *informe de análisis de conflictos.* El texto a imprimir proporciona información, al sistema de interfase interactivo, acerca de las características del conflicto y la acción de planificación elegida.

5.7.2 Heurísticas del meta-conocimiento de control

El metaconocimiento de control del módulo de planificación contiene el árbol de decisión que refleja el conocimiento heurístico acerca de la aplicabilidad de las fuentes de conocimiento, utilizadas en el contexto de *eventos de conflicto.*

A continuación se especifica esquemáticamente este meta-conocimiento. Los *parámetros de diferenciación* indican características definidas en el apartado 5.6. Los *valores* indican apreciaciones cualitativas de los anteriores parámetros. En la presente implementación, la apreciación cualitativa de los valores está dada en base a umbrales. La utilización de su caracterización utilizando conjuntos borrosos[23] aparece como una extensión natural de esta especificación. La *implicación* indica alguna característica relevante del plan de operaciones, resultado de la diferenciación. En el punto *acción indicada* se proponen las acciones que han resultado más adecuadas en los experimentos.

Parámetros de diferenciación: Dur_C y Tam_C

Valores: Dur_C y Tam_C PEQUEÑOS
Implicación: Secuencia válida
Acción indicada: **DOP**

Valores: EN OTRO CASO
Implicación: Resecuenciamiento
Parámetro de diferenciación: $PerfilCapacidad_{Rec_C}$

Valor: BAJO
Implicación: Perspectiva basada en los recursos
Parámetro de diferenciación: $RetrasoPrevisto_C$

Valor: POSITIVA ALTA
Implicación: Oportunidad de conmutar las demandas de las órdenes
Parámetro de diferenciación: Tam_C

Valor: GRANDE
Implicación: Necesidad de resecuenciamiento
Acciones indicadas: **CDEM , PREC**

Valor: PEQUEÑO
Parámetro de diferenciación: $EspacioDelante_C$

Valor: ALTO
Implicación: Existencia de flexibilidad para el resecuenciamiento

[23] "Fuzzy sets" según su denominación inglesa.

Acciones indicadas: **CDEM , PREC**

Valor: BAJO
Implicación: No existencia de flexibilidad para el resecuenciamiento
Acciones indicadas: **CDEM , PREC , POR-VP**

Valor: EN OTRO CASO
Implicación: Sin oportunidad de conmutar las demandas de las órdenes
Parámetro de diferenciación: Tam_C

Valor: GRANDE
Implicación: Necesidad de resecuenciamiento
Acción indicada: **PREC**

Valor: PEQUEÑO
Parámetro de diferenciación: $EspacioDelante_C$

Valor: ALTO
Implicación: Flexibilidad para el resecuenciamiento
Acción indicada: **PREC**

Valor: PEQUEÑO
Implicación: Sin flexibilidad para el resecuenciamiento
Acciones indicadas: **PREC , POR-VP**

Valor: ALTO
Implicación: Perspectiva basada en las órdenes
Parámetro de diferenciación: $RetrasoPrevisto_C$

Valor: NEGATIVO
Implicación: Ordenes pronto
Acción indicada: **POR-VC**

Valor: POSITIVO
Implicación: Ordenes tarde
Parámetro de diferenciación: $Varianza(RPrev_{Op}) \big| Op \in Confl_C$

Valor: ALTA
Implicación: Oportunidad de conmutar demandas
Acciones indicadas: **CDEM , POR-VP**

Valor: BAJA
Implicación: Sin oportunidad de conmutar demandas
Acción indicada: **POR-VP**

Como puede comprobarse en el esquema anterior, es interesante la aplicación del módulo de acción DOP cuando la duración del conflicto es poco importante. Esta decisión proviene del hecho de que el *desplazador de operaciones* mantiene la estabilidad de la secuencia de operaciones al nivel local del recurso. Esta heurística refleja la implicación de que en dicha situación todav'ia permanecen válidas las decisiones de secuenciamiento del corriente plan de operaciones. Esta fuente de conocimiento resuelve efectivamente conflictos de forma que se preservan las decisiones de secuenciamiento ya existentes.

Por contra, en casos en que el tamaño o la duración del conflicto son importantes, es necesario proceder a una reoptimización del plan de operaciones en la vecindad del conflicto. Si el conflicto está localizado a nivel de un recurso cuyo plan de operaciones está poco fragmentado (p.e. en caso de un recurso que constituya un cuello de botella), es necesaria una optimización centrada en este recurso a fin de asegurar la compactación para el cálculo eficaz de las restricciones ligadas a dicho recurso. Por otra parte, si el conflicto se localiza al nivel de un recurso donde el plan de operaciones está fragmentado, existe la oportunidad de realizar una optimización más eficiente considerando una perspectiva centrada en las órdenes. Una vez se ha adoptado la perspectiva conveniente, es necesario tener en cuenta las características adicionales del plan de operaciones a fín de seleccionar un módulo de acción adecuado.

Una consideración importante que no queda reflejada en la información relativa al metaconocimiento mostrada anteriormente, se refiere a la extensión de la revisión del plan de operaciones. En el caso de una revisión centrada en el recurso, la extensión de la revisión alcanza al plan de operaciones de producción local al recurso. En el caso de una revisión centrada en la orden, la extensión de la revisión depende de la presencia o no de de cuellos de botella hacia adelante. Más precisamente, si existe tal cuello de botella, la revisión para una orden particular se limita a la porción de operaciones de fabricación previas a la operación a realizar en el recurso que constituye el cuello de botella. Una aproximación de este tipo permite asumir después una perspectiva basada en el recurso a fin de replanificar los cuellos de botella situados más adelante.

5.8 Conclusiones

En este capítulo se ha abordado la problemática del planificador de operaciones. El método de planificación propuesto parte de los trabajos desarrollados en el sistema OPIS al que se le ha ido dotando de nuevas posibilidades.

En primer lugar se ha modificado la arquitectura del planificador y se ha propuesto un nuevo ciclo cerrado de control.

En segundo lugar se ha considerado insuficiente la información portada por los eventos, por lo que se ha dado énfasis a una etapa intermedia de análisis. En esta etapa se evalúan algunas características del evento y su entorno, para poder proceder a una elección más informada de la estrategia de planificación a realizar.

Para la caracterización de conflictos se proponen dos tipos de medidas: las primeras informan acerca de características locales (duración, tamaño y contención del conflicto) y las segundas dan información sobre el area del plan de operaciones que reodea el conflicto (espacio hacia delante, retraso previsto, espacio atrás y cuello de botella hacia delante). Siguiendo esta caracterización, se propone una serie de reglas heurísticas que permiten elegir "una adecuada" fuente de conocimiento a aplicar para los eventos conflicto. Para realizar este proceso de deducción se propone una implementación mediante reglas de OPS5. Un objetivo del planificador que se propone, es el servir como herramienta de desarrollo de estrategias de planificación y, de esta forma, facilitar la etapa de diseño con una herramienta de prototipado rápido.

EL ALMUERZO

No sin trabajo un cronopio llegó a establecer un termómetro de vidas. Algo entre termómetro y topómetro, entre fichero y curriculum vitae.

Por ejemplo, el cronopio en su casa recibía a un fama, una esperaza y un profesor de lenguas. Aplicando sus descubrimientos estableció que el fama era infra-vida, la esperanza para-vida, y el profesor de lenguas inter-vida. En cuanto al cronopio mismo, se consideraba ligeramente super-vida, pero más por poesía que por verdad.

A la hora del almuerzo este cronopio gozaba en oír hablar a sus contertulios, porque todos creían estar refiriéndose a las mismas cosas y no era así. La inter-vida manejaba abstracciones tales como espíritu y conciencia, que la para-vida escuchaba como quien oye llover -tarea delicada. Por supuesto la infra-vida pedía a cada instante el queso rallado, y la super-vida trinchaba el pollo en cuarenta y dos movimientos, método Stanley Fitzsimmons. A los postres las vidas se saludaban y se iban a sus ocupaciones, y en la mesa quedaban solamente pedacitos sueltos de la muerte.

Historias de cronopios y famas
JULIO CORTAZAR

Capítulo 6

Distribuidor de Operaciones y Sistema de Monitorización

La función del *distribuidor de operaciones*[1] es dar respuesta, en "tiempo real", a los problemas de decisión que surgen durante el control de un sistema de fabricación. Los problemas de decisión toman, al nivel del modelo de control, la forma de conflictos. Cada vez que el controlador detecta un conflicto, se plantea su solución aplicando una política de control. En este contexto, el distribuidor de operaciones constituye el marco donde son ejecutadas las políticas de control, que no son más que una llamada parametrizada al mencionado distribuidor. La arquitectura del distribuidor está inspirada en la del planificador de operaciones disponiendo, eso sí, de una gran flexibilidad para modificar su ciclo de control. De esta forma la política de control puede consistir únicamente en una simple regla heurística de despacho o, en el otro extremo, se procede a realizar una elaborada interpretación del plan de operaciones generado por el planificador y provocar, en su caso, una reacción para actualizar dicho plan. La política de control puede construirse modularmente utilizando la biblioteca de fuentes de conocimiento de que dispone el distribuidor.

Por otra parte, el sistema de monitorización proporciona una realimentación al resto de funciones del sistema de control, dando soporte de esta forma, al proceso de toma de decisiones. Esta realimentación consiste tanto en actualizar la información de la base de conocimiento para que refleje los hechos que están sucediendo realmente en la planta, como en detectar posibles contingencias para informar tanto al sistema de coordinación como al sistema de decisión creando y enviando los eventos de control pertinentes.

[1] Más conocido en la literatura como "dispatcher".

6.1 Toma de Decisiones en Tiempo de Ejecución: Distribuidor de Operaciones

Dentro de la arquitectura global de control propuesta, el distribuidor de operaciones constituye el módulo que realiza la interfase entre el sistema de coordinación y el sistema de toma de decisiones. El problema de control es asegurar que los volúmenes de producción, requeridos por el nivel superior, son conseguidos efectivamente. El propósito de este nivel consiste en proporcionar decisiones que optimicen localmente los objetivos de producción respetando las restricciones del sistema.

Entre los problemas característicos de este nivel destacan [GERS 86]: a) el movimiento de piezas a máquinas de forma que se reduzca el tiempo perdido, b) la elección de tiempos para la entrada de piezas en cada subsistema, c) la elección de la secuencia de máquinas que debe visitar una pieza, d) la elección de tiempos en los que las piezas deberán visitar las máquinas, etc. Un problema adicional importante es limitar el efecto de disrupciones en la operación de la planta, que no pueden ser especificadas con antelación pero son inevitables.

Entre los objetivos de producción a considerar en este nivel están alcanzar los tiempos de terminación exigidos, minimizar el inventario en proceso y maximizar la utilización de recursos. El punto clave en la resolución de estos problemas es la asignación de recursos (de fabricación, materiales, personal, transporte, almacenamiento, etc.) de manera eficiente teniendo en cuenta restricciones de objetivos (fecha debida, trabajo en proceso, costes, objetivos de utilización de máquinas), limitaciones físicas (capacidades de las máquinas, tamaño del producto, especificaciones de calidad), restricciones causales (relaciones de precedencia de operaciones y requerimientos de recursos), limitaciones de disponibilidad de recursos, preferencias, etc. [FOX 83,NEWM 88].

6.1.1 Algunas aproximaciones conocidas

La resolución de estos problemas se acomete fundamentalmente asignando prioridades a los trabajos, lo que se realiza por medio de un distribuidor ("dispatcher") o del propio supervisor de planta. Existen también otro tipo de sistemas cuya idea directriz es tratar de eliminar los trabajos en cola. Con ello, los problemas se relegan al momento de introducir cada trabajo en la planta y siguen políticas del tipo *justo-a-tiempo*[2].

Tradicionalmente, las aproximaciones utilizadas para abordar este nivel de control han estado basadas en la utilización de reglas de despacho [PANW 77, GRAV 81,RODA 88], bien aplicadas directamente para dar solución a los problemas de decisión, bien como reglas de decisión en la simulación para sistemas de

[2]"Just-in-time" (JIT) según su denominación inglesa.

planificación predictiva [LAW 86].

En los útimos tiempos se ha propuesto la utilización de técnicas de inteligencia artificial. En esta línea se han desarrollado algunos sistemas para control en "tiempo-real" [NEWM 88]. Cabría mencionar el sistema DISPATCHER [ACOC 86] basado en reglas e implementado en el lenguaje OPS5. El sistema se activa en respuesta a requerimientos de las estaciones de trabajo y selecciona la orden óptima. El sistema DEVISER es un ejemplo de aproximación dirigida por restricciones. Utiliza redes de operaciones parcialmente ordenadas en las que las actividades tienen asignadas ventanas de tiempo. La violación de estas restricciones provoca un encadenamiento hacia atrás en el proceso inferencial para buscar otra composición válida. Isenderg y col. [ISEN 88] proponen un modelo de simulación basado en el conocimiento para planificación de producción tiempo real (trilogía IA-simulación-reglas de despacho), que ha sido aplicado a industrias electrónicas. Destacan también los trabajos de Georgeff para control tiempo real de sistemas físicos complejos [GEOR 87]. Su campo de aplicación ha sido el control de operaciones espaciales. Georgeff apunta la simplicidad de los sistemas expertos convencionales para manejar los problemas de planificación y razonamiento tiempo real de dichas operaciones y propone un sistema basado en el concepto de *sistemas de razonamiento procedural.*

Existen asímismo trabajos que hacen uso de técnicas mixtas resultantes de combinar redes de Petri y técnicas de inteligencia artificial. En general utilizan la red de Petri para la representación del modelo y técnicas de representación y razonamiento de inteligencia artificial (normalmente basadas en reglas) para la inclusión de conocimiento específico relativo a la planificación de operaciones [PAGN 88,TAMU 89].

En los apartados siguientes se abordará el estudio del distribuidor de operaciones que se propone en nuestro esquema de control. En primer lugar se presenta el funcionamiento básico del subsistema de coordinación y la necesidad de dar solución a los conflictos que se originan en su evolución. Después de algunas consideraciones sobre la naturaleza del distribuidor, se pasará a desarrollar su arquitectura funcional y la construcción de sus bloques principales: restricciones, fuentes de conocimiento y políticas de control.

6.2 Funcionamiento del Sistema de Coordinación

La función básica del **sistema de coordinación** es gestionar y supervisar la ejecución de tareas por los controladores locales. El coordinador trabaja con un modelo del comportamiento de la planta, consistente en una red KRON. Su visión del estado de la planta en cualquier instante está definida por el marcado de la red. La evolución del estado de la planta está materializada por el disparo de los objetos

de acción.

Los objetos de acción de la red identifican tareas que deben realizarse a nivel de planta. Desde el punto de vista de su implementación, el disparo de un objeto de acción, con respecto a una ligadura, tiene asociada la ejecución de un procedimiento implementado por un protocolo de tipo "handshake" (consistente en el envío de una orden de ejecución de tarea a un controlador local y la recepción de un mensaje de finalización de tarea o mensaje de excepción por parte del controlador correspondiente). Estos procedimientos están configurados como tareas para permitir su ejecución concurrente. La gestión de estas tareas está soportada por el nucleo del sistema operativo.

La función de coordinación se materializa haciendo evolucionar la red KRON que representa el modelo del sistema de fabricación, lo que provoca la evolución del sistema en sí. Esto representa un control en bucle cerrado por el cual, el marcado de la red KRON sigue la evolución del estado del sistema de producción.

El coordinador puede recibir ciertos mensajes del sistema de monitorización informando de situaciones excepcionales. Estos mensajes son manejados por un submodulo especializado que puede alterar el estado de la red y abortar las tareas asociadas con el disparo de los objetos de acción.

6.2.1 Problemas de decisión en el distribuidor de operaciones

El sistema de coordinación trabaja con un modelo, una red KRON, que no es completamente determinista, y puede presentar, por tanto, cierto número de conflictos estructurales [SIL 85b]. Los conflictos estructurales plantean problemas de decisión referidos a cómo debería evolucionar el modelo (p.e. en qué momento debería dispararse un objeto de acción sensibilizado). Algunos de estos problemas han podido ser considerados previamente a la ejecución al producir el plan de operaciones en servicio, mientras que otros comprenden problemas a un nivel de toma de decisiones por debajo del reflejado en el plan de operaciones (p.e. en qué orden deberían cargarse dos productos que se encuentran esperando, en dos máquinas libres). Se pueden distinguir entre conflictos *relevantes* (con respecto al planificador de operaciones) y *no relevantes*. En cualquiera de los dos casos, se requiere la solución de estos problemas en tiempo real.

6.3 Principios de funcionamiento del Distribuidor de Operaciones

Como se ha mencionado con anterioridad, los problemas de decisión son resueltos aplicando políticas de control asociadas a los distintos conflictos. En la arquitectura de control que se propone en este trabajo, el distribuidor de operaciones constituye la parte del sistema de toma de decisiones que actua de interfase con el sistema de coordinación para dar solución a estos conflictos.

En el capítulo anterior se presentaron las características de un planificador de operaciones, que contemplaba la resolución del problema de planificación como una actividad dirigida por restricciones. Comenzando a un nivel alto de abstracción, la tarea del planificador consiste en un proceso incremental de propagación de restricciones, que comienza por una fase horizontal en la que se procede a la asignación de recursos e intervalos de tiempo a las diferentes operaciones. En este proceso oportunista de evaluación, se utilizan las restricciones (físicas, de objetivos, causales, de disponibilidad y de preferencia) como método heurístico para reducir el espacio de búsqueda. Estas asignaciones y los intervalos de tiempo reservados, constituyen nuevas restricciones a aplicar en los niveles inmediatamente inferiores de la jerarquía de abstracción del modelo, lo que constituye la propagación vertical. La metodología propuesta proporciona también una base para revisar incrementalmente el plan de operaciones en respuesta a cambios, no previstos, del entorno de producción.

Adoptando esta metodología, quedan todavía por resolver los problemas de decisión en tiempo de ejecución, relativos a la utilización del plan de operaciones, que son requeridos por el sistema de coordinación. Una estrategia extrema consistiría en dar total responsabilidad al planificador predictivo, requiriendo que cualquier discrepancia detectada entre el estado real del sistema de fabricación y el estado esperado, previsto por el plan de operaciones, provoque la revisión apropiada de dicho plan. Sin embargo, esto no resulta razonable en base a dos consideraciones:

- En la mayor parte de aplicaciones, el planificador predictivo debe operar necesariamente con valores aproximados de las restricciones temporales (p.e. duraciones de las operaciones y/o tiempos de puesta en marcha de los recursos), lo que provoca la aparición de algunas discrepancias entre el estado real de la factoría y el plan de operaciones. Tales discrepancias reflejan la "normal" impredectibilidad del sistema de producción. Adicionalmente, en ocasiones el planificador puede ser forzado llegando a producir decisiones sobre-restringidas[3]. En situaciones donde son necesarias asignaciones detalladas de recursos, para optimizar decisiones de secuenciamiento, puede ocurrir que únicamente las secuencias sean importantes. En general, la posibilidad de

[3] "Over constrained" según su denominación inglesa.

revisar el plan de operaciones debe ser considerada cuando exista alguna espectativa de que las discrepancias surgidas pudieran tener implicaciones más globales.

- Incluso si fuera posible el mantenimiento de un plan de operaciones predictivo al más alto nivel de detalle (lo que es improbable pero, en todo caso, depende de la aplicación específica), no tiene mucho sentido hacerlo. Multitud de decisiones de planificación a tomar estan restringidas a consideraciones locales (p.e. elección entre máquinas funcionalmente idénticas, secuencias de carga y descarga de productos, etc.) y no tienen consecuencias en el problema global de coordinación. El plan de operaciones predictivo debe operar a un nivel de detalle que sea suficiente para imponer una guía global permitiendo tanta flexibilidad en tiempo de ejecución como sea posible.

Teniendo en cuenta las anteriores consideraciones, el distribuidor de operaciones funciona a modo de sistema de decisión tiempo real conectando el sistema de coordinación y el de planificación. Su papel es interpretar el plan de operaciones de producción en el contexto de los problemas de decisión específicos que son enviados por el coordinador. Este proceso de interpretación está diseñado para compensar discrepancias menores entre el plan de operaciones y el estado actual de la planta y proporcionar soluciones al coordinador en tiempo real. Un efecto adicional de este proceso de interpretación es una mejor caracterización entre el grado de compatibilidad entre el plan de operaciones y el estado actual de la planta, que puede ser utilizado para determinar si está justificada la revisión del plan de operaciones.

6.3.1 Arquitectura y mecanismo de control del distribuidor

En el capítulo anterior se mostró como el proceso de resolución de problemas en el módulo del *planificador de operaciones* se ejecuta siguiendo, invariablemente, un ciclo cerrado de control. El mecanismo de control del distribuidor sigue, en parte, la misma filosofía de control presentada para el sistema de planificación de operaciones, pero adaptada para que cumpla los requerimientos de su funcionamiento como sistema de decisión con rápida respuesta:

- Para mejorar la **flexibilidad** de adaptación a diferentes estrategias, el ciclo debe poder ser readaptado para diferentes políticas de control. Esto se traduce en que algunas etapas del bucle sean opcionales.

- Para mejorar su **eficiencia**, la especificación de la política de control conlleva una selección de acciones, es decir, dispone como parámetros las fuentes de conocimiento útiles, especializadas en el problema de decisión que trata de

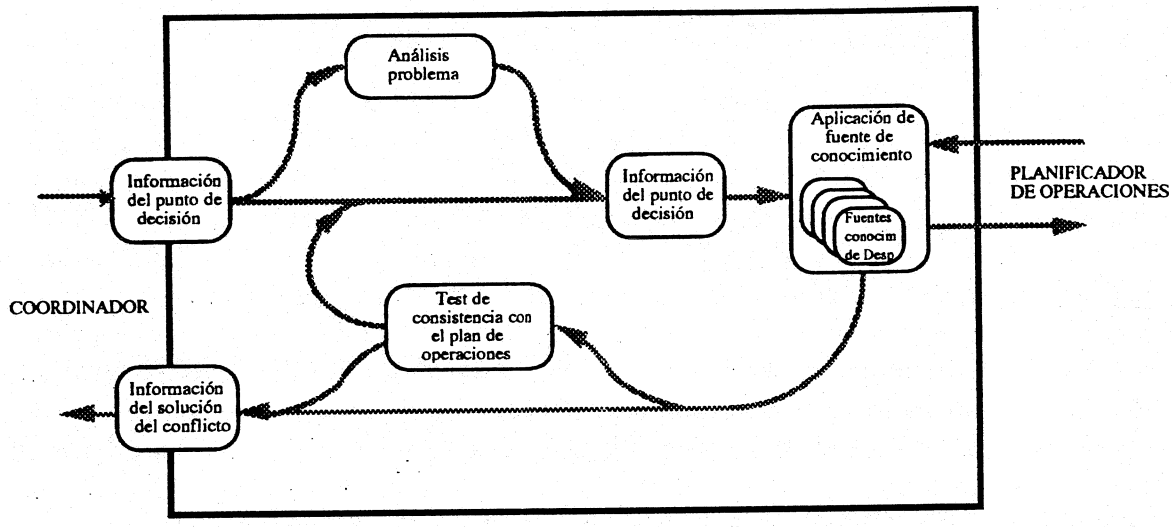

Figura 6.1: Bucle general de control del distribuidor.

resolver, por lo que no hay necesidad de comprobar la aplicabilidad del resto. La política de control puede considerarse como metaconocimiento asociado al punto de decisión, que especifica las fuentes de conocimiento más adecuadas para resolver sus problemas de decisión característicos.

La figura 6.1 muestra la estructura general del ciclo de control del distribuidor. El bucle de control general del distribuidor se convierte en el marco de trabajo sobre el que se construirán las políticas de control. De esta forma, la política de control indicará el tipo de ciclo de control del distribuidor y las fuentes de conocimiento a considerar.

El primer paso consiste en acomodar la información presente en el punto de decisión para crear un *requerimiento de decisión*, con objeto de que dicha información pueda ser reconocida y utilizada por el distribuidor.

Siguiendo la filosofía establecida en la construcción del planificador, el distribuidor dispone de la opción de realizar una etapa previa de análisis, con objeto de poder obtener una información más global acerca de las condiciones que rodean el punto de decisión. El análisis de los conflictos originados en los puntos de decisión es más simple que el mencionado para los conflictos del planificador debido a que se considera el estado real, por lo que no es necesario considerar efectos ni problemas en instantes anteriores. Los siguientes pasos consisten en la selección y aplicación de la fuente de conocimiento.

Para las fuentes de conocimiento del distribuidor que hacen uso de la información proporcionada por el planificador, existe la interesante posibilidad de ejecutar un procedimiento adicional que posibilita cerrar el bucle de realimentación del distribuidor, denominado *módulo de comprobación de consistencia* del distribuidor. El objetivo de este módulo es evaluar el nivel de compatibilidad entre el estado del

sistema de producción a corto plazo previsto por el planificador de operaciones y el nuevo estado producido por la decisión que es considerada.

En situaciones donde se encuentra algún grado de incompatibilidad, implicando que el estado del sistema de producción se desvía, de forma sensible, del estado producido por el planificador (la granularidad de esta estimación es especificada por la política de control), el bucle de control del distribuidor puede ser re-ejecutado de nuevo, buscando la aplicación de una acción más potente. La idea de esta estrategia consiste en explorar las consecuencias, a más largo término, de decisiones alternativas con objeto de realizar una elección más informada. Como se señaló anteriormente, un efecto de este proceso es obtener una mejor caracterización de la compatibilidad del plan de operaciones y el estado real del sistema de producción, lo que puede utilizarse para determinar la revisión del plan. Otra posibilidad para caracterizar el nivel de incompatibilidad, es verificar la lista de eventos pendientes. Las políticas de control pueden hacer uso de cualquiera de estas estimaciones cuando trabajen con un alto grado de acoplamiento con el plan de operaciones.

Cuando se completa este proceso de decisión, la información de la alternativa seleccionada es utilizada para resolver los conflictos surgidos al nivel de interpretación de la red.

6.4 Conocimiento Heurístico: Restricciones

Una parte importante del conocimiento acerca del sistema de fabricación consiste en la comprensión de las restricciones sobre la operación sistema, así como de las altenativas posibles a tomar cuando estas restricciones no pueden ser satisfechas. A Fox [FOX 83] se debe el primer planteamiento general del problema de planificación de operaciones en sistemas de fabricación flexible, como un problema de satisfacción de restricciones. Desde esta perspectiva, el problema puede verse como una variación del método de generación y test. El método comporta la existencia de un conjunto de operadores que definen el espacio de estados que representan previsiones de planes de operaciones parciales o totales, y existe conocimiento sobre el dominio que es utilizado para ponderar los estados dentro del espacio de búsqueda. El punto clave de esta aproximación es que este conocimiento puede verse como restricciones sobre el sistema.

La aproximación que se adopta para el distribuidor de operaciones, es similar a la adoptada para el planificador, si bien en este caso dotando al proceso de búsqueda de una mayor carga heurística. En este sentido, se sigue una reciente línea de trabajo que considera el método de resolución como una *búsqueda heurística restringida*[4] en la que se retienen las capacidades de síntesis de la búsqueda heurística y se extiende añadiendo las características estructurales de las técnicas de satisfación de

[4] "Constrained Heuristic Search" (CHS) según su denominación inglesa.

{*espec-relajacion*
espec-relajacion-de :
parametros :
fn-evaluacion : }

Figura 6.3: Objeto que describe la especificación de una relajación.

- *importancia* : define la importancia de la restricción en un rango continuo de 0 a 1, siendo 1 el más importante y 0 el menos importante.

- *limite* : mide el límite aceptable de utilidad de la restricción, por debajo del cual se considera que la restricción ha sido violada.

- *elasticidad* : define la pendiente de la recta de utilidad.

- *fn-evaluacion* : es un método que describe como debe ser evaluada la restricción.

La figura 6.3 muestra el objeto de especificación de relajación, la función de prioridad es un método dispuesto en el atributo de estado *fn-evaluacion* y *espec-relajacion-de* es una relación que conecta la especificación con el objeto de la restricción.

Desde la perspectiva de funciones de prioridad, las restricciones anteriores, consideradas individualmente, trabajarían como típicas reglas. Así, la aplicación de la restricción *fecha-debida* actuaría como la típica regla EDD[5] que da prioridad a las operaciones con menor fecha debida tratando de optimizar la *tardanza conjunta*[6].

Resumiendo, dada la utilidad de la relajación y su límite, se puede caracterizar las restricciones como *inflexibles* si la utilidad de la relajación es menor que el valor límite definido en el objeto de la restricción. En este caso la restricción no puede ser violada en ninguna circunstancia. Por tanto, es posible caracterizar una restricción como *flexible* o *inflexible* eligiendo los valores apropiados para el rango de las utilidades de la relajación y sus límites. Las restricciones de objetivos pueden ser combinadas como una forma de construir funciones de evaluación para obtener "buenas" soluciones respecto a varios criterios.

[5] "Earliest Due Date" según su acepción inglesa.
[6] "Tardiness" según su acepción inglesa.

6.4.3 Restricciones de preferencias

Las restricciones de preferencias son consideraciones heurísticas relacionadas con los planes de operaciones, y pueden considerarse, en algunos casos, como abstracciones de las anteriores. Así, una preferencia sobre una máquina para cierto tipo de operaciones puede ser una expresión del conocimiento del supervisor recogido a partir de su experiencia.

Las fuentes de conocimiento de distribuidor pueden operar con respecto a tres tipos de preferencias [MART 89], representadas declarativamente como restricciones relajables:

1. *Restricciones locales.* Definen preferencias subjetivas de tipo local, que pueden estar asociadas a las órdenes y recursos involucrados en el conflicto (p.e. prioridad de órdenes, preferencia de operaciones, fiabilidad de las máquinas, etc.).

2. *Restricciones de compatibilidad del plan de operaciones.* Definen preferencias relativas a los intervalos de tiempo que rodean los tiempos de comienzo y terminación especificados por el plan de operaciones en curso. Estas restricciones reflejan las consideraciones globales subyacentes en las decisiones del scheduler y admiten cierto grado de flexibilidad en su interpretación.

3. *Restricciones heurísticas.* Definen preferencias desarrolladas a partir de reglas heurísticas que sean implementables, al menos parcialmente, como funciones de prioridad (véase por ejemplo [PANW 77]). Así, la regla SPT[7] da prioridad a las operaciones con tiempo de proceso menor. Esta regla es conocida por sus buenos resultados cuando el objetivo es disminuir el tiempo de producción total[8] e incluso resulta óptima para el caso de una sóla máquina. Esta regla puede considerarse como una restricción de preferencia que da prioridad a las operaciones con menor tiempo de proceso.

6.5 Perspectivas Oportunistas

Una idea tradicional para la resolución de problemas complejos consiste en desglosar cada problema en un conjunto de problemas más pequeños cuya resolubilidad individual resulte más sencilla. Este mecanismo de descomposición puede tener lugar en varias etapas, en cada una de las cuales se aplica recursivamente la descomposición en subproblemas a partir de los problemas creados en los pasos previos. Eventualmente, se crean problemas suficientemente sencillos para los que se dispone de una solución o procedimiento heurístico.

[7] "Shortest Processing Time" según su acepción inglesa.
[8] "Makespan" según su acepción inglesa.

Como ya se ha adelantado, en términos generales un problema de planificación de operaciones consiste en asignar recursos e intervalos temporales a las operaciones, con objeto de satisfacer ciertos objetivos de producción. Si se examina esta definición, pueden distinguirse tres conjuntos de variables involucradas: *recursos*, *operaciones* y *tiempos*. Esto va a dar lugar a la identificación de al menos tres formas distintas de enfocar la descomposición de los problemas de planificación de operaciones:

1. **Descomposición basada en los recursos.** Según esta aproximación, el problema se enfoca desarrollando los planes de operaciones para cada recurso. En este caso se fija primero el *recurso*, para ir variando las operaciones asignadas y los intervalos de tiempo.

2. **Descomposición basada en las órdenes.** Considera una perspectiva ortogonal en la que se desarrollan los planes individuales de cada orden u operación agregada. La variable a fijar es la *orden*, sobre la que se tratará de asignar recursos y tiempos.

3. **Descomposición basada en eventos.** Esta descomposición sigue una orientación temporal. El plan de operaciones se desarrolla a modo de una simulación basada en eventos. Se considera la secuencia de instantes en los que es necesario tomar alguna decisión de planificación. Cada instante constituye un subproblema que se resuelve siguiendo una orientación basada en despacho. La variable a fijar es el *tiempo*, a partir de la cual se seleccionan operaciones y se asignan recursos.

La elección de cada una de estas estrategias dependerá del problema de decisión a resolver y de su contexto. La calidad de la solución alcanzada será afectada por la estrategia seguida, puesto que la fijación de la variable elegida en la estrategia impone reducciones drásticas del espacio de la solución y lleva a la consideración de información más local (ningún subproblema contiene toda la información del problema original). Esto da lugar, por otra parte, a la consecución de soluciones más rápidas, factor clave para el módulo del distribuidor. La metodología para la construcción de fuentes de conocimiento debe ser flexible para diseñar y experimentar con estas diferentes estrategias.

6.5.1 Tipos de fuentes de conocimiento del distribuidor

En el contexto de desarrollo que se propone, resultaría interesante disponer de una biblioteca de fuentes de conocimiento que sean directamente utilizables. De esta forma, el usuario podría elegir entre las que resulten más prometedoras e integrarlas de forma modular en sus políticas de control. Estas fuentes de conocimiento se pueden organizar atendiendo a su complejidad y a las características del proceso de

solución de problemas que emplean. En la jerarquía de especialización de fuentes de conocimiento, características del distribuidor de operaciones, se podrían distinguir cuatro tipos principales:

Despacho local Integra simples reglas o algoritmos de despacho donde la asignación de prioridades se basa exclusivamente en restricciones y heurísticas concernientes a características locales al punto de decisión. Ejemplos de estas fuentes de conocimiento son las reglas clásicas de despacho tales como SPT[9] (máxima prioridad al trabajo con tiempo de proceso más corto) o FIFO[10] (prioridad máxima al trabajo con más tiempo en cola de espera), y otras que consideran restricciones relativas a planificación de operaciones como la regla ELST[11] que considera los tiempos de comienzo más tardíos asignados por el planificador de operaciones y elije el trabajo con el tiempo de comienzo planeado más temprano. También son posibles combinaciones de varias de las heurísticas y restricciones propuestas anteriormente. [PANW 77].

Despacho global Utiliza información de otros puntos de la planta adicionales al lugar concreto en que surgió el problema de decisión. Ejemplo de despacho global es la regla EWINQ que elije el trabajo con menor cola de espera en su próxima máquina. [BAKE 74].

Despacho con mirada hacia delante [12] En esta estrategia se comienza con cada decisión alternativa que puede tomarse para resolver el conflicto, a partir de este estado inicial se realiza una búsqueda para explorar los conjuntos de decisiones de asignación de recursos y tiempos que siguen de cada elección durante un corto horizonte temporal. En cada paso de la búsqueda, los candidatos se evaluan con respecto a las restricciones y heurísticas (para el cálculo conjunto se utilizan sus funciones de utilidad) lo que proporciona la base para podar la búsqueda. [VEPS 84,KOCH 87].

La figura 6.4 muestra un ejemplo de este tipo de fuente de conocimiento, que realiza una búsqueda en rayo dirigida por restricciones. El objeto *mirada-adelante-busqueda-enrayo1* dispone información de los diversos métodos para la activación y finalización de la tarea así como para la realización de la búsqueda y sus parámetros.

Despacho con revisión reactiva del plan de operaciones Comprueba la existencia puntual de discrepancias con el plan de operaciones provisto por el nivel de decisión superior y en su caso provoca la activación del planificador de operaciones para generar una actualización reactiva del plan.

---

[9] "Shortest Processing Time" según su acepción inglesa.
[10] "First In First Out" según su acepción inglesa.
[11] "Earliest Latest Start Time" según su acepción inglesa.
[12] "Look ahead dispatch" según su acepción inglesa.

{*mirada-adelante-busqueda-enrayo1*
instance : fc-busqueda-mirada-adelante
; *Metodos para la activacion y finalizacion de la tarea*
ejecuta-tarea : distribuidor-busqueda.ejecuta-tarea
proced-decisiones-finales : distribuidor-busqueda.proced-decisiones-finales
precondicion :
; *Metodos para realizar la busqueda*
metodo : busqueda-enrayo.metodo
ejecuta-accion : accion-busqueda.ejecuta-accion
inicializa-busqueda : mirada-adelante.inicializa-busqueda
post-busqueda : mirada-adelante.post-busqueda
estado-objetivop : mirada-adelante.estado-objetivop
; *Parametros de la busqueda*
anchura-rayo : 9
subacciones : sa-lah-sprout-op1level
particiones-politica : pp-lah-1level }

Figura 6.4: Ejemplo de fuente de conocimiento de despacho con mirada hacia delante.

6.6 Políticas de Control

El sistema presentado ha sido diseñado con la idea de ser una herramienta potente para la construcción de sistemas inteligentes de control, pero esta "inteligencia" viene proporcionada por la habilidad para integrar esta experiencia sobre control. Es importante subrayar aquí que ningún método ha demostrado ser óptimo para cualquier entorno de fabricación flexible, por lo que, en general, la política de control resulta ser, una característica de cada sistema y de cada punto de decisión específico. Con este objetivo, el entorno de desarrollo proporciona una biblioteca de políticas de control, que pueden ser seleccionadas y asociadas a los puntos de decisión. Además facilita la generación de nuevas políticas de control mediante la fácil integración de métodos a través de una flexible agregación modular.

Debido a la complejidad de los problemas de planificación de operaciones, siempre resulta imprescindible un paso final de experimentación para decidir el tipo de guía de planificación útil en cada punto de decisión (el capítulo siguiente aborda este aspecto en mayor profundidad).

La figura 6.5 muestra los atributos básicos de una política de control: método para ejecutar la política (*ejecuta-politica*), relación con el conflicto al que pertenece (*asociada-a-conflicto*) y fuentes de conocimiento a considerar (*acciones*). Para

{*politica-de-control*
ejecuta-politica :
acciones :
asociada-a-conflicto : }

Figura 6.5: Objeto para la *política de control.*

ciertos puntos de decisón, fuentes de conocimiento que integren restricciones de preferencia individuales, simples reglas de despacho o heurísticas de planificación de operaciones, puede ser suficiente para obtener buenas prestaciones del sistema. Para puntos de decisión de mayor complejidad, puede ser necesaria la utilización de fuentes de conocimiento que realizen exámenes más exhaustivos en el espacio de soluciones, o la contrastación de resultados o combinación de varias fuentes simples.

Siguiendo una aproximación tradicional, los problemas de decisión que surgen en los conflictos se pueden refinar desglosandolos en problemas más simples de los que se dispone de conocimiento para su solución. La metodología de diseño impuesta en el modelado se complementa con un método de tratamiento de conflictos de tipo ascendente. Así, en la construcción de los objetos independientes (recursos y planes de trabajo) los conflictos corresponden con decisones operacionales aisladas para los que resulta más sencillo encontrar una política de control adecuada (al menos de forma relativamente).

Desde un punto de vista *estructural*, los orígenes de los conflictos pueden reconocerse a partir de ciertas relaciones. Para los orígenes de los conflictos más simples, se puede pensar en una caracterización atendiendo a la estructura creada por las relaciones de red, en la que distinguimos dos tipos básicos:

1. *eleccion-de-marca*: es el típico conflicto originado cuando un objeto de acción dispone de más de una ligadura consistente.

2. *camino-alternativo*: tiene su origen cuando un objeto de marcado forma parte de una ligadura consistende de más de un objeto de acción.

El origen del conflicto queda especificado por el tipo de origen del conflicto y el dominio y rango de la relación que lo ha originado. El proceso incremental de diseño mediante el establecimiento de relaciones de sincronización provoca otro proceso incremental de agregación de conflictos lo que da lugar, en el nivel de representación del dominio, a nuevas caracterizaciones de orígenes de conflictos como por ejemplo:

- *origen-sincronizacion-parcial*: al que dan lugar las relaciones de sincronización parcial.

- *origen-sincronizacion-condicional*: identifica los conflictos originados al establecer relaciones de sincronización condicional.

Tomando ahora un punto de vista operacional, interesa reconocer los problemas de decisión que surgen en los conflictos, lo que facilitará la construcción de sus políticas de control asociadas. A este respecto, [GERS 86] propone la siguiente agrupación de tipos de decisones operacionales:

- Secuenciamiento de piezas de entrada en el sistema.
- Secuenciamiento de las suboperaciones de proceso en una pieza.
- Secuenciamiento de operaciones.
- Elección de máquinas.
- Elección dispositivos de transporte:
  - movimiento del dispositivos de transporte y
  - requerimiento del dispositivo de transporte.
- Elección de operaciones y frecuencias para pasar el test de calidad.

Existen adicionalmente otros criterios para seleccionar las fuentes de conocimiento y sus parámetros:

1. *objetivos de producción* como cumplir fechas debidas (minimizar tardanza o minimizar número de órdenes tardías), minimizar trabajo en proceso, minimizar tiempo conjunto de puesta en marcha, etc. y
2. *características del entorno de fabricación* como celdas con grupos de máquinas idénticas, situaciónes de saturación de recursos, roturas de máquinas, etc.

Afortunadamente las políticas de control utilizadas para resolver los problemas parciales proporcionan una información muy importante para abordar la solución del conflicto complejo, lo que aboga por una construcción de las políticas de control de tipo incremental siguiendo la jerarquía de agregación de conflictos.

6.7 Ejemplo de Funcionamiento del Distribuidor

Para ilustrar las ideas relativas a la integración del proceso de toma de decisiones en el modelo del sistema, retornaremos el ejemplo presentado en el punto 4.5. En capítulos anteriores se modeló la celda de fabricación desde el punto de vista

físico de recursos (*celda-robotizada1*) y planes de trabajo (*P1* y *P2*), así como la coordinación entre ambas perspectivas (figura 4.22). En la aproximación propuesta en esta memoria, el procedimiento para la toma de decisiones sigue una estrategia que contempla puntos en los que la red KRON presenta indeterminaciones, a los que denominaremos *puntos de decisión.*

En los apartados siguientes se presentar la aplicación de la metodología adoptada en el ejemplo. El proceso se inicia con la identificación de los puntos de decisión y la asignación de políticas de control a dichos puntos. A continuación se presenta una situación concreta de la celda en un momento determinado para el que se dispone de un plan realizado por el planificador de operaciones. Finalmente, se mostrará cómo las políticas de control pueden hacer interpretaciones de este plan con objeto de proporcionar soluciones en los puntos de decisión, tanto para las situaciones previstas, como para aquellas en las que surjan discrepancias con el estado previsto.

6.7.1 Detección de puntos de decisión: descomposición jerárquica de conflictos de red

Examinando la red de la figura 4.22, se deduce la existencia de varios conflictos potenciales de origen estructural. La jerarquía de agregación de conflictos surge de forma relativamente natural, a partir de conflictos simples, surgidos de redes KRON parciales, que se van acoplando formando otros más complejos, originados por el establecimiento de los diversos tipos de *relaciones de sincronización.* Esta descomposición jerárquica proporciona asímismo una base para facilitar el diseño incremental de la estrategia de toma de decisiones. Los conflictos simples presentan una forma más sencilla de resolución al estar las decisiones generalmente más focalizadas, tanto por representar problemas más concretos como por imponer objetivos a satisfacer menos ambiciosos. Esta norma resulta útil para abordar multitud de casos, sin embargo hay ocasiones en las que esta descomposición resulta menos automática y exigen una actuación más "heurística" por parte del diseñador.

Para abordar la estrategia global de resolución de conflictos, considerese el *conflicto simple* identificado por el objeto *confl-plan-P1* de la figura 6.6. Este objeto indica una inderminación de la red KRON representante del plan de proceso $P1$, que surge entre los objetos de acción *comienzo11* y *comienzo13* y precisa de una decisión para determinar cual de las dos trayectorias del proceso (que comienzan con el disparo de *comienzo11* y *comienzo13*) habrá de seguir la pieza. Puede considerarse que el resto de los objetos de acción tienen un conflicto simple, relativamente trivial[13], e identificaremos su origen como del tipo *eleccion-de-marca.*

El establecimiento de ciertas relaciones entre objetos de acción puede provocar

[13]Aunque el conflicto es teóricamente posible en todo objeto de acción, los conflictos efectivos resultan menos frecuentes por lo que la política de control suele ser nula o consistir en una regla de despacho muy simple.

{*confl-plan-P1*
instance : conflicto-plan-proceso
objetos-de-accion : comienzo11 comienzo13
politica-de-control : pc-plan-P1
origen :
{**instance** : camino-alternativo
domain : (P1 op0-P1)
range : comienzo11 comienzo13 }
conflicto-de : conflicto-celda1 }

Figura 6.6: Objetos conflicto simple *confl-plan-P1.*

; *Conflicto originado por el establecimiento de relaciones de sincronizacion.*
{*confl-carga-celda1*
instance : conflicto-carga
objetos-de-accion : carga-M1 carga-M2 carga-M3
politica-de-control : pc-carga-celda1
origen :
{**instance** : origen-sincronizacion-parcial
domain : salida-de-maquina-almacen1
range : carga-M1 carga-M2 carga-M3 }
conflicto-de : confl-carga-descarga-celda1 }
; *Conflicto originado por el establecimiento de relaciones de sincronizacion parcial.*
{*confl-M2-proceso*
instance : sincron-proceso-recurso
objetos-de-accion : carga-M2 comienzo4 sig-op1 sig-op3
politica-de-control : pc-M2-procesos
origen :
{**instance** : origen-sincronizacion-parcial
domain : carga-M2
range : comienzo4 sig-op1 sig-op3 }
conflicto-de : conflicto-celda1 }

Figura 6.7: Objetos conflicto *confl-carga-celda1* y *confl-M2-proceso.*

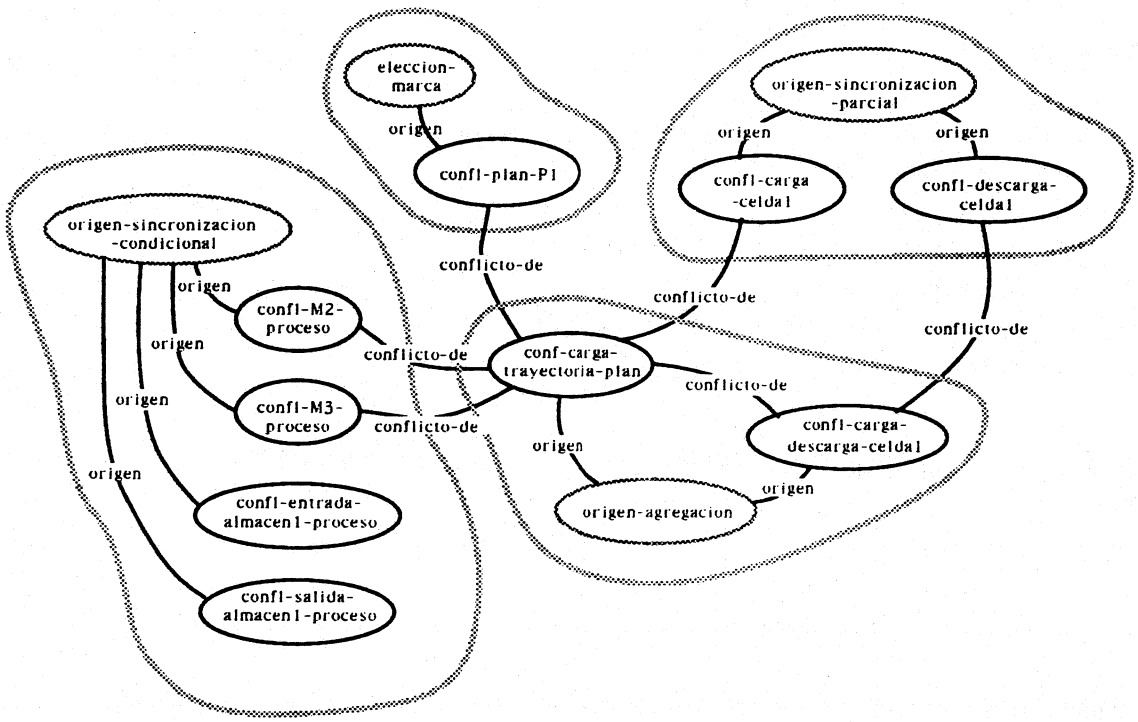

Figura 6.8: Gráfico de relaciones entre conflictos.

la creación de nuevos *conflictos compuestos*, así las relaciones de sincronización parcial y condicional son el origen de otros tantos conflictos. La figura 6.8 muestra gráficamente algunas relaciones entre los conflictos del ejemplo. Surgen conflictos de este tipo al establecer varias relaciones de sincronización parcial o condicional desde un mismo objeto de marcado. En una primera aproximación, se pueden reconocer conflictos básicos directamente a partir de los objetos de acción conectados:

- El objeto conflicto *confl-carga-celda1* (figura 6.7), está originado por las relaciones de sincronización parcial entre *salida-de-maquina-almacen1* y los objetos de acción *carga-M1, carga-M2* y *carga-M3*. *Confl-carga-celda1* representa el problema de decisión para elegir la máquina a alimentar con una pieza de las disponibles en el almacén. Para la descarga se puede identificar otro conflicto análogo *confl-descarga-celda1*.

- Las relaciones de sincronización condicional, que conectan planes de operaciones y recursos, originan también conflictos, en general de solución más simple. Por ejemplo la sincronización entre *carga-M2* con *comienzo4*, *sig-op1* y *sig-op4* se identifica mediante el conflicto *confl-M2-proceso*. Las sincronizaciones condicionadas de *carga-M3*, *entrada-almacen1* y *salida-almacen1* dan origen también a sus correspondientes conflictos.

{*confl-carga-trayectoria-plan*
instance : conflicto-carga-plan
subconflictos : confl-carga-celda1 confl-plan-P1
politica-de-control : pc-carga-eleccion-plan-celda1
conflicto-de : confl-carga-descarga-celda1 }

{*confl-carga-descarga-celda1*
instance : conflicto-carga-descarga
subconflictos : confl-carga-trayectoria-plan confl-descarga-celda1
politica-de-control : pc-carga-descarga-celda1
conflicto-de : confl-celda1 }

Figura 6.9: Objetos conflictos a nivel agregado de *celda-robotizada1*.

Algunos de los conflictos reseñados anteriormente no son independientes, sino que pueden verse como solapados en conflictos agregados más complejos. La relación *conflicto-de* permite conectar conflictos a diferentes niveles de agregación, su inversa es *subconflictos*.

- Los conflictos *confl-plan-P1* y *confl-carga-celda1* están solapados (objeto *confl-carga-trayectoria-plan* de la figura 6.9). El problema de decisión supone la elección combinada de pieza y trayectoria de proceso o, en otros términos, debe decidirse el par máquina-pieza que define la especificación de carga que deberá ser llevada a cabo por el robot.
- Además de los conflictos surgidos en la actuación del robot para efectuar la carga (*confl-carga-trayectoria-plan*) y la descarga (*confl-descarga-celda1*) de una pieza, existe un conflicto adicional que supone la decisión de utilización del robot en alguno de los dos sentidos (objeto *confl-carga-descarga-celda1* mostrado en la figura 6.9).

6.7.2 Asociación de políticas de control

Las políticas de control tienen como objetivo dar solución a los conflictos. Para el conflicto *confl-carga-descarga-celda1*, que corresponde al mayor nivel de agregación, se ha elegido como ejemplo la política *pc-carga-descarga-celda1* (en la figura 6.10 se muestran algunos objetos utilizados en la definición de esta política). La idea de esta política consiste:

1. En comprobar la compatibilidad con el plan de operaciones previsto, para lo que utiliza la fuente de conocimiento *fc-compatibilidad-plan*, que es la de

```
                                    ; Política de control
{pc-carga-descarga-celda1
    instance :  politica-control
    ejecuta-politica :  ejecuta-multiple-politica-con-analisis
    acciones :  fc-compatibilidad-plan  fc-busqueda-completa-celda1
        fc-refinamiento-celda1
    acciones-analisis :  fc-analisis-celda1
    reaccion-del-planificador :  t
    politica-control-de :  confl-carga-descarga-celda1 }
                                    ; Fuente de conocimiento
{fc-compatibilidad-plan
    instance :  fc-despacho-local
    ejecuta-tarea :  despacho.ejecuta-tarea
    proced-decisiones-finales :  despacho.proced-decisiones-finales
    subacciones :
    particion-restricciones-fc :  part-disrupcion-plan  part-restricciones-locales }
                                    ; Particiones de restricciones
{part-disrupcion-plan
    instance :  politica-particion
    importancia-particion :  1.0
    espec-particion :  restr-disrupcion-inc-restric  restr-disrupcion-opor-restric }
{part-restricciones-locales
    instance :  politica-particion
    importancia-particion :  0.1
    espec-particion :  restr-preferencia-cola  restr-importancia-ordenes
        rest-preferencia-robot }
                                    ; Restricciones
{restr-disrupcion-inc-restric
    instance :  restriccion
    importancia :  1.0
    precondicion-satisfecha :  t
    tipo-efecto :  local
    tipo-existencia :  estatica
    encuentra-objeto :  general
    encuentra-atributo :  disrupcion-sigue-inc
    evaluacion :  disrupcion-sigue-eval }
```

Figura 6.10: Algunos objetos relacionados con la política de control *pc-carga-descarga-celda1*: politica de control, fuentes de conocimiento, particiones y restricciones.

mayor prioridad. Para ello aplica dos tipos de partición de restricciones, unas de compatibilidad con el plan y otras, con poco peso específico, consistentes en preferencias locales.

2. Si la primera fase falla se realiza una segunda fase de análisis del conflicto, *fc-analisis-celda1*, que permite comprobar la viabilidad de una resolución por subconflictos y proporciona la información para decidir entre la aplicación de *fc-busqueda-completa-celda1* y *fc-refinamiento-celda1*. La primera realiza una búsqueda considerando el conflicto agregado, mientras que la segunda efectúa una llamada a la política de control del subconflicto.

3. Si finalmente no ha sido encontrada una solución suficientemente satisfactoria se invoca la reacción del siguiente nivel de la jerarquía de decisión (atributo *reaccion-del-planificador* con valor verdad).

Desde el punto de vista de planificación de operaciones, el robot se considera un recurso no-restrictivo puesto que sus tiempos de paletización son sensiblemente inferiores a la media de los tiempos de proceso de las operaciones, por lo que las políticas de control no tienen que abordar específicamente su problemática.

6.7.3 Resolución de los conflictos de red

Para ilustrar el funcionamiento del distribuidor, se considerará una situación particular. Se asume que para fabricar tres piezas del tipo $P1$ (que denominaremos como $P11$, $P12$ y $P13$) y dos del tipo $P2$ ($P21$ y $P22$), el módulo de planificación de operaciones ha proporcionado el plan de operaciones que se encuentra esquematizado en el gráfico Gantt de la figura 6.11. Se asume también que en el momento considerado como tiempo inicial de observación, el almacén está vacío y que en la máquina $M3$ se está llevando a cabo la operación $OP4$ sobre la pieza $P21$.

A continuación se muestran cuatro problemas de decisión típicos que pueden surgir en el modelo y el proceso de razonamiento que es utilizado por el distribuidor para resolverlos. El instante en el que aparece el conflicto correspondiente se asume que coincide con el origen de tiempos de gráfico Gantt de la figura 6.11.

Problema de decisión 1

Situación Llegada de la pieza $P11$ al almacén. El objeto de acción *carga-M1* está habilitado debido a la disponibilidad de la máquina $M1$ y a la presencia de la pieza en *almacen1* y $P1$.

Conflicto Desde el punto de vista de la red no existe un conflicto real, sin embargo, desde el punto de vista de la toma de decisiones, el distribuidor dispone de dos alternativas:

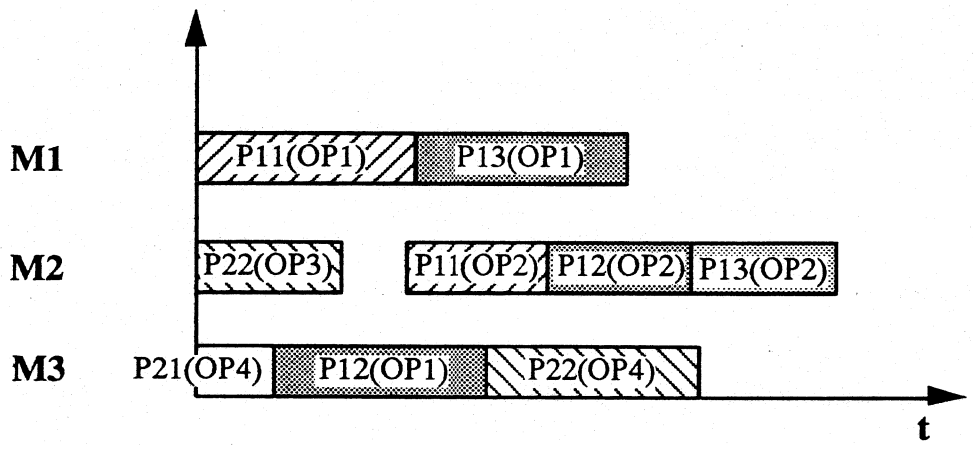

Figura 6.11: Gráfico Gantt de un posible plan de operaciones de operaciones.

- autorizar el disparo del objeto de acción, asignando el recurso $M1$ (la máquina $M3$ está ocupada), o
- no autorizar dicho disparo.

Proceso de resolución El primer paso de la política de control, *pc-carga-descarga-celda1*, consiste en determinar el grado de compatibilidad de la situación actual con el plan de operaciones inicialmente previsto (gráfico Gantt de la figura 6.11). Este efecto se consigue debido a que se aplica inicialmente la fuente de conocimiento *fc-compatibilidad-plan*. En este caso, el grado de compatibilidad es muy alto, y el disparo resulta congruente con el plan de operaciones disponible.

Decisión La decisión del distribuidor es autorizar el disparo de la transición *carga-M1* con respecto a la sustitución: $< vrecurso >= M1$, $< vpieza >= P11$.

Problema de decisión 2

Situación Llegada simultanea al almacén de tres productos del tipo $P1$ (piezas $P11$, $P12$ y $P13$). El objeto de accíon *carga-M1* está habilitado.

Conflicto Existe un conflicto entre las piezas $P11$, $P12$ y $P13$ que estan compitiendo por el recurso $M1$. Desde el punto de vista operacional, el procedimiento de decisión debe dilucidar si se ha de introducir alguna pieza y, en tal caso, cuál de las tres es la más adecuada.

Proceso de resolución En este caso, las restricciones de compatibilidad del plan de operaciones sólo se encuentran parcialmente satisfechas (debido a la imprevista disponibilidad de las piezas $P12$ y $P13$ en el instante de tiempo

considerado). El análisis mediante *fc-analisis-celda1*, detecta la necesidad de realizar un razonamiento más exhaustivo limitado al conflicto *confl-carga-trayectoria-plan*, su política de control aplica una acción de búsqueda con objeto de encontrar la mejor decisión entre autorizar el disparo del objeto de acción con respecto a la sustitución: $< vrecurso >= M1$, $< vpieza >= P11$ (que es, de hecho, compatible con el plan) y el intento de mejorar el plan de operaciones calculado tomando una decisión alternativa.

Decisión La decisión final depende del área del plan de operaciones que rodea el punto de decisión, en principio tendrá más prioridad la sustitución: $< vrecurso >= M1$, $< vpieza >= P11$ prevista inicialmente.

Problema de decisión 3

Situación Llegada al almacén de una pieza de cada tipo: $P11$ y $P22$.

Conflicto El conflicto surge en el objeto de acción *carga-M3*. Cualquiera de las dos acciones: carga de la pieza $P11$ o carga de $P22$, puede ser ejecutada, pero la capacidad disponible del robot solo permite la ejecución de una de ellas en cada momento.

Proceso de resolución Las restricciones de compatibilidad del plan de operaciones están plenamente satisfechas en este caso (el estado de las piezas $P11$ y $P22$ es completamente consistente con el estado anticipado por el planificador de operaciones). Sin embargo, sólo puede ser tomada una única decisión (el robot es el recurso generador del conflicto, puesto que no existen máquinas disponibles donde sea posible la ejecución simultanea de más de una pieza). Para determinar la sustitución a autorizar en el disparo se aplica nuevamente *fc-compatibilidad-plan*, pero en este caso, la evaluación de las otras restricciones locales relevantes a las órdenes (preferencias de cola y prioridades) y a los recursos (restricciones de preferencia asociadas con el robot) constituyen un factor clave en la decisión.

Problema de decisión 4

Situación Llegada al almacén de las piezas $P12$ y $P13$.

Conflicto El conflicto surge en la acción de carga de las máquinas, y están involucradas las operaciones $P12 - OP1$ y $P13 - OP1$ compitiendo por el recurso $M1$.

Proceso de resolución Las restricciones de compatibilidad del plan de operaciones están muy pobremente satisfechas en este caso precisándose, una vez más, de la ejecución de una búsqueda. Las entidades directamente involucradas en el conflicto inicial son las operaciones

$P12-OP1$ y $P13-OP1$ y el recurso $M1$, sin embargo la búsqueda es extendida a un área mayor involucrando también a otras entidades incluidas en un cierto entorno temporal que rodea el punto de decisión: operaciones $P21 - OP4$, $P11 - OP1$ y $P22 - OP4$, y recurso $M3$. Se calcula una estimación de los tiempos de llegada de estas últimas operaciones así como el instante de terminación estimado para la operación $P21 - OP4$. Desde este punto de partida se exploran las consecuencias de diversas decisiones:

1. "no realizar ninguna acción hasta la llegada de $P11$",
2. "disparo del objeto de acción respecto a la sustitución: $< vrecurso >= M1$, $< vpieza >= P12$" o
3. "disparo respecto a la sustitución: $< vrecurso >= M1$, $< vpieza >= P13$".

Una vez se ha seleccionado la decisión más satisfactoria (según la política de control elegida), se evalúa el grado de compatibilidad del futuro estado que produce la decisión para determinar si debe ser informado el módulo de planificación de operaciones que supone el siguiente nivel superior en la jerarquía de decisión.

6.8 Monitorización y Gestión de Excepciones

La capacidad de monitorizar el trabajo llevado a cabo en la planta y de detectar eventos inesperados, es fundamental en cualquier sistema de control de fabricación. El monitor es el subsistema que cierra el bucle entre la planificación de la producción y la realización real de la planta [HIGG 90].

En el área de redes de Petri se han realizado algunos trabajos para la monitorización de sistemas de fabricación. Destacan los realizados por Valette y col. [VALE 87] que utilizan una aproximación de modelado relativamente cercana a la propuesta en el presente trabajo (redes de Petri de decisión). En sus trabajos más recientes [VALE 89] han incrementado su potencia descriptiva, incluyendo marcados imprecisos en lo que denominan Fuzzy-time Petri nets que aplican también a problemas de monitorización.

El propósito del **módulo de monitorización**, integrado en la arquitectura de control, es captar los cambios que suceden en la planta y actualizar el modelo (información relacionada con operaciones, recursos o materiales y cualquier otra información de realimentación del estado proveniente del sistema de fabricación a controlar) y detectar situaciones donde el comportamiento real se desvía de las predicciones realizadas sobre el sistema. Este módulo está integrado por tres componentes principales: sistema de recogida de información externa, módulo de actualización del estado y módulo de gestión de excepciones, que se examinan

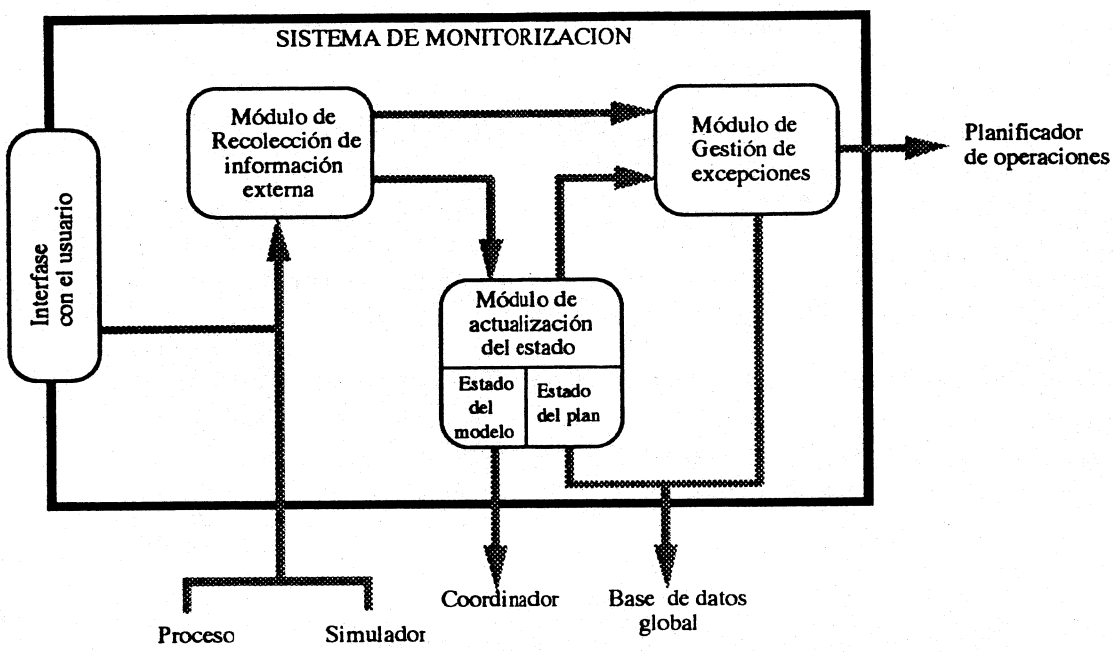

Figura 6.12: Sistema de monitorización y gestión de excepciones.

a continuación con más detalle (la figura 6.12 muestra un esquema funcional del sistema de monitorización y gestión de excepciones).

6.8.1 Módulo de recolección de información externa

Trabaja como interfase entre la información externa proveniente del proceso, sensores y otros dispositivos automáticos de información, y el sistema de control. Los datos deben ser muestreados continuamente en tiempo real, y el proceso debe ser robusto frente a lecturas espúreas y malfuncionamiento de los sensores. Recoge la información sobre los instantes de finalización de las operaciones así como los cambios de estado de los recursos. Puede clasificar y filtrar los eventos de funcionamiento normal de las condiciones de excepción de los procesos de fabricación de bajo nivel tales como disrupciones operacionales o roturas de máquinas. Los eventos anormales son enviados al módulo de gestión de excepciones.

6.8.2 Módulo de actualización del estado

Su cometido es actualizar el estado del modelo para reflejar las nuevas condiciones de la planta. Informa al interpretador de la red de la finalización física de las actividades, lo que habilita el disparo de los objetos de acción condicionados por

estos eventos externos. Por otra parte, actualiza la información del estado del plan de operaciones elaborado. Esto permite la detección de situaciones donde el comportamiento real de la planta se desvía de las predicciones del planificador de operaciones.

Desde una perspectiva basada en restricciones, esto equivale a una comparación entre las restricciones definidas en el modelo (p.e. las reservas de recursos contenidas en el plan de producción) y las restricciones resultantes de la operación de la planta (esta perspectiva ofrece una aproximación directa a la identificación de desviaciones). Si tiene lugar tal desviación, el módulo de actualización del estado pasa información al módulo de gestión de excepciones para controlar el problema.

Otro de los cometidos de este módulo es recopilar datos para elaborar estadísticas de los componentes físicos (p.e. utilización de herramientas, máquinas, tiempos de caída de máquinas, tiempos de puesta en marcha, etc.), operaciones (p.e. tiempos de procesamiento) y órdenes (p.e. satisfacción de fechas debidas).

6.8.3 Módulo de gestión de excepciones

El módulo de gestión de excepciones tiene como responsabilidad la gestión de las condiciones excepcionales que surjan en los componentes físicos y en el plan de operaciones previsto. Su trabajo se desarrolla en dos direcciones:

1. Informa al sistema de coordinación en caso de una excepción en componentes físicos. En respuesta, el coordinador puede alterar el estado de la red y abortar las tareas asociadas con el disparo de objetos de acción

2. Envía al planificador de operaciones el evento correspondiente a la excepción. En su caso el planificador puede desencadenar una reacción para replanificar la zona del plan de operaciones que se haya visto afectada.

Una situación de excepción típica en el plan de operaciones surge cuando una operación finaliza con posterioridad a su tiempo de finalización previsto, y su intervalo de ejecución se superpone al intervalo de tiempo planeado de su siguiente operación, originando una violación de las restricciones temporales. Para gestionar esta contingencia, el módulo de excepciones crea un evento de conflicto de tiempo y lo envía al planificador de operaciones. La situación inversa, en la que la operación finaliza con anterioridad al instante previsto, puede ser también gestionada e informada al planificador como una oportunidad para mejorar el plan de operaciones.

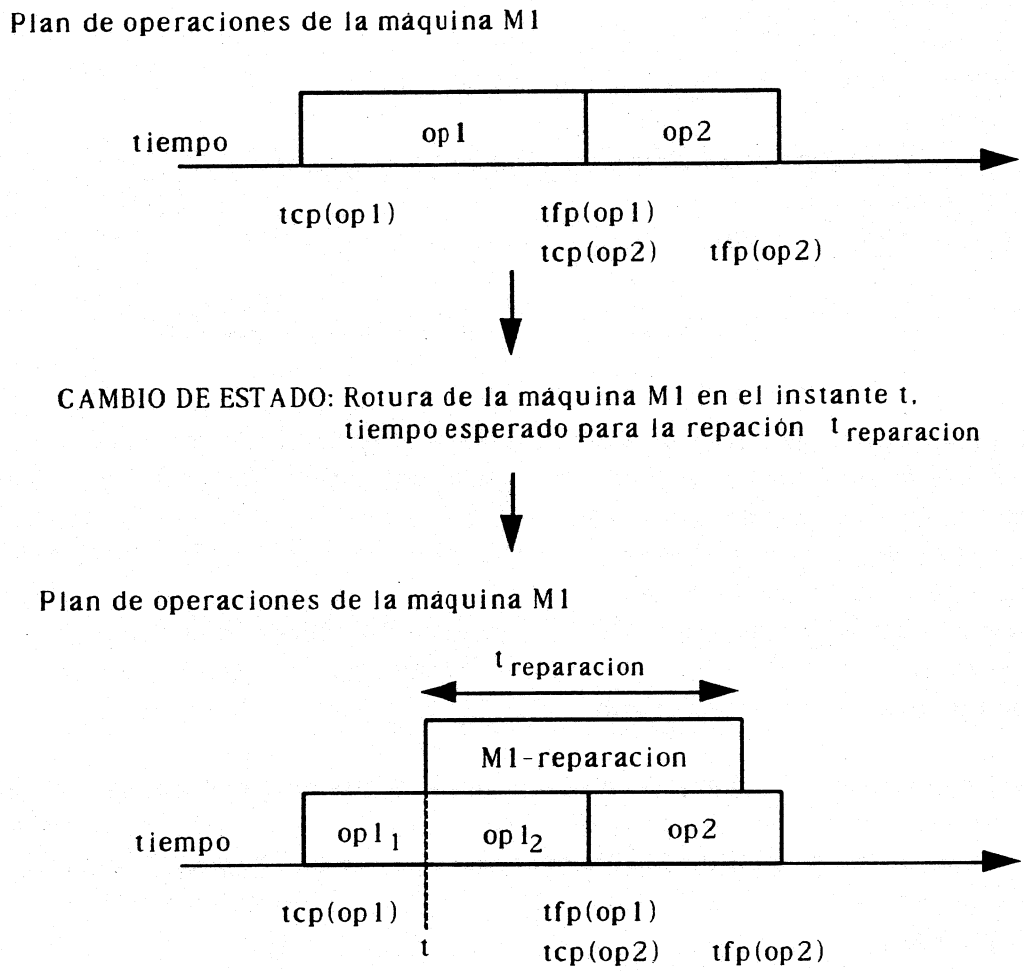

Figura 6.13: Rotura de máquina.

Rotura de una máquina

En el apartado de experimentación se han utilizado únicamente dos tipos de situaciones excepción: rechazos en control de calidad y roturas de máquinas. Para ilustrar la metodología de trabajo se pasa a explicar más en detalle este último tipo.

La gestión de la situación de excepción es más compleja cuando implica una contingencia física importante tal como la rotura de una máquina. Una vez ha sido detectado este problema, el curso de la acción a llevar a cabo consiste en conducir al sistema a un estado seguro. Para realizar esto, el módulo puede alterar el estado de la red y abortar las tareas asociadas con el disparo de los objetos de acción, que son responsabilidad del coordinador. Por otra parte, este módulo activa además una extensión de la red que tiene por objetivo gestionar la anomalía.

{*rotura-maquina-6045*
instance : rotura-maquina-introducida
introducida-por : modulo-monitorizacion
recurso : $M1$
operaciones-interrumpidas : $op1_r$
operaciones-en-conflicto : ($op1_r$ *M1-reparacion*) ($op2$ *M1-reparacion*)
capacidad-perdida : 1 }

Figura 6.14: Evento que informa de la rotura de una máquina.

Una vez que ha sido realizada esta acción de información al coordinador se pasa a otra fase de información al planificador de operaciones. Para explicar la gestión de esta excepción considerense los gráficos Gantt del ejemplo de la figura 6.13. El gráfico superior muestra las ventanas de tiempo planeadas para dos operaciones $op1$ y $op2$ a realizar en la máquina $M1$. Supongase ahora un cambio de estado que supone la rotura de la máquina $M1$ en el instante t, ocurrido mientras se procesaba $op1$, para el que estaba prevista una duración de t_{rep}. La gestión de este proceso puede descomponerse en varios pasos:

- La reparación de la máquina supondrá su indisponibilidad durante el tiempo que esta dure. Para poder manipular esta circunstancia de manera homogénea con el resto de procesos de planificación, se crea una operación especial, *M1-reparacion*, a la que se asigna unas restricciones temporales inviolables entre t y $t + t_{rep}$.
- Como la operación $op1$ no ha llegado a ser completada, se crean dos nuevas operaciones en secuencia, que sustituyen a $op1$:
  1. $op1_c$ correspondiente a la parte de $op1$ que ha sido finalizada (comprendida entre $t_{cp}(op1)$ y t), y
  2. $op1_r$ correspondiente a la porción de $op1$ que resta por realizar (comprendida entre t y $t_{fp}(op1)$)
- La nueva operación crea una violación de capacidad en el recurso (supuesto de capacidad 1):
  - ($op1_r$ *M1-reparacion*)
  - ($op2$ *M1-reparacion*)
- Este hecho introduce una inconsistencia en el plan de operaciones previsto, el gestor de excepciones informa al planificador enviando un evento con alta prioridad (*rotura-maquina-6045* de la figura 6.14) requiriendo una acción correctora por parte del planificador.

6.9 Conclusión

El carácter no determinista de la red favorece la aparición de conflictos estructurales en el sistema de coordinación. Estos conflictos son resueltos aplicado las políticas de control asociadas a los conflictos.

En general, la responsabilidad de las políticas de control es llevar a cabo el plan de operaciones elaborado por el planificador. Si esto no es posible (p.e. existen incompatibilidades), cuenta con fuentes de conocimiento con capacidad para realizar una rápida búsqueda para encontrar una "buena" solución. En esta búsqueda se consideran únicamente restricciones locales y se favorecen las soluciones que disminuyan la distancia entre el plan de operaciones y el estado real de la planta de fabricación. El fracaso en producir una interpretación "satisfactoria" del plan de operaciones en curso resulta en una invocación del planificador de operaciones que trabaja fuera de línea.

Si bien esta es una "buena" estrategia a seguir, el distribuidor de operaciones permanece como un sistema flexible para el diseño de algoritmos de control heurísticos. El usuario debe tener la posibilidad de probar con distintas políticas para poder elaborar la más adecuada. Con este fin, el distribuidor debe disponer de un repertorio de reglas heurísticas y algoritmos de búsqueda que el usuario pueda manipular modularmente.

TORTUGAS Y CRONOPIOS

Ahora pasa que las tortugas son grandes admiradoras de la velocidad, como es natural.

Las esperanzas lo saben, y no se preocupan.

Las famas lo saben y se burlan.

Los cronopios lo saben, y cada vez que encuentran una tortuga, sacan la caja de tizas de colores y sobre la redonda pizarra de la tortuga dibujan una golondrina.

Historias de cronopios y famas
JULIO CORTAZAR

Capítulo 7

Resultados Experimentales y Simulación

El objetivo final del estudio experimental es, en primer lugar, determinar un conjunto de criterios que proporcionen una base para decidir entre estrategias reactivas alternativas, y en segundo término, desarrollar un conjunto de heurísticas, basadas en estos criterios, para coordinar el proceso reactivo. Estos criterios y heurísticas, que fueron surgiendo paralelamente con el desarrollo de los experimentos, ya han sido presentados en capítulos anteriores, el presente capítulo aborda su justificación en base a los resultados experimentales.

Con este objetivo se han realizado dos series de experimentos, todos basados en una porción de fábrica de ensamblado de tarjetas de computador. La primera serie trata de analizar el comportamiento del planificador de operaciones trabajando en modo reactivo. La idea es realizar un análisis comparativo de las estrategias reactivas alternativas, con respecto a actualizaciones específicas del estado, bajo diferentes circunstancias. La segunda serie analiza las decisiones generadas por el distribuidor de operaciones. El método en este caso ha sido incluir indeterminaciones en la información relativa a ciertos parámetros (p.e. tiempos de proceso y condiciones de terminación), a través de un proceso de simulación de eventos discretos se evalán los efectos compuestos de distintas estrategias sobre cierto periodo de tiempo. Adicionalmente se presenta una visión general del simulador.

7.1 Introducción

La evaluación experimental de gran parte de sistemas complejos de IA es una tarea difícil y de controversia [FOX 83]. Cabría mencionar dos causas fundamentales:

- No existen métricas simples para evaluar la eficacia de un sistema de producción. Sólo en los sistemas en los que el dominio disponga de un criterio claro que determine el alcance del objetivo, como que el diagnóstico sea correcto (p.e. MYCIN) o una frase sea reconocida (p.e. Hearsay-II), es posible determinar sencillamente el éxito del sistema. En el sistema de toma de decisiones propuesto se trata de lograr un compromiso entre objetivos múltiples y a veces contrapuestos.

- Existe una importante cantidad de parámetros, que pueden hacer variar sustancialmente la naturaleza del problema.

El problema, a la hora de diseñar los experimentos, residirá tanto en elegir los parámetros que permitan componer situaciones representativas, como en encontrar parámetros que permitan realizar la valoración de las decisiones de control.

El objetivo que se plantea en el presente capítulo es comprobar la adecuación del sistema de toma decisiones para el funcionamiento en modo reactivo. Para este análisis se realizaron dos conjuntos de experimentos que cubren los dos aspectos mencionados.

El primero aborda el aspecto de la planificación reactiva de un modo estático. En él, se parte de un plan de operaciones previamente elaborado, en el que se introduce un conflicto en el plan de operaciones, originado por una contingencia externa. El propósito de los experimentos es valorar las prestaciones del sistema para resolver el problema. Con este objeto se proponen algunos parámetros que permitan su medida.

El segundo tiene en cuenta un problema de tipo más dinámico que se aborda mediante simulación. Se trata de comparar las diferencias en las prestaciones de políticas de control que hacen uso de reglas de despache tradicionales, con las prestaciones de políticas que utilizan información del plan de operaciones elaborado por el planificador del sistema de decisión (interpretación del plan). El problema otra vez es la inexistencia una medida simple de la eficacia del trabajo de una planta de fabricación. En este caso se propondrán algunas conclusiones a partir de varias medidas de prestaciones tradicionales (trabajo en proceso, tardanza, órdenes tardías, etc.).

7.1.1 El Entorno de Fabricación

El entorno de fabricación utilizado en los experimentos corresponde a una planta de fabricación de tarjetas de computador, instalado en la factoría IBM Card Line en Poughkeepsie, NY, (USA). El sistema consiste en una línea de ensamblaje y test de tarjetas, que contiene 10 sectores (recursos agregados). En la línea se realizan aproximadamente unas 18 operaciones a nivel sector por tarjeta. Aunque las tarjetas

recorren dichos sectores esencialmente en línea, las operaciones de control de calidad pueden determinar operaciones adicionales de reparación y reprocesamiento. Esto hace que las tarjetas procedan por la planta en una forma no lineal.

El sistema de decisión trabaja en los experimentos con el mismo tipo de restricciones de coste de almacén, trabajo en proceso y fecha debida para todas las órdenes. La planta recibe órdenes por unidades de producto con una fecha de entrega adjunta. Se asigna asímismo una fecha de entrada a cada orden, que especifica el instante más temprano en el que puede comenzar su procesamiento. El planificador de operaciones sólo tiene en cuenta las órdenes conocidas.

7.2 Estudio Experimental del Planificador de Operaciones en Modo Reactivo

El objetivo de este primer conjunto de experimentos es realizar un análisis comparativo de estrategias reactivas alternativas, con respecto a actualizaciones específicas del estado, bajo diferentes circunstancias. En otras palabras, comprobar la capacidad del modelo para seleccionar, adecuadamente, acciones en planificación reactiva.

Con el propósito mencionado, se ha generado un plan de operaciones adecuado para la realización de los experimentos. Para la generación de este plan se han considerado características tales como *contención de los recursos*, *cumplimiento de fechas debidas*, *cantidad de órdenes tardías*, etc. La carga de producción consiste en 30 órdenes de trabajo, las operaciones para producir cada orden se han instanciado probabilísticamente[1], contabilizando un total de 510 operaciones. A partir de una situación de estabilidad identificada por el seguimiento fiel del plan de operaciones, se introducen dos tipos de eventos externos correspondientes a contingencias de alta prioridad: *roturas de máquinas* y *condiciones de fallo en operaciones de proceso* (que generan operaciones adicionales de reparación). Estos eventos provocan además la aparición de conflictos, por contención de recursos al exceder su capacidad, y la violación de restricciones temporales en el plan de operaciones, con los consiguientes eventos de violación de límites de tiempo.

7.2.1 Diseño de los experimentos

Los resultados que se presentan corresponden a una serie de doce experimentos, en los que, con objeto de crear conflictos, se han introducido condiciones externas en

[1]Para la instanciación probabilística de las operaciones de una orden, se utilizan las probabilidades indicadas en los objetos de acción que corresponden al fin de operación de su plan de proceso (véase la figura 4.21).

FC	*PORD-VP*	*PORD-VP*	*PREC*	*DOP*	*CDEM*
num	21	29	13	24	12

Figura 7.1: Activaciones de fuentes de conocimiento de planificación consideradas en los experimentos.

diversas partes del plan de operaciones. Los problemas introducidos consisten en condiciones de fallo en operaciones de control de calidad y roturas de máquinas, y han sido elegidos con el propósito de crear una amplia gama de conflictos de tiempo y capacidad.

Cada experimento consiste en aplicar, al plan de operaciones inicial con el conflicto, alternativamente las acciones de planificación de operaciones reactivas presentadas en el apartado 5.5.1, que denotaremos en las gráficas de resultados por sus siguientes símbolos clave:

1. *PORD-VC* : Planificador de órdenes con visibilidad completa
2. *PORD-VP* : Planificador de órdenes con visibilidad parcial
3. *PREC* : Planificador de recursos
4. *DOP* : Desplazador de operaciones a derecha
5. *CDEM* : Conmutador de demandas
6. *ALEA* : Acción aleatoria

La opción *ALEA* consiste en la aplicación de una acción elegida aleatoriamente, los pesos asignados a cada una de las acciones se tomaron a partir de las ocurrencias de dichas acciones según fueron seleccionadas por el metaconocimiento de planificación en una serie de experimentos. Se contabilizaron 99 ocurrencias de las diferentes acciones de planificación para las que se obtuvo la distribución mostrada en la figura 7.1.

Estos experimentos presentan el inconveniente de que algunas acciones no tienen la posibilidad de dejar el plan de operaciones libre de conflictos, puesto que la actuación sobre un área del plan puede generar nuevas violaciones de restricciones en otras áreas adyacentes. Al no tener sentido evaluar la calidad de un plan de operaciones en el que permanecen inconsistencias, la evaluación de prestaciones se ha aplicado a **series** de acciones de planificación de operaciones reactivas, en las que se toma la primera de ellas como representativa para la serie. En los casos necesarios se han ensayado varias secuencias y se han tomado los datos de la secuencia con mejores resultados.

La evaluación de los resultados de los experimentos se basó en la comparación respecto a dos tipos de criterios para los que se adoptaron las siguientes medidas empíricas:

Criterios basados en la calidad del plan de operaciones. Dan idea de los cambios según criterios de optimización globales:

- COT: cambio en órdenes tardías,
- CTT: cambio en tiempo de tardanza y
- CTP: cambio en el tiempo de trabajo en proceso.

Criterios basados en la disrupción (alteración) del plan de operaciones. Informan de los cambios realizados en el plan de operaciones inicial y por tanto dan idea de la disrupción ocasionada en el plan y el coste computacional de la acción aplicada:

- NOC: número de órdenes cambiadas,
- NRC: número de reservas cambiadas y
- $TMCR$: tiempo medio cambiado por reserva.

Por cada experimento se representan los resultados de las distintas medidas, obtenidas al aplicar exhaustivamente las 6 acciones de planificación comentadas. Las figuras 7.2 a la 7.7 representan dichos resultados respecto a cada uno de los criterios de evaluación mencionados arriba. Los resultados experimentales mostrados, se encuadran dentro de una serie más larga de ensayos, en la que cada uno está identificado por su posición en la serie. Los experimentos presentados corresponden a los números: $10, 12, 16, 22, 24, 26, 28, 30, 32, 34, 36$ y 38.

Permanece todavía el problema de determinar la mejor estrategia respecto al conjunto completo de medidas, con la dificultad de mezclar valores de diferentes magnitudes, que pueden aportar diferentes pesos en la estimación global. Para poder llevar a cabo esta determinación se ha procedido en dos pasos:

1. *Cálculo de una medida de prestaciones normalizada para cada criterio.* Consistente en la conversión de los valores absolutos recogidos en los experimentos a un valor normalizado. La medida normalizada para cada criterio, $\{COT, CTT, CTP, NOC, NRC, TMCR\}$, ha sido calibrada con relación a valores medios de una serie mayor de 26 experimentos, siguiendo la siguiente expresión:

$$XN_{X_i^a} = \frac{X_i^a - Med_X}{Var_X}$$

donde:

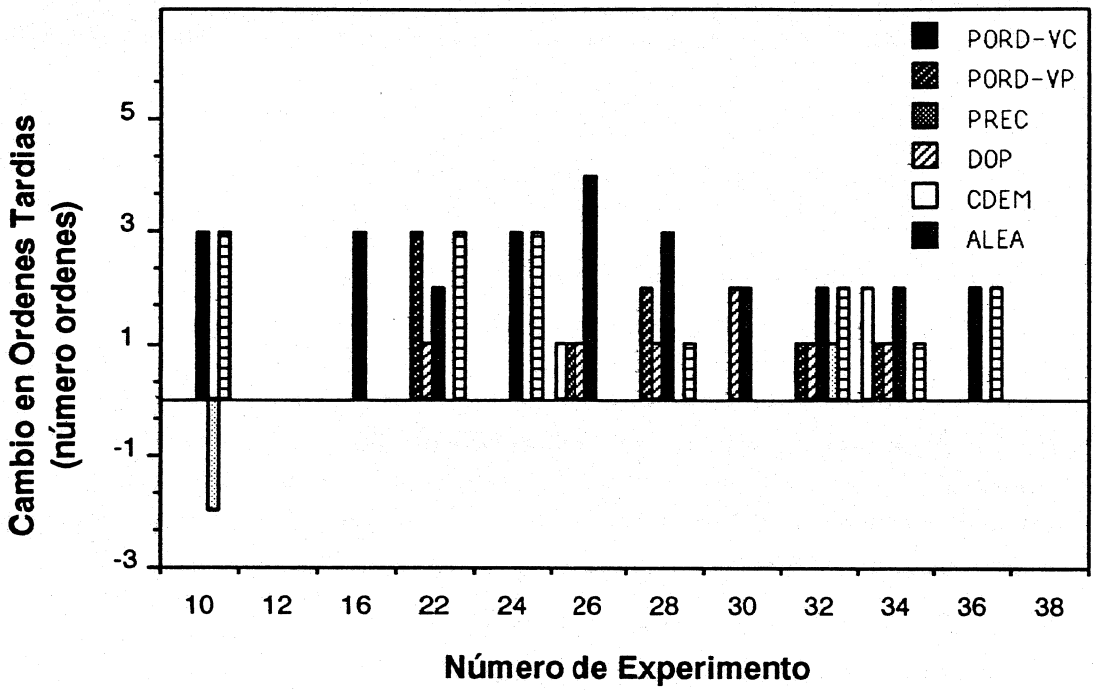

Figura 7.2: Cambio en órdenes tardías por experimento y fuente de conocimiento.

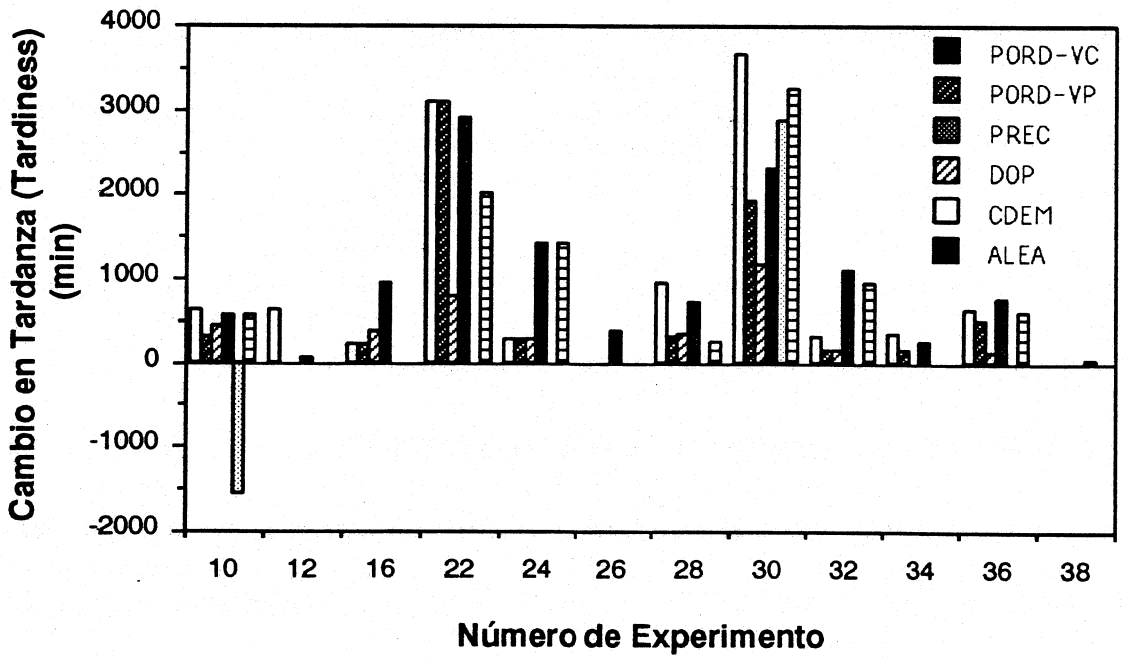

Figura 7.3: Cambio en tiempo de tardanza por experimento y fuente de conocimiento.

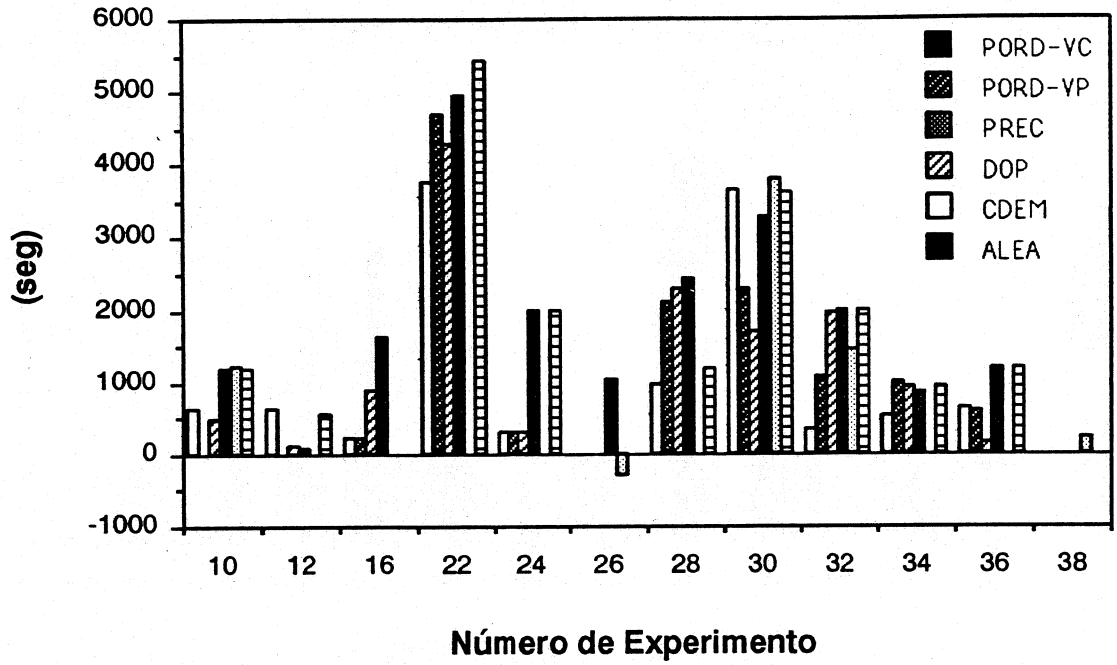

Figura 7.4: Cambio en el tiempo de proceso por experimento y fuente de conocimiento.

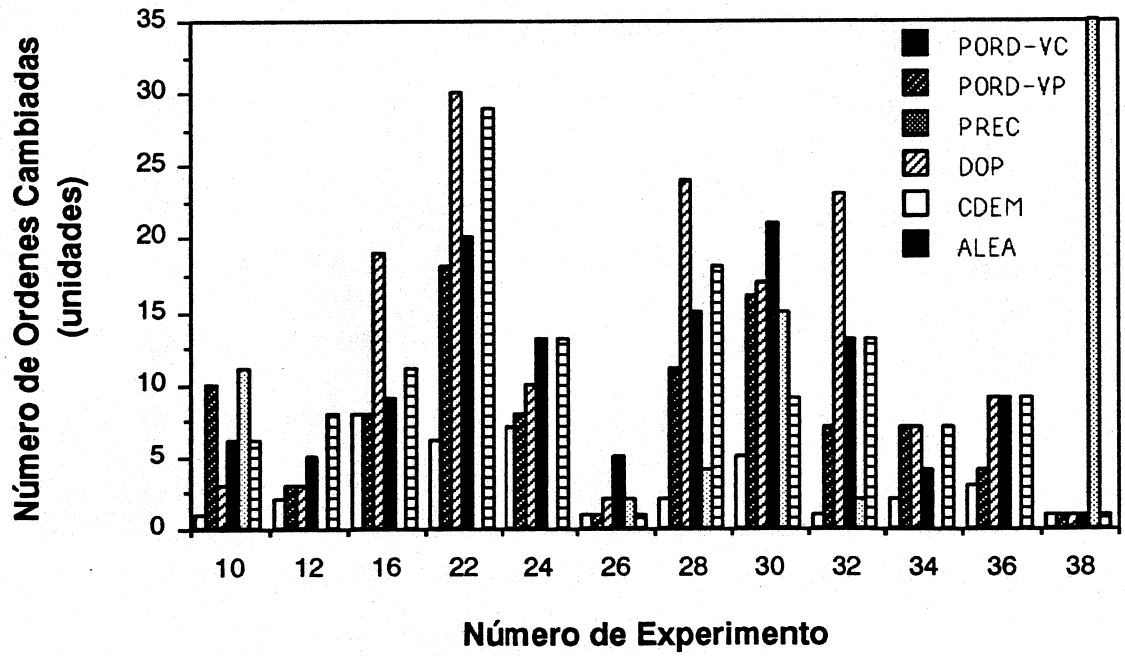

Figura 7.5: Número de órdenes cambiadas por experimento y fuente de conocimiento.

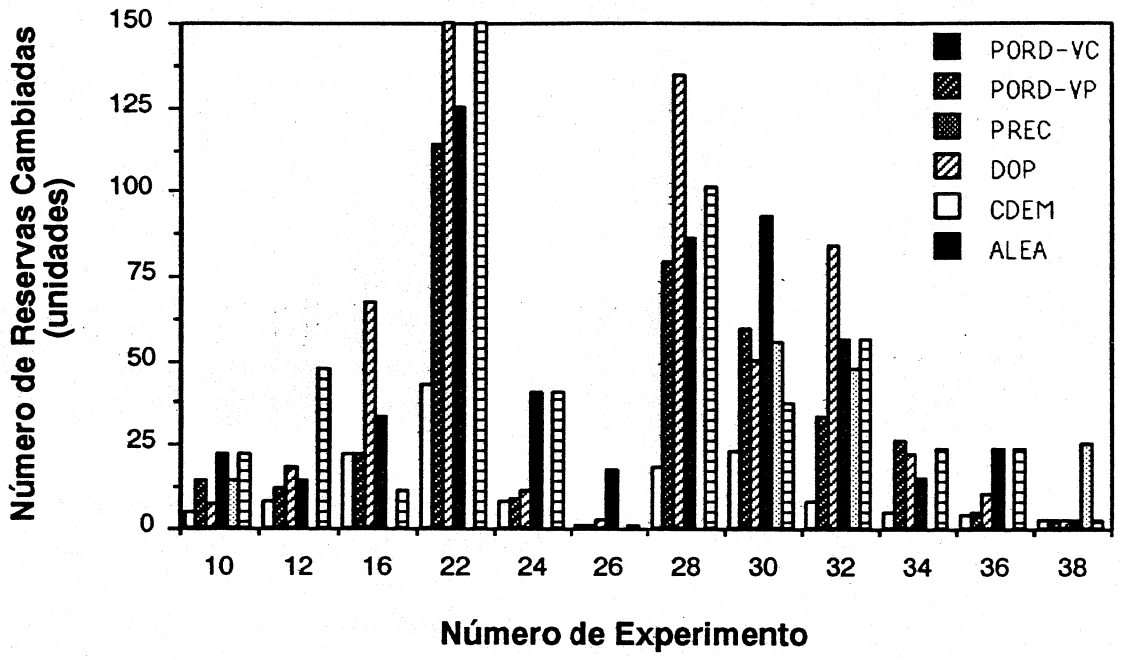

Figura 7.6: Número de reservas cambiadas por experimento y fuente de conocimiento.

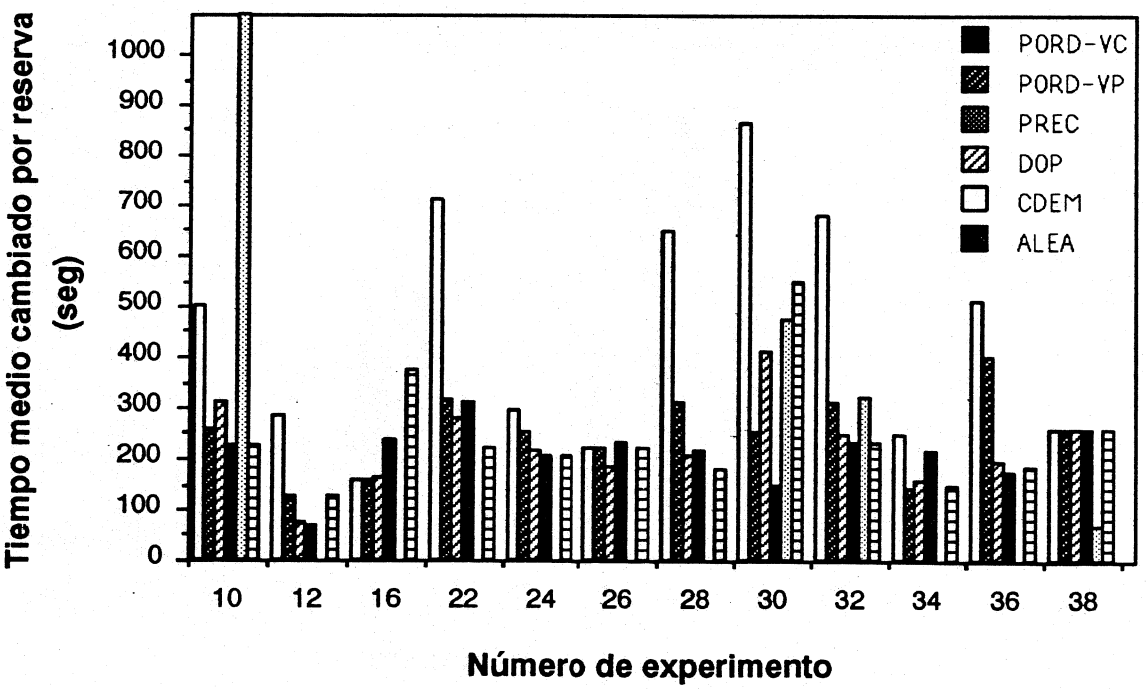

Figura 7.7: Tiempo medio cambiado por reserva por experimento y fuente de conocimiento.

$i \in \{10, 12, 16, 22, 24, 26, 28, 30, 32, 34, 36, 38\}$ es el número de experimento representado,

$a \in \{PORD\text{-}VP\ ,\ PORD\text{-}VP\ ,\ PREC\ ,\ DOP\ ,\ CDEM\ ,\ ALEA\}$ es la acción realizada,

X_i^a es el valor experimental en el experimento i al aplicar la acción a,

Med_X es la media de todos los valores y

Var_X es la varianza.

2. *Cálculo de la medida normalizada para el conjunto de los criterios.* Consiste en un sumatorio ponderado de las medidas normalizadas respecto a cada uno de los criterios.

   Para la distribución de pesos a dar a cada uno de los criterios se dió una importancia del 75% a los criterios relacionados con la calidad del plan de operaciones y el 25% a los relacionados con su disrupción. La tabla siguiente muestra los distintos pesos que se han adoptado empíricamente.

Criterio	*COT*	*CTT*	*CTP*	*NOC*	*NRC*	*TMCR*
Peso	0.1	0.6	0.05	0.1	0.05	0.1

La figura 7.8 recoge los resultados de conjunto normalizados para cada experimento y para cada acción de planificación. Dicha figura pone de manifiesto cual ha resultado ser la mejor política de acción para cada experimento reactivo. Como datos importantes cabría destacar que en el 60% de los casos, la decisión proporcionada utilizando el metaconocimiento del sistema, produjo los mejores resultados, y en el 88% de los casos la decisión del sistema produjo unos resultados dentro del 7% de los resultados obtenidos por la mejor política. A la vista de estos resultados se podría concluir la bondad del metaconocimiento de control utilizado por el planificador y los parámetros utilizados para la caracterización de los conflictos del plan de operaciones. Sin embargo es de destacar también que existe un extenso margen, superando los porcentajes mencionados, para mejorar la aplicación de acciones. Si bien este hecho se justifica, debido a la extraordinaria complejidad que supone el desarrollo de una caracterización de conflictos completa.

7.3 Modulo de Simulación

Un objetivo importante en la elaboración de una herramienta de desarrollo, es construir un lenguaje que pueda soportar múltiples funciones. Una función particularmente importante en un sistema de fabricación integrada por computador, es la posibilidad de simular el sistema de fabricación para poder analizar el

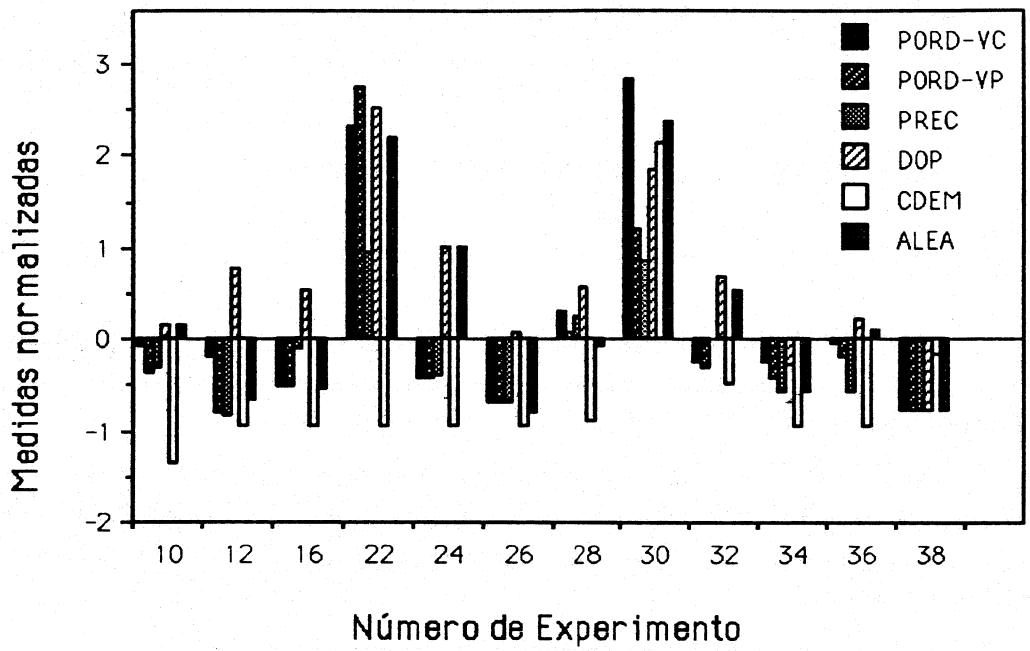

Figura 7.8: Tabla general de resultados normalizados por acción de planificación y experimento.

comportamiento de actuales y futuros modos de producción bajo diferentes condiciones [CROO 85].

Existen varios tipos de simulaciones que tienen considerable importancia práctica, entre ellas se incluyen simulaciones por *eventos discretos*, *continua* y *Monte Carlo*. La simulación por eventos discretos ha sido utilizada extensivamente en aplicaciones de gestión de fabricación, sistemas de inventarios, sistemas de conmutación de mensajes, y redes de transporte, distribución y comunicación. Con este fin se encuentran disponibles lenguajes de simulación tales como: GPSS, SLAM, SIMULA, SIMAN, SIMFACTORY, SIMSCRIPT [SHAN 83,BANK 85].

Estos lenguajes permiten construir modelos, realizar simulaciones y proporcionar información estadística. No obstante, se les critica su falta de capacidad para proporcionar una comprensión profunda del comportamiento del modelo [SATH 86].

La IA ha permitido abordar con relativo éxito el tratamiento de estos problemas desde otra perspectiva, se ha aprovechado la sinergia entre la simulación de eventos discretos y las aproximaciones adoptadas por los programadores que desarrollan software para IA, lo que ha dado lugar a los sistemas de simulación basados en el conocimiento. Estas herramientas se basan en la incorporación de esquemas de representación del conocimiento de IA en modelos de simulación. Su mayor logro ha consistido en facilitar la organización de la información sobre la naturaleza de los objetos y el aumento de la velocidad en la construcción de nuevos

modelos (prototipado rápido) [GARZ 86]. A este respecto, habría que destacar el sistema KBS [REDD 86] caracterizado por la posibilidad de modelar organizaciones complejas, reconocer relaciones causa-efecto y generar escenarios automáticamente.

7.3.1 Objeto simulador

La utilización de KRON para el modelado de un sistema, permite utilizar el mecanismo de control de la red KRON para provocar la ejecución simulada del sistema y con ello la evolución de las actividades de producción descritas en el modelo.

La simulación es responsabilidad de un monitor de ejecución, implementado en modo orientado a objeto, representado por un objeto prototipo denominado *simulacion-sistema.* Los sistemas de simulación de eventos discretos necesitan una *agenda* para conservar los eventos de simulación pendientes, que determinan la secuencia de cambios de estado. Entre eventos, el estado del sistema permanece inmutable, nada relevante ocurre. La ocurrencia de un evento de simulación está representada por una noticia de evento. Noticias de eventos representando eventos futuros están almacenados en la *agenda* y ordenados por su tiempo de ocurrencia (a igualdad de tiempo por prioridad del evento) y están representados mediante objetos denominados *evento-noticia* que portan la siguiente información:

- *Tiempo*: tiempo del evento.
- *Relativo*: "t" si el tiempo está referido al tiempo actual, "nil" para tiempo absoluto.
- *Prioridad*: prioridad del evento (respecto a otros eventos ejecutados al mismo tiempo).
- *Objeto*: objeto a enviar el mensaje.
- *Mensaje*: nombre del mensaje.
- *Parametros*: otros parámetros adicionales.

La evolución de la simulación está determinada por un reloj que el simulador utiliza asíncronamente con una estrategia para el avance del tiempo del modelo denominada *siguiente-evento.* El valor del reloj salta al tiempo del siguiente evento de la agenda, una vez se ha establecido que no puede ejecutarse ninguna acción más en dicho instante de tiempo.

7.3.2 Estructura del modelo de simulación

La estructuración del modelo puede verse como una colección de objetos interaccionando y moviendose a través de secuencias de acciones. La red KRON del modelo permite formalizar la descripción de estas acciones y proporciona la base para el proceso de simulación.

Las posibles acciones físicas que pueden llevarse a cabo en la planta, a un cierto nivel de abstracción, están representadas por los objetos de acción a ese nivel de abstracción. Por tanto, los posibles eventos estarán asociados a dichos objetos de acción.

El disparo de los objetos de acción marca las posibilidades de evolución del modelo. Siguiendo la aproximación propuesta en [VALE 87], se pueden distinguir dos tipos de noticias de eventos asociados con los objetos de acción:

1. *Noticias de eventos internos*: Están relacionados con objetos de acción cuyo disparo puede ser decidido atendiendo exclusivamente al estado de la red y la política de control asociada al objeto de acción (p.e. indicación de comienzo de una operación).

2. *Noticias de eventos externos*: Están relacionados con objetos de acción cuyo disparo depende de la recepción de un mensaje de validación desde el entorno (p.e. indicación de fin de ejecución de una operación). La noticia del evento a poner en la agenda indica el mensaje de validación externa, que se envía al módulo de monitorización. La recepción del mensaje provoca el disparo del objeto de acción. Sin embargo, con el objeto de detectar errores, la función del interpretador de la red comprueba previamente su sensibilización.

Las noticias de eventos internos son meras deducciones de control y señales enviadas al entorno, por lo que son generados automáticamente en el proceso de interpretación de la red. La ejecución del evento provoca el envío del mensaje *revisar-accion* al objeto de acción.

Las relativas a eventos externos pueden ser generadas de dos formas atendiendo al agente introductor:

- *Introducidas por el usuario* en las especificaciones de diseño del experimento relativas a condiciones del entorno físico (p.e rotura de una máquina o generación de nuevas órdenes de proceso) o modificaciones en los objetivos (p.e. modificación de una fecha debida). El instante del evento y sus características son especificados por el usuario.

- *Introducidas automáticamente por el gestor del simulador* durante el proceso de interpretación de la red. Se generan al disparar objetos de acción

pertenecientes a planes de proceso, el instante del evento se calcula a partir de la especificación de la duración, slot *espec-duracion*, en el objeto de la operación.

La ejecución del evento externo se realiza mediante el envío del mensaje respectivo (*fin-de-operacion, rotura-de-maquina, nueva-orden-introducida, nueva-fecha-debida-para-orden, etc.*) a un objeto especial llamado *interfase-entorno*, que genera los eventos de interfase apropiados y los envía al módulo de monitorización.

7.3.3 Ejecución de la Simulación

Las características de la simulación a realizar están descritas en un objeto denominado *especificacion-experimento*. Las funciones de interfase para la introducción de los parámetros y la adaptación del modelo de acuerdo a los parámetros son accesibles mediante el envío de mensajes a dicho objeto. El proceso de simulación comienza adecuando el modelo a las especificaciones de la simulación, la descripción resultante permanece fija durante la ejecución completa del experimento.

7.3.4 Funciones Estadísticas del Simulador

Los valores de los parámetros utilizados en la descripción de las actividades de producción y la propia ejecución del proceso fabricación son de naturaleza estocástica, esto motiva la necesidad de incluir en el simulador un conjunto de funciones estadísticas. El simulador permite que el sistema adapte internamente sus parámetros dirigido por muestras tomadas de distribuciones de probabilidad. El interfase permite al usuario elegir selectivamente entre una variedad de distribuciones (constante, normal, normal truncada, lognormal, uniforme, exponencial, triangular, poisson, etc.).

Distribuciones relacionadas con operaciones y órdenes

La asignación de distribuciones de probabilidad para las operaciones puede hacerse a dos niveles en la jerarquía de representación: nivel de prototipo y nivel de instancia. Para ambas, el simulador dispone de la opción de modificar todas las operaciones o selectivamente operaciones particulares. Pueden asociarse distribuciones sobre dos parámetros de las operaciones: tiempos de proceso, y condiciones de terminación (para operaciones de control de calidad).

Asimismo, el usuario debe tener la posibilidad de generar aleatoriamente distintos parámetros característicos de las órdenes: tasa de llegada de nuevas órdenes, tiempo

{*parametros-rotura-recurso*
is-a : objeto
parametros-rotura-de :
distrib-prevision-rotura :
descripcion : probab. de tiempo para la rotura
distrib-probabilidad-rotura-por-turno :
descripcion : probab. de rotura en un turno de trabajo
distrib-duracion-rotura :
descripcion : tiempo estimado de duracion de la rotura
distrib-perdida-capacidad :
descripcion : distrib. de la perdida de capacidad estimada
distrib-trabajo-dejado-de-hacer :
descripcion : estimacion del trabajo dejado de hacer durante la rotura }

Figura 7.9: Objeto representativo de los parametros de rotura de un recurso

de liberación de la órden[2], y fecha debida de la órden.

Distribuciones relacionadas con recursos

Un factor de incertidumbre relacionado con los recursos y de gran importancia a la hora de la planificación y el control de la producción es la posibilidad de rotura o caída de capacidad de los recursos. El sistema debe disponer de la posibilidad de introducir noticias de eventos de rotura de recurso con gran flexibilidad para poder comprobar la sensibilidad de las respuestas del sistema de control. Para ello, cada recurso dispone de un objeto llamado *parametros-rotura-recurso* que guarda las distribuciones de probabilidad relativas a la caracterización de su rotura (figura 7.9).

La introducción de una rotura de máquina en el simulador se recoge como noticia de evento y puede generarse siguiendo diferentes condiciones:

- *Condición de tiempo*: el usuario especifica un instante exacto para que ocurra la rotura.
- *Previsión de rotura*: el instante de la rotura se calcula al comienzo de la simulación utilizando la distribución de previsión de rotura.
- *Condición aleatoria*: la ocurrencia de la rotura del recurso se calcula en cada turno.

[2]"Release time" según su denominación inglesa.

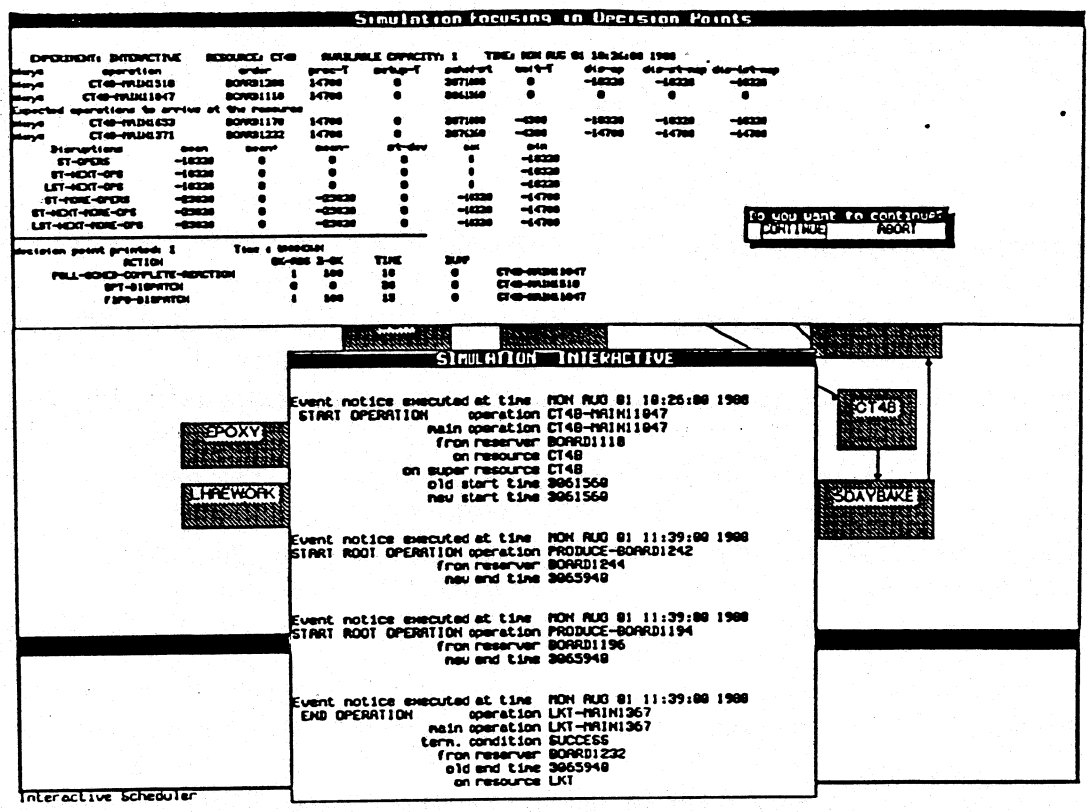

Figura 7.10: Ejemplo de información mostrada por el simulador en tiempo de ejecución.

- *Condición de cuello de botella*: cada turno se examina si el recurso está en alta contención (relacionando las operaciones en cola y la capacidad del recurso).

La rotura de un recurso se informa a través de una noticia de *evento externo*, para su ejecución el manager de la simulación envía un mensaje de rotura al objeto *interfase-entorno*, el cual proporciona la información apropiada al módulo de monitorización.

7.3.5 Otras características

El usuario de la simulación dispone de varios controles con los que puede adaptar la ejecución de la simulación a sus necesidades para producir un entorno particular de producción (especificación de parámetros del proceso, políticas de control), controlar los parámetros de la simulación (por pasos, por intervalos, tiempo, etc.) y especificar secuencias de experimentos. Adicionalmente, la interfase con el ususario debe ser muy potente para poder aprovechar toda la información fruto de la simulación, la figura 7.10 muestra un informe de ejecución.

7.4 Estudio Experimental de Políticas de Control por Simulación

En los experimentos presentados en anteriormente se trataba de comprobar la capacidad reactiva del planificador de operaciones. Para ello, se partía de un plan de operaciones en el que se introducía un conflicto que quedaba reflejado como una violación de restricciones. A partir de esta nueva situación se comprobaba las características del nuevo plan generado por el planificador. El distribuidor de operaciones debe trabajar, en cambio, en condiciones más cercanas al tiempo real, donde es necesario dar respuestas en un plazo de tiempo más reducido. En su proceso de toma de decisiones, el distribuidor puede hacer uso de la información proporcionada por el nivel superior del sistema de decisión, para lo cual dispone de las apropiadas políticas de control. El objetivo de esta segunda fase de experimentos es comprobar las ventajas que supone la utilización de estas políticas de control, que realizan una interpretación del plan de operaciones elaborado por el planificador.

Utilizando el simulador esbozado en la sección anterior, se han completado una nueva serie de experimentos, para las que se han recogido las prestaciones de diversas políticas de control usadas comúnmente en sistemas tiempo real existentes. Se han introducido también variantes de estas políticas para incluir características que recojan interpretaciones de un plan de operaciones elaborado previamente.

El modelo de sistema de fabricación para estos experimentos, es el mismo al utilizado para los experimentos anteriores. En los planes de operaciones se omitieron ciertas secuencias lineales de operaciones para acortar los grafos de proceso, pero todavía capturan la esencia de las características generales de flujo del sistema de fabricación. Para aumentar la carga de los recursos, sin aumentar innecesariamente las simulaciones, ha sido reducida también su capacidad.

7.4.1 Diseño de los experimentos

Los experimentos se basan en la simulación del procesamiento de 30 órdenes. La fecha más temprana de comienzo y su fecha debida se han calculado siguiendo distribuciones normales. Las órdenes están compuestas por una media de 11.51 operaciones, oscilando entre un mínimo de 7 y un máximo de 17. Este valor depende del resultado de las operaciones de control de calidad y de las consiguientes operaciones de reparación, lo que condujo a una media de 345.5 operaciones simuladas por experimento. Se dispone también, inicialmente, de un plan de operaciones proporcionado por el planificador. La figura 7.11 muestra las características, tanto de las órdenes y recursos en el momento inicial, como del plan de operaciones generado (estadísticas de órdenes y recursos y su gráfico Gantt).

Las políticas de control utilizadas en los experimentos fueron las siguientes:

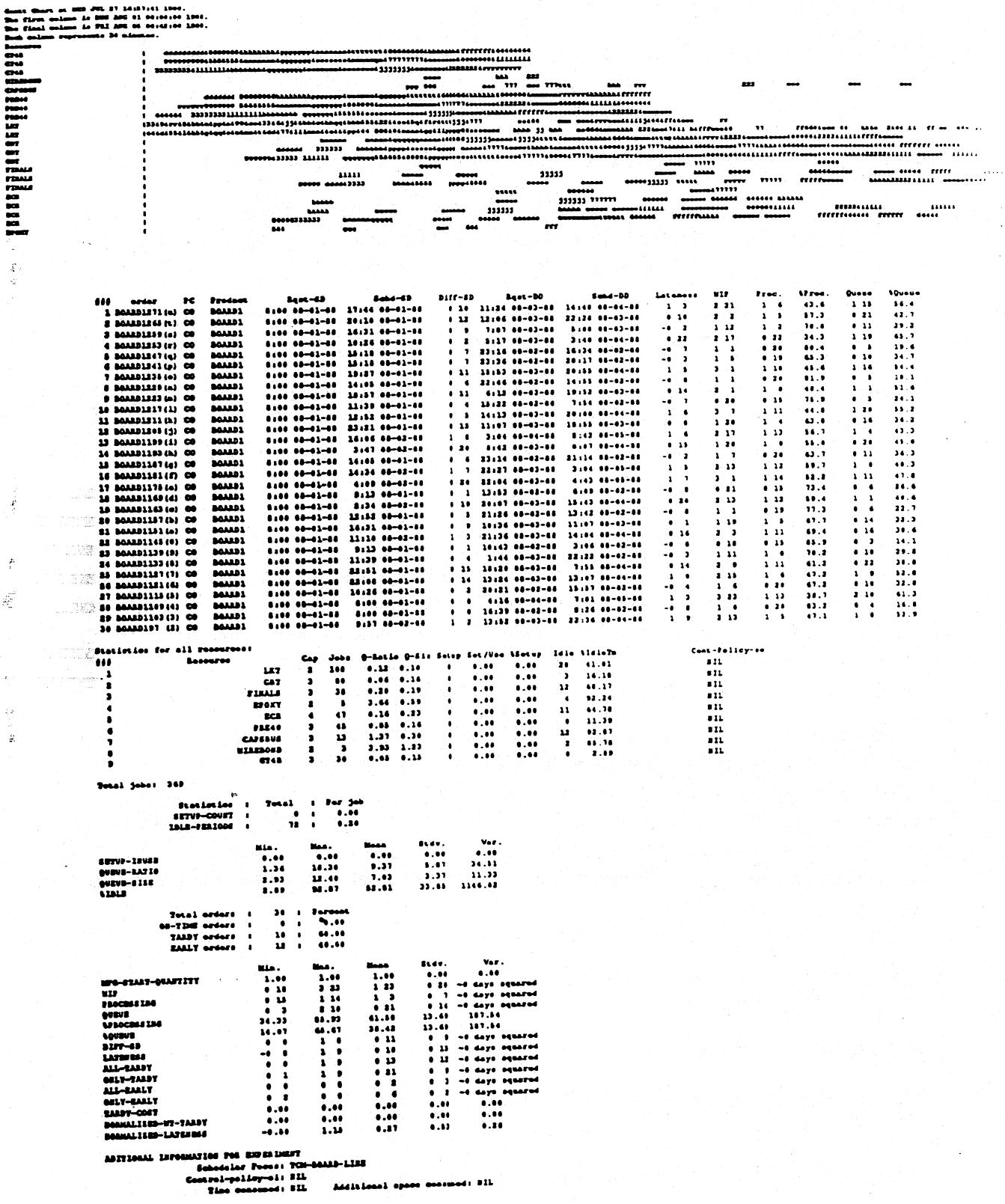

Figura 7.11: Plan de operaciones inicial proporcionado por el planificador.

SPT Elige la operación con tiempo de proceso más corto.

SPT-OI Elige la operación con tiempo de proceso más corto, para el tiempo de comienzo de ejecución de la orden se adopta el instante calculado por el planificador de operaciones.

FIFO Elige la operación que haya permanecido mayor tiempo en la cola de espera.

FIFO-OI Elige la operación que haya permanecido mayor tiempo en la cola de espera, para el tiempo de comienzo de ejecución de la orden se adopta el instante calculado por el planificador de operaciones.

ELST Elige la operación que tiene el tiempo de comienzo inferior según el plan de operaciones generado por el planificador.

ELST-OI Elige la operación que tiene el tiempo de comienzo inferior según el plan de operaciones generado por el planificador, para el tiempo de comienzo de ejecución de la orden se adopta también el instante calculado por el planificador.

Las políticas *SPT* y *FIFO* no utilizan en absoluto la información porporcionada por el planificador, las políticas *SPT-OI* y *FIFO-OI* hacen uso del instante de comienzo de procesamiento de la orden disponible en el plan de operaciones, la política *ELST* contempla la información del plan en las decisiones acerca del comienzo de las operaciones, mientras que la política *ELST-OI* realiza una interpretación más completa utilizando también el instante de comienzo de procesamiento de la orden.

En este caso resulta también difícil la evaluación de prestaciones respecto a las diferentes políticas, debido a la diversidad de criterios a contemplar. La información completa proporcionada por el simulador en cada experimento es similar a la mostrada en la figura 7.11, en las tablas de resultados de la siguiente sección se muestran únicamente los parámetros que consideramos más indicativos (la discriminación entre las diferentes prolíticas se hará contemplando conjuntamente estos parámetros), que son:

TEP Tiempo consumido por una orden desde el comienzo de su procesamiento hasta su finalización.

TCola Suma de los tiempos de espera de todas las operaciones de una orden.

%Proc Relación porcentual entre el tiempo de procesamiento efectivo de una orden y su TEP.

%Cola Relación porcentual entre el *TCola* de una orden y su *TEP*.

Retraso Diferencia algebraica entre el tiempo de terminación de la orden y su fecha debida.

Tarde (T) Diferencia positiva entre el tiempo de finalización de una orden y su fecha debida (cero en caso de ser negativa). La *T* indica la consideración de todas las órdenes.

Tarde (S) Diferencia positiva entre el tiempo de finalización de una orden y su fecha debida (cero en caso de ser negativa). La *S* indica la consideración únicamente de las órdenes tardías.

Pronto (T) Diferencia positiva entre la fecha debida de una orden y su tiempo de finalización (cero en caso de ser negativa). La *T* indica la consideración de todas las órdenes.

Pronto (S) Diferencia positiva entre la fecha debida de una orden y su tiempo de finalización (cero en caso de ser negativa). La *S* indica la consideración únicamente de las órdenes tempranas.

Los parámetros *TEP, TCola, %Proc* y *%Cola* están relacionados con costes por material en proceso y los parámetros *Retraso, Tarde (T), Tarde (S), Pronto (T)* y *Pronto (S)* se refienren a costes por no cumplimiento de fechas debidas y costes de almacén.

7.4.2 Resultados experimentales

En el primer grupo de resultados experimentales que se presenta, se ha tratado de comprobar el comportamiento de las diferentes políticas de control al introducir cambios en las condiciones de terminación de las operaciones. En estos experimentos, las condiciones de terminación no siguen los pronósticos contemplados por el plan de operaciones sino que son evaluadas nuevamente al concluir cada operación de control de calidad. Cada vez que la condición de terminación generada no coincide con la pronosticada por el plan, se produce una secuencia que se diferencia parcialmente de la pronosticada. El objetivo es comprobar en que medida, las políticas que hacen una interpretación del plan, llevan a mejores resultados aún siguiendo planificaciones que no coinciden exactamente con las reales.

Las tablas 7.1 y 7.2 reflejan los resultados experimentales. En la primera se ha utilizado el plan de operaciones tal como ha sido generado por el planificador, mientras que en la segunda se han eliminado del plan todas las operaciones que corresponden a subsecuencias de reparación. En general se establece una clara diferencia entre las políticas que no contemplan el tiempo de comienzo de procesamiento de la orden y sus correspondientes que si lo hacen (acabadas en *-OI*). Por otra parte, parece intuirse de los resultados alguna mejora de las políticas del tipo *ELST* con respecto al resto, fundamentada en un mejor porcentaje de tiempo de procesamiento y, por tanto, un menor porcentaje de tiempo de espera, así como ciertas mejoras (en este caso menos claras) en relación con los parámetros relativos a costes por entrega tardía y costes de almacén.

		SPT	*SPT-OI*	*FIFO*	*FIFO-OI*	*ELST*	*ELST-OI*
TEP	med	1 20	1 13	2 2	1 12	1 12	1 12
(dias horas)	dt	0 16	0 15	0 12	0 9	0 11	0 16
TCola	med	0 21	0 14	1 3	0 12	0 13	0 13
(dias horas)	dt	0 14	0 13	0 8	0 4	0 8	0 14
%Proc	med	57.75	68.62	46.68	65.28	66.37	71.76
(%)	dt	19.64	19.82	7.45	9.93	14.76	20.89
%Cola	med	42.25	31.38	53.32	34.72	33.63	28.24
(%)	dt	19.64	19.82	7.45	9.93	14.76	20.89
Retraso	med	0 5	0 0	0 10	-0 1	0 3	0 12
(dias horas)	dt	0 18	0 18	0 21	0 13	1 0	0 10
Tarde (T)	med	0 10	0 7	0 15	0 5	0 11	0 1
(dias horas)	dt	0 12	0 10	0 16	0 6	0 16	0 21
Tarde (S)	med	0 17	0 13	0 23	0 9	0 21	0 9
(dias horas)	dt	0 11	0 10	0 15	0 6	0 16	0 12
Pronto (T)	med	0 5	0 7	0 4	0 5	0 9	0 20
(dias horas)	dt	0 8	0 11	0 7	0 10	0 12	0 11
Pronto (S)	med	0 12	0 16	0 1	0 11	0 19	0 9
(dias horas)	dt	0 7	0 12	0 8	0 11	0 11	0 11

Tabla 7.1: Experimentos con incertidumbre en las condiciones de terminación[3].

		SPT	*SPT-OI*	*ELST*	*ELST-OI*
TEP	med	1 20	1 13	1 12	1 12
(dias horas)	dt	0 16	0 15	0 14	0 14
TCola	med	0 21	0 14	0 13	0 13
(dias horas)	dt	0 14	0 13	0 9	0 12
%Proc	med	57.75	68.62	66.81	69.28
(%)	dt	19.64	19.82	13.82	19.79
%Cola	med	42.25	31.38	33.19	30.72
(%)	dt	19.64	19.82	13.82	19.79
Retraso	med	0 5	0 0	0 3	0 1
(dias horas)	dt	0 17	0 18	0 23	0 19
Tarde (T)	med	0 10	0 8	0 11	0 9
(dias horas)	dt	0 12	0 10	0 15	0 10
Tarde (S)	med	0 17	0 13	0 20	0 16
(dias horas)	dt	0 11	0 10	0 15	0 9
Pronto (T)	med	0 5	0 7	0 8	0 16
(dias horas)	dt	0 7	0 10	0 11	0 9
Pronto (S)	med	0 12	0 17	0 17	0 8
(dias horas)	dt	0 7	0 10	0 10	0 11

Tabla 7.2: Experimentos con incertidumbre en las condiciones de terminación.

		SPT	*SPT-OI*	*FIFO*	*FIFO-OI*	*ELST*	*ELST-OI*
TEP	med	2 8	2 1	2 18	1 21	2 6	1 21
(dias horas)	dt	0 19	0 18	0 13	0 16	0 12	0 20
TCola	med	1 5	0 22	1 15	0 18	1 4	0 18
(dias horas)	dt	0 18	0 17	0 9	0 11	0 16	0 21
%Proc	med	53.38	60.52	40.63	62.82	52.92	69.67
(%)	dt	19.85	21.44	8.84	12.72	21.38	25.10
%Cola	med	46.54	39.39	59.33	37.05	47.01	30.29
(%)	dt	19.94	21.45	8.80	12.88	21.48	25.06
Retraso	med	0 17	0 12	1 2	0 9	0 17	0 11
(dias horas)	dt	0 18	0 18	0 15	0 16	1 2	0 22
Tarde (T)	med	0 18	0 15	1 3	0 12	0 22	0 15
(dias horas)	dt	0 16	0 15	0 15	0 12	0 20	0 18
Tarde (S)	med	0 23	0 22	1 4	0 20	1 17	1 0
(dias horas)	dt	0 15	0 13	0 13	0 9	0 18	0 17
Pronto (T)	med	0 1	0 3	0 0	0 3	0 4	0 4
(dias horas)	dt	0 3	0 6	0 1	0 5	0 8	0 7
Pronto (S)	med	0 7	0 9	0 2	0 8	0 14	0 11
(dias horas)	dt	0 3	0 8	0 1	0 6	0 8	0 7

Tabla 7.3: Experimentos con incertidumbre en los tiempos de proceso desviación típica del 5%.

El segundo grupo de experimentos tenía por objeto comprobar la sensibilidad de las diferentes políticas con respecto a incertidumbres asociadas a la información sobre los tiempos de proceso de las operaciones. El planificador de operaciones trabaja con información aproximada sobre los tiempos de proceso, que no tiene por que coincidir exactamente con los valores reales, así el plan realizado con estos valores aproximados no podrá seguirse de forma exacta. Estos experimentos tratan de poner de manifiesto la utilidad de la interpretación del plan de operaciones aún cuando se realize a partir de valores imprecisos de los tiempos de procesamiento.

Los tiempos de proceso de las operaciones en los experimentos se han evaluado considerando una distribución normal, cuya media era la información disponible sobre el tiempo de proceso de cada operación y la desviación típica se daba en relación a dicho tiempo. Las tablas 7.3 y 7.4 muestran los resultados para desviaciones típicas del 5% y del 1% de la media del tiempo de proceso respectivamente.

En este caso resulta también clara la superioridad de las políticas que hacen una interpretación del plan de operaciones, destaca particularmente la mejora del porcentaje del tiempo de procesamiento de las órdenes proporcionada por la política *ELST-OI* (69.67% y 68.29%) con respecto a los porcentajes más próximos (62.82% y 56.25%) respectivamente.

		SPT	*ELST*	*ELST-OI*
TEP	med	2 6	2 6	1 22
(dias horas)	dt	0 21	0 11	0 20
TCola	med	1 3	1 4	0 19
(dias horas)	dt	0 18	0 16	0 21
%Proc	med	56.25	52.80	68.29
(%)	dt	21.01	20.82	24.92
%Cola	med	44.27	47.28	31.67
(%)	dt	19.87	21.16	25.32
Retraso	med	0 15	0 17	0 10
(dias horas)	dt	0 18	1 1	0 22
Tarde (T)	med	0 17	0 21	0 15
(dias horas)	dt	0 15	0 20	0 17
Tarde (S)	med	1 0	1 6	1 0
(dias horas)	dt	0 13	0 17	0 16
Pronto (T)	med	0 2	0 4	0 4
(dias horas)	dt	0 4	0 8	0 7
Pronto (S)	med	0 7	0 14	0 11
(dias horas)	dt	0 4	0 8	0 7

Tabla 7.4: Experimentos con incertidumbre en los tiempos de proceso desviación típica del 1%.

7.5 Conclusiones

Este capítulo describe un análisis experimental de la arquitectura de toma de decisiones de control, propuesta en los capítulos anteriores. Se han realizado dos tipos de experimentos para analizar los dos componentes del sistema de toma de decisiones. La mayor dificultad de estos tests proviene de la imposibilidad de realizar un análisis experimental completo, debido la extraordinaria cantidad de parametros posibles y de criterios de evaluación.

La primera parte de los experimentos se centra en la capacidad reactiva del planificador de operaciones, el objetivo final es comprobar la validez del metaconocimiento de control que es utilizado por el planificador, respecto a otro tipo de políticas que suponen aplicaciones más "ciegas" de las fuentes de conocimiento (hay que resaltar que la propia fase de experimentación ha servido de realimentación para la generación de las heurísticas de metaconocimiento). La segunda parte de los experimentos se concentran en el distribuidor de operaciones. Su objetivo es analizar la utilidad de políticas de control que adoptan criterios basados en la interpretación del plan de operaciones.

Dentro de este marco experimental relativo y limitado, se aprecia claramente la superioridad de la estructura de metaconocimiento utilizada y también, parece deducirse de los resultados ciertas ventajas en la utilización de políticas inteligentes de interpretación del plan.

LA FOTO SALIO MOVIDA

Un cronopio va a abrir la puerta de calle, y al meter la mano en el bolsillo para sacar la llave lo que saca es una caja de fósforos, entonces este cronopio se aflige mucho y empieza a pensar que si en vez de la llave encuentra los fósforos, sería horrible que el mundo se hubiera desplazado de golpe, y a lo mejor si los fósforos están donde la llave, puede suceder que encuentre la billetera llena de fósforos, y la azucarera llena de dinero, y el piano lleno de azucar, y la guía del teléfono llena de música, y el ropero lleno de abonados, y la cama llena de trajes, y los floreros llenos de sábanas, y los tranvías llenos de rosas, y los campos llenos de tranvías. Así es que este cronopio se aflige horriblemente y corre a mirarse al espejo, pero como el espejo está algo ladeado lo que ve es el paraguero del zaguán, y sus presunciones se confirman y estalla en sollozos, cae de rodillas y junta sus manecitas no sabe para que.

Historias de cronopios y famas
JULIO CORTAZAR

Capítulo 8

Conclusiones

El trabajo desarrollado en esta memoria intenta dar respuestas a una parte de los problemas que plantea el diseño de sistemas de control de sistemas de fabricación. En concreto, se ha abordado el problema de modelar estos sistemas de control utilizando una herramienta válida, tanto para el diseño del subsistema de control de planta, como de los subsistemas de decisión "tiempo real" y de planificación. Para ello se ha propuesto una solución en la que se integran técnicas de representación del conocimiento utilizadas en IA; redes de Petri de alto nivel para describir aspectos dinámicos (especialmente relativos al carácter concurrente de un sistema de fabricación); y la programación orientada a objeto.

A continuación se resumen los aspectos desarrollados en esta memoria que, a nuestro entender, constituyen aportaciones:

- La progresión hacia una profunda integración de las redes de Petri de alto nivel en entornos de representación del conocimiento, en concreto, entornos basados en frames y orientados a objeto. Nociones como arcos, lugares, transiciones, colores, conjuntos de colores, etc., son traducidos a conceptos como relaciones, restricciones, objetos, atributos, etc., de manera que pueden ser contemplados de la misma forma que el resto de características que definen los objetos en sistemas tipo frame. Adicionalmente, la representación posibilita la construcción de jerarquías de especialización de objetos, con la consiguiente jerarquía de especialización de redes de Petri de alto nivel. En esta jerarquía, el comportamiento dinámico de objetos más especializados puede heredarse de objetos menos especializados sucesores en la jerarquía. Desde su consideración como herramienta de modelado de sistemas de eventos discretos, KRON soporta criterios de generalidad, accesibilidad y extensibilidad.

- El planteamiento de KRON como una herramienta de modelado con una visión unificada RdP/IA. Un modelo KRON puede ser observado indistintamente desde estas dos perspectivas. Un objeto de acción de un modelo KRON

puede contemplarse como una transición en una red de Petri o como una regla precondición/postcondición en un sistema basado en reglas o un operador en un espacio de búsqueda. Valores en un atributo de estado pueden verse como marcas en un lugar o aserciones lógicas y restricciones según la perspectiva adoptada. Esta visión unificada permite la transferencia de resultados y conceptos entre ambos tipos de técnicas con el consiguiente enriquecimiento mutuo.

- La generalidad de KRON como herramienta de representación permite su extensión al nivel de aplicación y, concretamente, a sistemas de fabricación. Se proponen primitivas para el modelado de conceptos específicos como recursos, operaciones, planes de trabajo, etc., y se presentan métodos para su interconexión y la sincronización de sus acciones. Igualmente se estructuran mecanismos para la construcción de jerarquías de abstracción de modelos de sistemas de fabricación.

- Otro aspecto desarrollado es el concerniente a la metodología de modelado y de razonamiento, principalmente en el nivel del dominio. Se fundamenta en la contemplación oportunista de un sistema de fabricación desde dos perspectivas de recursos y operaciones. El proceso de construcción del modelo y las inferencias para el control del sistema, se orientan según la perspectiva que, en cada momento, ofrece más posibilidades para la representación y solución del problema. En la etapa de modelado, la conexión de perspectivas se lleva a cabo a través de sincronizaciones especiales entre los planes de procesos y el modelo físico de la planta. En la etapa de razonamiento en la toma de decisiones, la conexión se realiza por medio de restricciones (fundamentalmente temporales y de secuencia).

- Se ha presentado una nueva base para la parametrización de los planes de operaciones y una metodología para representar y gestionar el conocimiento heurístico para la generación y revisión de dichos planes.

- Se ha mostrado la utilidad de la interpretación del plan de operaciones elaborado a un nivel superior en la jerarquía de control, como un método para encontrar soluciones rápidas en entornos de sistemas de fabricación complejos y cambiantes.

- Otro aspecto es el relacionado con los ciclos de los distintos módulos que componen el sistema de control. La idea subyacente para todos ellos es la constitución de un ciclo con realimentación por eventos. Planificador, distribuidor y coordinador pueden recibir información interna y externa representada por eventos, en base a estos eventos, cada módulo puede realizar las acciones reactivas oportunas.

- Es posible extender las ideas desarrolladas en KRON a otros dominios. Aunque inicialmente estas ideas han surgido para resolver problemas en el

dominio de fabricación, son aplicables a otros dominios de naturaleza dinámica concurrente en los que se requieran herramientas para la representación del conocimiento y deban realizarse tareas de planificación y decisión.

Durante la realización del trabajo que aquí se presenta, han ido surgiendo nuevas posibilidades de trabajo que podrían constituir una prolongación de esta tesis:

En relación con la herramienta de representación de base hay prolongaciones naturales como la integración en KRON de interpretaciones temporizadas y estocásticas de redes de Petri. Otra área importante es la profundización en las características analíticas proporcionadas por la red. Al nivel del dominio, puede resultar interesante el trabajo sobre jerarquías, la consideración de objetos con buenas propiedades estructurales que faciliten la tarea de prototipado rápido, y la realización de procedimientos de diagnosis en la etapa de monitorización a partir de la estructura de red.

Las áreas futuras de trabajo relacionadas con la toma de decisiones de control se pueden agrupar en cuatro apartados:

1. Planificación de operaciones reactiva: validación del modelo de selección de acciones reactivas, consideración de reacciones compuestas y valoración de la eficiencia de las reacciones.

2. Relación con el sistema de coordinación de planta: profundizar en el concepto de "interpretación del plan de operaciones" en tiempo de ejecución y estudiar la robusticidad del plan de operaciones con relación a la política de control.

3. Arquitecturas de planificación de operaciones descentralizadas: distribución de responsabilidad estructural y funcional y comunicación/negociación de restricciones vertical y lateral.

4. Adquisición automática de la experiencia de planificación.

Bibliografía

[ACOC 86] Acock M. and Zemel R.: "DISPATCHER: AI Software for Automated Material Handling Systems. *ULTRATECH-AI in Manufacturing Conference*, Society of Manufacturing Engineers, Long Beach, CA, Sept. 1986.

[ALAN 84] Alanche P., Benzakour K., Dolle F., Gillet P., Rodrigues P. and Valette R.: "PSI: a Petri net based simulator for flexible manufacturing systems." *Advances in Petri nets 1984, LNCS 188, pp. 1-14*. Springer Verlag. 1984.

[ALJA 87] Al-Jaar R.Y. and Desrochers A.A.: "A survey of Petri Nets in Automated Manufacturing Systems." *Procs. IMACS 1988, Vol 3*, Paris, Julio 1988. Una versión extendida se encuentra también como: "Petri Nets in Automation and Manufacturing" *Technical Report, RAL 99, Department of Electrical, Computer and Systems Engineering*, Rensselaer Polytechnic Institute, Troy, NY. Nov. 1987.

[ALLA 82] Alla H., Ladet P., Martínez J. and Silva M.: "Modelling and Validation of Complex Systems by Colored Petri Nets: Application to a Flexible Manufacturing System". *5nd European Workshop on Petri Net Applications and Theory, pp.122-140*, Aarhus, . June, 1984.

[ALLE 84] Allen J.F.: "General Theory of action and time." *Artificial Intelligence, vol. 23, no. 2*, July. 1984.

[ATAB 87] Atabakhche H.:"Utilisation Conjointe de líntelligence Artificielle et des Reseaux de Petri: Application au Controle déxecution dún Plan de Fabrication". *These Docteur de lúniversite Paul Sabatier de Toulouse.* 1987.

[BAKE 74] Baker K.R.: *Introduction to Sequencing and Scheduling.* John Wiley & Sons, Inc., NewYork. 1974.

[BALD 87] M. Baldassari and G. Bruno: "An Environment for Object-Oriented Conceptual Programming Based on PROT Nets." *Technical Report, Dipartimento di Automatica e Informatica,* Politecnico di Torino. 1987.

[BANK 85] Banks J. and Carson J.S.: "Process-Interaction Simulation Languages". *Simulation, Vol. 44, No. 5, pp. 225-235*, May. 1985.

[BARR 82] Barr A. and Feigenbaum E. A.: *The Handbook of Artificial Intelligence, Vol.1*. William Kaufmann, Inc, Los Altos, California. 1982.

[BECK 86] Beck C.L. and Krogh B.H.: "Models for Simulation and Discrete Control of Manufacturing Systems". *Proc. of 1986 IEEE International Conference on Robotics and Automation, pp. 305-310*, San Francisco, CA, April, 1986.

[BENS 86] Bensana E.M., Correge M., Bel G. and Dubois D.: "An Expert-System Approach to Industrial Job-Shop Scheduling". *Proc. of 1986 IEEE International Conference on Robotics and Automation, pp. 1645-1650*, San Francisco, CA, April, 1986.

[BRAC 79] Brachman R.J.: "On the epistemological status of semantic networks." En *Associative Networks, Cap. 1, pp. 3-46*. N. V. Findler, Academic Press. 1979.

[BROW 86] Brownston L., Farrell R., Kant E. and Martin N.: *Programming Expert Systems in OPS5. An Introduction to Rule-Based Programming.* Addison-Wesley Publishing Company, Inc., USA. 1986.

[BRU 86a] Bruno G. and Elia A.:"Operational Specification of Process Control Systems: Execution of Prot Nets Using OPS5". *10th World IFIP Congress, pp. 35-40*. Dublin, Sep. 1986.

[BRU 86b] Bruno G. and Balsamo A.:"Petri Net-Based Object-Oriented Modelling of Distributed Systems". En *Proc. Object-Oriented Programming Systems, Languages and Applications, COPSLA86*, Portland, Oregon. Nov. 1986.

[BRU 86c] G. Bruno and G. Marchetto: "Process-Translatable Petri Nets for the rapid prototyping of process control". *IEEE Trans. on Software Eng., Vol SE-12, No 2, pp. 346-357.* 1986.

[BRUC 72] Bruce B.C.: "A model for temporal references and its application in a question answering program." *Artificial Intelligence, vol. 3, pp. 1-25.* 1972.

[BRUN 86] Bruno G.A., Elia A. and Laface P.: "A Rule-Based System to Schedule Production".*IEEE Computer, Vol. 19, No. 7, pp. 76-86*. 1986.

[BUZA 86] Buzacott J.A. and Yao D.D.: "Flexible manufacturing systems: a review of analytical models". *Management Science, Vol 32, No 7, pp. 890-905*, Jul. 1986.

[CARD 85] Cardelli L. and Wegner P.: "On Understanding Types, Data Abstraction, and Polymorphism". *Computing Surveys, Vol. 17, No. 4, pp. 471-518*, Dec. 1985.

[CAST 85] Castelain E., Corbeel J. and Gentina J.: "Comparative Simulations of Control Processes described by Petri Nets". *Proc. IEEE COMPINT'85 Conf.*, 1985.

[CAST 89] Castelain E., Corbeel D. and Gentina J.C.: "Prototyping of FMS from the Design of a Pregraph Based on Some Extended Petri-Nets". *Proc. of the 6th IFAC Symp. on Information Control Problems in Manufactruing Technology, INCOM'89, pp. 437-442*. Madrid, Sept. 1989.

[CHRE 88] Chretienne P.: "Timed Petri Net Schedules". En *Advances in Petri Nets 1988, LNCS 340, pp. 62-84*, Springer Verlag. 1988.

[COLO 86] Colom J.M., Silva M. and Villarroel J.L.: "On software implementation of Petri nets and colored Petri nets using high-level concurrent languages." *7th European Workshop on Application and Theory of Petri Nets, pp. 207-241*. Oxford, Jul. 1986.

[COLO 87] Colom J.M., Martínez J., Silva M.: "Packages for Validating Discrete Production Systems Modeled with Petri Nets". *Proc. of 1st. IMACS Symposium on Modelling and Simulation for Control of Lumped and Distributed Parameter Systems*. Lille, France. June. 1987.

[CROO 85] Crookall J.R.: "Planning and Simulation of FMS". *Annals of the CIRP, Vol. 34, No. 2, pp. 577-584*. 1985.

[DAR 82] Dar-El E.M. and Karni R.: "A Review of Production Scheduling and its Applications in On-line Hierarchical Computer Control Systems". En *On-line Production Scheduling and Plant-Wide Control, pp. 1-25*, Kompass&Williams. 1982.

[DELG 86] Delgrande J.P. and Mylopoulos J.: "Knowledge Representation: Features of Knowledge". En *Fundamentals in Man-Machine Communication: Speech, Vision and Language*, Haton J.-P. (Ed.), Cambridge University Press, Cambridge. 1986.

[DESC 84] Descotte Y. and Latombe J.-C.: "GARI: An expert system for process planning". En *Solid Modelling by Computers*, Pickett M.S. and Boyse J.W. (Eds.). Plenum Press, New York. 1984.

[DESC 85] Descotte Y. and Latombe J-C.: "Making Compromises among Antagonist Constraints in a Planner". *Artificial Intelligence, Vol. 27, pp. 183-217*. 1985.

[DUBO 83] Dubois D. and Stecke K.E.: "Using Petri Nets to Represent Production Processes". *Proc. 22nd IEEE Conf. Decision and Control, pp. 1062-1067*, Dec. 1983.

[ELGE 88] Elgelmore R.S. and Morgan A.J. (editores): *Blackboard Systems*, Addison-Wesley Publishing Company, Inc. 1988.

[ERMA 80] Erman L.D., Hayes-Roth F., Lesser V.R. and Reddy D.R.: "The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty". *Computing Surveys, Vol. 12, No. 2, pp. 213-253*, June. 1980.

[FEIG 63] Feigenbaum E.A. and Feldman J.: *Computers and thought*. McGraw-Hill, New York. 1963.

[FLEI 89] Fleischhack H. and Weber A.: "Rule Based Programming, Predicate Transition Nets and the Modeling of Office Procedures and Flexible Manufacturing Systems." *9th European Workshop on Application and Theory of Petri Nets, Vol II*. Venice. July. 1989.

[FOX 83] Fox M.S.: "Constraint Directed Search: A Case Study of Job-Shop Scheduling." *PhD dissertation, Computer Science Dep.*, Carnegie Mellon University. Pittsburgh.,PA (USA). 1983.

[FOX 84] Fox M.S. and Smith S.F.: "ISIS a knowledge-based system for factory scheduling". *Expert Systems, Vol. 1, No. 1.* July. 1984.

[FOX 85] Fox M.S.: "Knowledge Representation for Decision Support." En *Knowledge Representation for Decision Support Systems, pp. 3-26*, L.B. Methlie and R.H. Sprague (Eds.), Elsevier Science Publishers, North-Holland. 1985.

[FOX 86] Fox M.S.: "Observations on the Role of Constraints in Problem Solving". *Proc. of the Annual Conference of the Canadian Society for Computational Studies of Intelligence*, Montreal, Quebec. May. 1986.

[FOXa 89] Fox M.S., Sadeh N. and Baykan C.: "Constrained Heuristic Search". *Technical Report, Robotics Institute and Computer Science Department*, Carnegie Mellon University, Pittsburgh, PA. 1989.

[FOXb 89] Fox M.S.: "Second Generation Comercial Applications of Artificial Intelligence". *Technical Report, Robotics Institute and Computer Science Department*, Pittsburgh, PA. 1989.

[GARE 79] Garey M.R. and Johnson D.S.: *Computers and Intractability. A Guide to the Theory of NP-Completeness*. Freeman W.H. and Company, New York. 1979.

[GARZ 86] Garzia R.F., Garzia M.R. and Zeigler B.P.: "Discrete-event simulation". *IEEE Spectrum, Vol.23, No. 12, pp. 32-36*, Dec. 1986.

[GENR 86] Genrich H.J.: "Predicate/Transition Nets". En G. Goos, J. Hartmanis (Eds.), *Petri Nets: Central Models and their Properties, pp. 207-247*, Lecture Notes in Computer Science 254, Springer-Verlag, 1986.

[GENT 87] Gentina J. and Corbeel D.:"Colored Adaptive Structured Petri Net: a tool for the automatic synthesis of hierarchical control of Flexible Manufacturing Systems". En *[IEEE 87], pp. 1180-1185*. 1987.

[GEOR 87] Georgeff M.P.: "An embedded Real-Time Reasoning System". *Technical Report*, Representation and Reasoning Program, Artificial Intelligence Center, Computer and Information Sciences Division. Nov. 1987.

[GERS 86] Gershwin S.B., Hildebrant R.R., Suri R. and Mitter S.K.: "A Control Perspective on Recent Trends in Manufacturing Systems". *IEEE Control Systems Magazine, pp. 3-15*, April. 1986.

[GLYN 89] P.W. Glynn: "A GSMP Formalism for Discrete Event Systems". *Proceedings. of the IEEE, Special Issue on Dynamics of Discrete Event Systems*, Jan. 1989.

[GRAV 81] Graves S.C.: "A review of production scheduling". *Operations Research*, 29(4), pp646-675. 1981.

[HALE 83] Haley P., Kowalski J., McDermott J. and McWhorter: "PTRANS: A Rule-based Management Assistant". *Dept. of Computer Science*, Carnegie Mellon University, Pittsburgh, PA. 1983.

[HAYE 85] Hayes-Roth B.: "A Blackboard Architecture for Control". *Artificial Intelligence, Vol. 26, No. 2, pp. 251-321*. 1985.

[HIGG 90] Higgins P. and Browne J.: "The monitor in production activity control systems". *Production Planning & Control, Vol. 1, No. 1, pp. 17-26.* 1990.

[HO 89] Y.-C. Ho (ed.): "Special Issue on Dynamics of Discrete Event Systems". *Proceedings. of the IEEE*, Jan. 1989.

[HOLL 87] Hollinger D. and Bel G.:"An Object-Oriented Approach for CIM Systems Specification and Simulation". En *Computer Applications in Production and Engineering*, K. Bo, L. Estensen, P. Falster and E.A. Warman, eds., Elsevier Science Publishers, IFIP, 1987.

[HUBE 84] P. Huber, A.M. Jensen, L.O. Jepsen:"Towards reachability trees for high level Petri nets", *Advances in Petri Nets 1984, Lecture Notes in Computer Science 188, pp. 215-233*, Springer-Verlag, Berlin. 1984.

[HUBE 89] Huber P., Jensen K. and Shapiro M.: "Hierarchies in Coloured Petri Nets". *Proc. of the 10th International Conference on Application and Theory of Petri Nets, pp. 192-209*, Bonn, Germany, June. 1989.

[IEEE 87] Invited Sessions on Petri Nets and Flexible Manufacturing. *Proc. of 1987 IEEE International Conference on Robotics and Automation, Vol 2, pp. 999-1018 y 1160-1186*, Raleigh, North Carolina, March-April 1987.

[INAN 89] Inan K.M. and Varaiya P.P.: "Algebras of Discrete Event Models." *IEEE Proceedings (Special Issue on Discrete Event Systems), Vol 77, No 1, pp. 24-38*, Jan. 1989.

[ISEN 88] Isenberg R. and HuebnerM.: "A Workcell Controller Using a Knowledge-Based Simulation Model for Real-Time Production Planning in the Electronics Industry". *Proc. IMACS, Vol. 3, pp. 425-429*, Paris, Jul. 1988.

[JENS 81] Jensen K.: "Colored Petri nets and the invariant method". *Theoretical Computer Science 14, pp. 317-336*, North Holland. Publ. Co., June. 1981.

[JENS 86] Jensen K.:"Colored Petri Nets". En G. Goos, J. Hartmanis (Eds.), *Petri Nets: Central Models and their Properties, pp. 248-299*, Lecture Notes in Computer Science 254, Springer-Verlag, 1986.

[JETE 86] Jeter M.W.: *Mathematical Programming. An Introduction to Optimization*, Marcel Dekker, Inc., New York. 1986.

[KAMA 86] Kamath M. and Viswanadham N.: "Applications of Petri Net Based Models in the Modelling and Analysis of Flexible Manufacturing Systems". *Proc. of the IEEE Conference on Robotics and Automation, pp. 312-317*, San Francisco, CA. 1986.

[KCRA 86] *Knowledgecraft Reference Manual.* Carnegie Group Inc., Pittsburgh, PA. 1986.

[KOCH 87] Kochhar S. and Morris R.J.T.: "Heuristic Methods for Flexible Flow Line Scheduling". *Journal of Manufacturing Systems, Vol. 6, No. 4, pp. 299-314.* 1987.

[KOWA 74] Kowalski R.: "Predicate logic as a programming language". *IFIP Information Processing, pp. 569-574.* 1974.

[KROG 87] Krogh B.H. and Sreenivas R.S.: "Essentially Decision Free Petri Nets for Real-Time Resource Allocation". En *[IEEE 87], pp. 1005-1011.* 1987.

[KUSI 88] Kusiak A. (editor): *Artificial Intelligence Implications for CIM.* IFS (Publications) Ltd/Springer-Verlag, Bedford, UK. 1988.

[LAUT 85] K. Lautenbach: "On Logical and Lineal Dependencies." *GMD Report #147*, Sankt Augustin, Germany. 1985.

[LAW 86] Law A.M.: "Simulation series: Part I: Introducing simulation: A tool for analyzing complex systems". *Industrial Engineering, pp. 46-63*, May. 1986.

[LAWR 86] Lawrence S. and Morton T.: "PATRIARCH: Hierarchical Production Scheduling". *Symposium on Real Time Optimization in Automated Manufacturing Facilities*, Nationale Bureau of Standards, Gaithersburg, MD, January. 1986.

[LEPA 85] LePape C.M.: "A Daily Workshop Scheduling System:. *Expert-Systems 85 - Fifth Technical Conference of the British Computer Society Specialist Group on Expert Systems, pp. 195-212*, Cambridge University Press, University of Warwick, England, December, 1985.

[LEPA 87] LePape C. and Smith S.F.:"Management of Temporal Constraints for Factory Planificación". En C. Rolland, M. Leonard, y F. Bodart (eds.), *Proc. IFIP TC 8/WG 8.1 Working Conf. on Temporal Aspects in Information Systems (TAIS 87)*, Elsevier Science Publishers, pp. 165-176. Mayo, 1987.

[LOON 88] Looney C.G.:"Fuzzy Petri Nets for Rule-Based Decisionmaking". *IEEE Transactions on Systems, Man, and Cybernetics, vol. 18, no. 1, pp. 178-183*. January/February. 1988.

[LOOPS 83] Bobrow D.G. and Stefik M.: "The LOOPS Manual". Xerox Corporation. 1983.

[MARS 85] M.A. Marsan, G. Balbo and K. Trivedi (Eds.):*IEEE/ACM Procs. International Workshop on Timed Petri Nets*, Torino (Italy), July, 1985.

[MART 84] Martínez J.: "Contribución al análisis y modelado de sistemas concurrentes mediante redes de petri." *Tesis Doctoral, ETSIIZ*, Universidad de Zaragoza. Oct. 1984.

[MART 85] Martínez J. and Silva M.:"A Language for the Description of Concurrent Systems Modeled by Colored Petri Nets: Application to the Control of Flexible Manufacturing Systems". *Languages for Automation, pp. 369-388*, Plenum Publishing Co., New York 1985.

[MART 86] Martínez J., Alla H. and Silva M.: "Petri nets for the specification of FMSs." *Modeling and Design of Flexible Manufacturing Systems (A. Kusiak, ed.), pp. 389-406*. Elsevier Science Pub. 1986.

[MART 87] Martínez J., Muro P.R. and Silva M.:"Modeling, Validation and Software Implementation of Production Systems using High Level Petri Nets". En *[IEEE 87] pp. 1180-1185*. 1987.

[MART 89] Martínez J., Muro P.R., Silva M., Smith S.F. and Villarroel J.L.: "Merging Artificial Intelligence Techniques and Petri Nets for Real Time Scheduling and Control of Production Systems". En *Artificial Intelligence in Scientific Computation: Towards second generation systems, pp. 307-313.* Huber R. et al. (editores), J.C. Baltzer AG, Scientific Publishing Co. 1989.

[McDE 82] McDermott D.: "A temporal logic for reasoning about processes and plans." *Cognitive Science, vol. 6, pp. 101-105.* 1982.

[McDE 82] McDermott J.: "R1: a rule-based configurer of computer systems". *Artificial Intelligence, Vol. 19, pp. 39-88.* 1982.

[MINS 75] Minsky M.:"A framework for representing knowledge." En Winston P. (Ed.), *The psycology of computer vision, pp.211-277.* McGraw-Hill, New Jork. 1975.

[MUR 86a] Murata T., Komoda N. and Matsumoto K.:"A Petri Net Based Controller for Flexible and Maintainable Sequence Control and its Applications in Factory Automation". *IEEE Transactions on Industrial Electronics*, pp. 1-8, vol. IE-33, num. 1. February 1986.

[MUR 86b] Murata T. and Zhang D.: "A High-Level Petri Net Model for Parallel Interpretation of Logic Programs." *Procs. of the 1986 International Conf. on Computer Languages, IEEE Comp. So.*, Oct. 1986.

[MURA 89] Murata T.: "Petri Nets: Properties, Analysis and Applications". *Proceedings of the IEEE, Vol. 77, No. 4, pp. 541-580*, April. 1989.

[MURO 87] Muro P.R. y Martínez J.: "Panorámica de soluciones al Problema de Secuenciamiento o Scheduling en Job-Shops". *Informe Técnico ETSIIZ-GISI-ITI-87-3*, Departamento de Ingeniería Eléctrica e Informática, Universidad de Zaragoza. 1987.

[MURO 89] P. Muro y J.L. Villarroel: "KRON: Redes Orientadas a la Representación del Conocimiento". *Actas de la III Reunión Técnica de la AEIA 1989, pp. 3-22.* Madrid, 15-17 Nov. 1989.

[NARA 85] Narahari Y. and Viswanadham N.: "A Petri net approach to the modelling and analysis of flexible manufacturing systems." *Annals of Operations Research, Vol 3, pp. 449-472.* 1985.

[NEWE 72] Newell A. and Simon H.A.: en *Human Problem Solving*, Englewood Cliffs, Prentice-Hall. 1972.

[NEWM 88] Newman P.A.: "Scheduling in CIM Systems". En *Artificial Intelligence Implications for CIM, pp. 361-402*, Kusiak A. (Ed.), IFS (Publications) Ltd/Springer-Verlag, Bedford, UK. 1988.

[NII 86a] Nii H.P.: "Blackboard Systems". *AI Magazine Vol. 7, Num. 2, pp. 38-53*. 1986.

[NII 86b] Nii H.P.: "Blackboard Systems". *AI Magazine Vol. 7, Num. 3, pp. 82-106*. 1986.

[NILS 82] N.J. Nilsson:"*Principles of Artificial Intelligence*". Springer Verlag. 1982.

[ORCI 84] Orciuch E. and Frost J.: "ISA: Intelligent Scheduling Assitant". *First Conference on Artificial Intelligence Applications, pp. 314-320*, IEEE Computer Society, Denver, CO, Dec. 1984.

[OW 84] Ow P.S.: "Heuristic Knowledge and Search for Scheduling". *PhD Thesis, Graduate School of Industrial Administration*, Carnegie Mellon University, Pittsburgh, PA. 1984.

[OW 88a] Ow P.S. and Smith S.F.: "Viewing Scheduling as an Opportunistic Problem Solving Process". En Jeroslow R.G. (editor), *Annals of Operations Research: Approaches to Intelligent Decision Support*, Batzer Scientific Publishing Co. 1988

[OW 88b] Si Ow P., Smith S.F. and Thiriez A.:"Reactive Plan Revision". En *Proc. AAAI-88*. Agosto, 1988.

[PAGN 88] Pagnoni A.: "A Petri Net-Based System for Flexible Manufacturing Management". *9th European Workshop on Application and Theory of Petri Nets, Vol. II, pp. 270-278*, Venice, Italy, June. 1988.

[PANW 77] Panwlaker S.S. and Iskander W.: "A survey of scheduling rules". *Operations Research, 25(1), pp.45-61*. 1977.

[PETE 81] Peterson J.L.:"*Petri Net Theory and the Modelling of Systems*". Prentice-Hall Inc., Englewood Cliffs, NJ.1981.

[PETE 86] G. Peterka and T. Murata: "Proof procedure and answer extraction in Petri net model of logic programs." *Technical Report #86-15, Department of Electrical Engineering and Computer Science*, University of Illinois at Chicago. Sept. 1986.

[PHOE 88] The PHOENIX Group.:"The OPIS Factory Modeling Primitives". *Intelligent Systems Laboratory Working Paper*, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA. 1988.

[PNPM 87] *International Workshop on Petri Nets and Performance Models*, Madison, Winsconsin. IEEE Computer Society Press. 1987.

[QUIL 66] Quillian M.R.: "Semantic Memory". *Report AFCRL-66-189*. Bolk Beranek & Newman, Cambridge, Massachussetts. 1966.

[RAMA 89] Ramadge P.J.G. and Wonham W.M.: "The control of discrete event systems." *IEEE Proceedings. (Special Issue on Discrete Event Systems), Vol 77, No 1, pp. 81-98*, Jan. 1989.

[REDD 86] Reddy Y.V.R., Fox M.S., Husain N. and McRoberts M.: "The Knowledge-Based Simulation System". *IEEE Software, pp. 26-37*, March. 1986.

[RIBA 88] Ribaric S.:"Knowledge Representation Scheme Based on Petri Net Theory". *Int. Journal of Pattern Recognition and Artificial Intelligence*, vol. 2, no. 4, pp. 691-702. December. 1988.

[RIEG 77] Rieger C. and Grinberg M.:"The Declarative Representation and Procedural Simulation of Causality in Physical Mechanisms". In *Proc. 5th Int. Joint Conf. Artif. Intell.*, pp. 250-255. 1977.

[RODA 88] Rodammer F.A. and WhiteK.P.: "A Recent Survey of Production Scheduling". *IEEE Transactions on Systems, Man, and Cybernetics, Vol. 18, No. 6, pp. 841-851*, Nov. 1988.

[SACE 74] Sacerdoti E.D.: "Planning in a hierarchy of abstraction spaces". *Artificial Intelligence, Vol. 5, pp. 115-135*. 1974.

[SATH 85] Sathi A., Fox M.S. and Greenberg M.: "Representation of Activity Knowledge for Project Management." *IEEE Trans. on Patt. Anal. and Mach. Intell.*, vol. PAMI-7, no. 5, September. 1985.

[SATH 86] Sathi N., Fox M.S., Baskaran V. and Bouer J.: "An Artificial Intelligence Approach to the Simulation Life Cycle". En *Proc. of INEX '86, GDI Expert Systems Conference*, Regensdorf, Switzerland. Oct. 1986.

[SHAN 83] Shannon R.E. and Phillips D.T.: "Comparison of Modelling Languages for Simulation of Automated Manufacturing Systems". *Proc. Autofact 5 Conference*, Society of Manufacturing Engineers. 1983.

[SIBE 85] C. Sibertin-Blanc: "High-level Petri Nets with Data Structures" *6th European Workshop on Applications and Theory of Petri Nets*, Espoo, Finland. 1985.

[SIL 85a] M. Silva, J. Martínez, H. Alla, P. Ladet:"Generalized inverses and the calculation of symbolic invariants for colored Petri nets", *Technique et Science Informatique, Vol. 4, No. 1, pp. 113-126*. 1985.

[SIL 85b] M. Silva: "Las Redes de Petri: en la Automática y la Informática". Editorial AC, Madrid. 1985.

[SIL 85c] M. Silva and J. Martínez: "A software environment for designing with colored Petri nets and their implementation. Application to flexible manufacturing systems." *Technical Report*, Universidad de Zaragoza. 1985.

[SIL 86a] M. Silva, T. Murata:"B-fairness and Structural B-fairness in Petri nets models of concurrent systems", *Internal Report*, Dpto. Automática, Universidad de Zaragoza (Spain). June. 1986.

[SIL 86b] M. Silva: "Towards a synchronic theory for P/T nets". En *Concurrency and Nets, pp. 435-460*, K.Voss, H.J.Genrich, G. Rozenberg (eds), Spinger-Verlag. 1986.

[SILV 89] Silva M. and Valette R.: "Petri Nets and Flexible Manufacturing." *Technical Report, Dep. Ingeniería Eléctrica e Informática*, Universidad de Zaragoza. 1989.

[SIMO 83] Simon H.A.: "Search and Reasoning in Problem Solving". *Artificial Intelligence, Vol. 21, pp. 7-29.* 1983.

[SMIT 83] Smith S.F.: "Exploiting temporal knowledge to organize constraints." Intell. Syst. Lab., Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-RI-TR-83-12, 1983.

[SMITa 86] Smith S.F., Fox M.S. and Ow P.S.: "Constructing and Maintaining Detailed Production Plans: Investigations into the Development of Knowledge-Based Factory Scheduling Systems". *AI Magazine, pp. 45-60*, Fall. 1986.

[SMITb 86] Smith S.F., Ow P.S., LePape C., McLaren B. and Muscettola N.: "Integrating Multiple Scheduling Perspectives to Generate Detailed Production Plans", *ULTRATECH-AI in Manufacturing Conference*, Society of Manufacturing Engineers, Long Beach, CA, September. 1986

[SMIT 87] Smith S.F.: "A Constraint-Based Framework for Reactive Management of Factory Planes de operaciones". *Proceedings: International Conference on Expert Systems and the Leading Edge in Production Planning and Control*, Charleston (South Carolina, U.S.A.), May. 1989.

[SMIT 89] Smith S.F.:"A Framework for Modeling Manufacturing Systems". *Draft Paper, Phoenix project*, Intelligent Systems Laboratory, Robotics Institute, CMU. 1989.

[STAN 83] Standridge C.R.: "Performing Simulation Projects with the Extended Simulation System (TESS)". *Simulation, Vol. 45, No. 6, pp. 283-291.* 1985.

[STEa 81] Stefik M.: "Planning with Constraints (MOLGEN: Part 1)". *Artificial Intelligence, Vol. 16, pp. 111-140*. 1981.

[STEb 81] Stefik M.: "Planning and Meta-Planning (MOLGEN: Part 2)". *Artificial Intelligence, Vol. 16, pp. 141-170*. 1981.

[STEF 85] Stefik M. and Bobrow D.G.:"Object-Oriented Programming: Themes and Variations." *The AI Magazine*, pp. 40-62. 1985.

[STEF 86] Steffen M.S.: "A Survey of Artificial Intelligence-Based Scheduling Systems". *Procc. Fall Industrial Engineering Conference*, Boston, MA, Dec. 7-10. 1986.

[TAMU 89] Tamura H., Yamagata K. and Hatono I.: "Decision Making for Flexible Manufacturing - OR and/or AI Approaches in Scheduling -". *Syst. Anal. Model. Simul. Vol. 6, No. 6, pp. 363-371*. 1989.

[TPN 85] *International Workshop on Timed Petri Nets*, Torino, Italy. IEEE Computer Society Press. 1985.

[VALE 82] Valette R., Courvoisier M. and Mayeux D.: "Control of Flexible Production Systems and Petri Nets". En *Applications and Theory of Petri Nets, Informatik-Fachberichte No. 66, pp. 264-277*. 1982.

[VALE 87] Valette R., Atabakhche H.:"Petri Nets for Sequence Constraint Propagation in Knowledge Based Approaches". In K. Voos, H.J. Genrich, G. Rozenberg (Eds.), *Concurrency and Nets*, Advances in Petri Nets, Springer-Verlag, pp. 555-569. 1987.

[VALE 89] Valette R., Cardoso J. and Dubois D.: "Monitoring manufacturing systems by means of Petri nets with imprecise markings". *Proc. IEEE International Symposium on Intelligent Control 1989, pp. 233-238*, Albany, NY, Sept. 1989.

[VALE 90] Valette R. and Bako B.: "Software Implementation of Petri Nets and Compilation of Rule-based Systems". *LAAS Report*, LAAS-CNRS, 7 Av. du Colonel Roche. 31077 Toulouse Cedex, France. Jan. 1990.

[VALE 82] Valette R. et al.:"Control of flexible production systems and Petri nets". *Informatik Fachberichte 66*, Springer-Verlag, pp. 264-267. 1982.

[VEPS 84] Vepsalainen A.P.J.: "State Dependent Priority Rules for Scheduling". *PhD Thesis, CMU-RI-TR-84-19, Graduate School of Industrial Administration and The Robotics Institute*, Carnegie-Mellon University, Pittsburgh, PA. 1984.

[VILL 88] Villarroel J.L., Martínez J. and Silva M.: "GRAMAN: A Graphic System for Manufacturing System Design". *Proc. IMACS International Symposium on System Modelling and Simulation (SMS'88)* , Cetraro, Italy. Sept. 1988.

[VILL 90] Villarroel J.L.:"Integración Informática del Control de Sistemas de Fabricación Flexible". Tesis Doctoral, Departamento de Ingeniería Eléctrica e Informática, Universidad de Zaragoza. En preparación.

[WAH 89] Wha B.W., Lowrie M.B. and Li G.-J.: "Computers for Symbolic Procesing". *Proceedings of the IEEE, Vol. 77, No. 4, pp. 509-540.* April, 1989.

[WATE 78] Waterman D.A. and Hayes-Roth F. (Eds.): *Pattern-directed inference systems.* Academic Press, New York. 1978.

[WINS 77] Winston P.H.: *Artificial Intelligence.* Addison-Wesley, Reading Mass. 1977.

[ZHOU 88] M.C. Zhou, F. DiCesare and A.A. Desrochers: "A Formal Methodology for Synthesis of Petri Net Models and Controllers for Manufacturing Systems." *Technical Report, Department of Electrical, Computer and Systems Engineering*, Rensselaer Polytechnic Institute, Troy, NY. Nov. 1988.

[ZISM 78] Zisman M.D.: "Use of production systems for modelling asynchronous concurrent processes". Ref. [WATE 78], pp.53-68. 1978.

Anexo A

Generación del Mundo de Bloques

Este apéndice muestra el código utilizado para construir el modelo del mundo de bloques, así como los objetos que son generados. El código presentado corresponde a una versión de KRON desarrollada en el lenguaje *LOOPS* corriendo sobre una máquina XEROX.

A.1 Funciones para la Generación del Modelo del Mundo de Bloques

```
(CL:DEFUN  DEFINST  (NAME SUPER)
     (LET* ((OBJ (_ ($! SUPER) |New|)))
          (_ OBJ |SetName| NAME)))


(CL:DEFUN  GENERATE-EJEMPLO  NIL
     (DEF-ARC-COND 'OBJ '(|Schema-name| <VOBJ>))
     (DEF-ARC-COND 'OBJ1 '(|Schema-name| <VOBJ1>))
     (DEF-ARC-COND 'OBJ2 '(|Schema-name| <VOBJ2>))
     (DEF-ARC-COND 'OBJ1-SOBRE-OBJ2 '(|Schema-name| SOBRE <VOBJ2>))
     (DEF-ARC-COND 'QUITA '(|Schema-name| SOBRE <VQUITA>))
     (DEF-ARC-COND 'PONE '(|Schema-name| SOBRE <VPONE>))

          (* |;;| "creo instancia objetos conflicto")
     (DEFINST 'CONFLICT-COGE 'CONFLICT)
     (DEFINST 'CONFLICT-DEJA 'CONFLICT)

          (* |;;| "")
```

```
(DEF-AO-CONFLICT
     '((MUNDO-BLOQUES1 COGE)
     (MUNDO-BLOQUES1 DESAPILA)) 'CONFLICT-COGE)
(DEF-AO-CONFLICT
     '((MUNDO-BLOQUES1 DEJA)
     (MUNDO-BLOQUES1 APILA)) 'CONFLICT-DEJA)

(GENERATE-MUNDO-BLOQUES)
(GENERATE-ROBOT)

(BIDIRECTIONAL-SYNCHRONIZATION
     '(ROBOT1 COGE)
     '((MUNDO-BLOQUES1 COGE)
     (MUNDO-BLOQUES1  DESAPILA)))
(BIDIRECTIONAL-SYNCHRONIZATION
     '(ROBOT1 DEJA)
     '((MUNDO-BLOQUES1 DEJA)
     (MUNDO-BLOQUES1 APILA)))

(COMPILE-BIDIRECTIONAL-SYNCHRONIZATION '(ROBOT1 COGE))
(COMPILE-BIDIRECTIONAL-SYNCHRONIZATION '(ROBOT1 DEJA))

(|PutVal| 'B1 SOBRE 'B2)
(|PutVals| 'MUNDO-BLOQUES1 'LIBRE '(B1 B3))
(|PutVals| 'MUNDO-BLOQUES1 'SOBRE-MESA '(B2 B3))
(|PutVal| 'MUNDO-BLOQUES1 'SOBRE '(B1))
(|PutVal| 'ROBOT1 'MANO-VACIA 'NEUTRO))

(CL:DEFUN  GENERATE-MUNDO-BLOQUES  NIL
     (DEFMOCLASS 'BLOQUE 'MARKING-OBJ
          :PROP  '((VOLUMEN NIL)
            (PESO NIL)
            (COLOR NIL)))
     (DEFCLASSRELATION 'SOBREREL 'REL-ATR
          :DOMAIN     'BLOQUE
          :RANGE     'BLOQUE
          :INVERSE     'DEBAJO
          :RESTRICTION     'RESFUN)
     (DEFRELINST 'SOBREREL 'SOBRE)
     (DEFMOINST 'BLOQUE 'B1)
     (DEFMOINST 'BLOQUE 'B2)
     (DEFMOINST 'BLOQUE 'B3)
     (DEFAOCLASS 'ACTIONS-MUNDO-BLOQUES 'ACTION-OBJ)
     (DEFAOCLASS 'MB-COGE 'ACTIONS-MUNDO-BLOQUES
```

```
          :PRE      '((LIBRE OBJ) (SOBRE-MESA OBJ))
          :POST      '((COGIDO OBJ))
          :DATA      '(<OBJ>) :PREDICATE 'T)
     (DEFAOCLASS 'MB-DEJA 'ACTIONS-MUNDO-BLOQUES
          :PRE      '((COGIDO OBJ))
          :POST      '((LIBRE OBJ) (SOBRE-MESA OBJ))
          :DATA      '(<OBJ>) :PREDICATE 'T)
     (DEFAOCLASS 'MB-APILA 'ACTIONS-MUNDO-BLOQUES
          :PRE      '((COGIDO OBJ1) (LIBRE OBJ2))
          :POST '((SOBRE OBJ1-SOBRE-OBJ2))
          :DATA      '(<OBJ1> <OBJ2>) :PREDICATE 'T)
     (DEFAOCLASS 'MB-DESAPILA 'ACTIONS-MUNDO-BLOQUES
          :PRE      '((SOBRE OBJ1-SOBRE-OBJ2) (LIBRE OBJ1))
          :POST      '((LIBRE OBJ2) (COGIDO OBJ1))
          :DATA      '(<OBJ1> <OBJ2>) :PREDICATE 'T)
     (DEFSOCLASS 'MUNDO-BLOQUES 'STATE-OBJ
          :ACT
               '((COGE MB-COGE)
               (DEJA MB-DEJA)
               (APILA MB-APILA)
               (DESAPILA  MB-DESAPILA))
          :STAT
               '((LIBRE BLOQUE)
               (SOBREMESA BLOQUE)
               (COGIDO BLOQUE)
               (SOBRE BLOQUE)))
     (DEFSOINST 'MUNDO-BLOQUES 'MUNDO-BLOQUES1))

(CL:DEFUN  GENERATE-ROBOT NIL
     (DEFAOCLASS 'ACTIONS-ROBOT 'ACTION-OBJ)
     (DEFMOCLASS 'NEUTRO 'MARKING-OBJ)
     (DEFMOINST 'NEUTRO 'N)
     (DEFAOCLASS 'R-COGE 'ACTIONS-ROBOT
          :PRE      '((MANO-VACIA QUITA))
          :POST      '((MANO-OCUPADA PONE))
          :DATA      '(<QUITA> <PONE>)
          :PREDICATE      T)
     (DEFAOCLASS 'R-DEJA 'ACTIONS-ROBOT
          :PRE      '((MANO-VACIA QUITA))
          :POST      '((MANO-OCUPADA PONE))
          :DATA      '(<QUITA> <PONE>)
          :PREDICATE      T)
     (DEFSOCLASS 'ROBOT 'STATE-OBJ
          :ACT      '((COGE R-COGE) (DEJA R-DEJA))
```

```
        :STAT      '((MANO-VACIA BLOQUE) (MANO-OCUPADA BLOQUE)))
    (DEFSOINST 'ROBOT 'ROBOT1))
```

A.2 Objetos Prototipos del Mundo de Bloques

```
[DEFCLASS  ACTIONS-MUNDO-BLOQUES
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| ACTION-OBJ)]

[DEFCLASS  ACTIONS-ROBOT
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| ACTION-OBJ)]

[DEFCLASS BLOQUE
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:18 by ") )
   (|Supers| MARKING-OBJ)
   (|InstanceVariables|
     (VOLUMEN (NIL)
     (PESO (NIL))
     (COLOR (NIL))
     (INSTANCES (B3 B2 B1))
     (SOBRE #(STATE NIL DPUT)
          TYPE      (SOBRE))
     (DEBAJO NIL
          TYPE      (SOBRE)))]

[DEFCLASS  MB-APILA
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| ACTIONS-MUNDO-BLOQUES)
   (|InstanceVariables|
     (ASSOC-DATA (<OBJ1> <OBJ2>))
     (PREDICATE (T))
     (PRE-NET-RELATIONS
          ((COGIDO OBJ1)
          (LIBRE OBJ2)))
     (POST-NET-RELATIONS ((SOBRE OBJ1-SOBRE-OBJ2)))
     (ASSOC-PROT-SO (MUNDO-BLOQUES)))]

[DEFCLASS  MB-COGE
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| ACTIONS-MUNDO-BLOQUES)
```

```
   (|InstanceVariables| (ASSOC-DATA (<OBJ>))
     (PREDICATE (T))
     (PRE-NET-RELATIONS
          ((LIBRE OBJ)
          (SOBRE-MESA OBJ)))
     (POST-NET-RELATIONS
          ((COGIDO OBJ)))
     (ASSOC-PROT-SO (MUNDO-BLOQUES)))]

[DEFCLASS  MB-DEJA
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| ACTIONS-MUNDO-BLOQUES)
   (|InstanceVariables|
     (ASSOC-DATA (<OBJ>))
     (PREDICATE (T))
     (PRE-NET-RELATIONS
          ((COGIDO OBJ)))
     (POST-NET-RELATIONS
          ((LIBRE OBJ)
          (SOBRE-MESA OBJ)))
     (ASSOC-PROT-SO (MUNDO-BLOQUES)))]

[DEFCLASS  MB-DESAPILA
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| ACTIONS-MUNDO-BLOQUES)
   (|InstanceVariables|
     (ASSOC-DATA (<OBJ1> <OBJ2>))
     (PREDICATE (T))
     (PRE-NET-RELATIONS
          ((SOBRE OBJ1-SOBRE-OBJ2)
          (LIBRE OBJ1)))
     (POST-NET-RELATIONS
          ((LIBRE OBJ2)
          (COGIDO OBJ1)))
     (ASSOC-PROT-SO (MUNDO-BLOQUES)))]

[DEFCLASS  MUNDO-BLOQUES
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| STATE-OBJ)
   (|InstanceVariables| [
     (ACTIONS (DESAPILA APILA DEJA COGE))
     (COGE MB-COGE)
     (DEJA MB-DEJA)
```

```
      (APILA MB-APILA)
      (DESAPILA MB-DESAPILA)
      (STATES (SOBRE COGIDO SOBREMESA LIBRE))
      (LIBRE NIL)
           TYPE      (STATE-SLOT)
           CONSTRAINT      (BLOQUE))
      (SOBREMESA NIL)
           TYPE      (STATE-SLOT)
           CONSTRAINT      (BLOQUE))
      (COGIDO NIL)
           TYPE      (STATE-SLOT)
           CONSTRAINT      (BLOQUE))
      (SOBRE NIL)
           TYPE      (STATE-SLOT)
           CONSTRAINT      (BLOQUE)))]

[DEFCLASS  NEUTRO
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| MARKING-OBJ)
   (|InstanceVariables|
     (INSTANCES (N)))]

[DEFCLASS  R-COGE
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| ACTIONS-ROBOT)
   (|InstanceVariables|
     (ASSOC-DATA (<QUITA> <PONE>))
     (PREDICATE (T))
     (PRE-NET-RELATIONS
          ((MANO-VACIA QUITA)))
     (POST-NET-RELATIONS
          ((MANO-OCUPADA PONE)))
     (ASSOC-PROT-SO (ROBOT)))]

[DEFCLASS R-DEJA
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| ACTIONS-ROBOT)
   (|InstanceVariables|
     (ASSOC-DATA (<QUITA> <PONE>))
     (PREDICATE (T))
     (PRE-NET-RELATIONS
          ((MANO-VACIA QUITA)))
     (POST-NET-RELATIONS
```

```
          ((MANO-OCUPADA PONE)))
     (ASSOC-PROT-SO (ROBOT)))]

[DEFCLASS  ROBOT
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:19 by ") )
   (|Supers| STATE-OBJ)
   (|InstanceVariables|
     (ACTIONS (DEJA COGE))
     (COGE R-COGE)
     (DEJA R-DEJA)
     (STATES (MANO-OCUPADA MANO-VACIA))
     (MANO-VACIA NIL)
          TYPE     (STATE-SLOT)
          CONSTRAINT     (BLOQUE))
     (MANO-OCUPADA NIL)
          TYPE     (STATE-SLOT)
          CONSTRAINT     (BLOQUE)))]

[DEFCLASS  SOBREREL
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 14-Mar-90 10:18 by ") )
   (|Supers| REL-ATR)
   (|InstanceVariables|
     (DOMAIN (BLOQUE))
     (RANGE (BLOQUE))
     (INVERSE DEBAJO)
     (RESTRICTION RESFUN)
     (INSTALLED SOBRE))]
```

A.3 Instancias de Objetos del Mundo de Bloques

```
[DEFCLASS  NEUTRO
   (|MetaClass| |Class| |Edited:|  (* \; "Edited 13-Mar-90 22:01 by ") )
   (|Supers| MARKING-OBJ)
   (|InstanceVariables|
     (INSTANCES (N)))]

[DEFINST ACTION-OBJ (#:ACT-OBJ-FUSION256 "JA 0.bB[:.Z.155")
     (PRE-NET-RELATIONS
          ((ROBOT1 MANO-VACIA QUITA)
          (MUNDO-BLOQUES1 LIBRE OBJ)
```

```
          (MUNDO-BLOQUES1 SOBRE-MESA OBJ)))
     (POST-NET-RELATIONS ((ROBOT1 MANO-OCUPADA PONE)
          (MUNDO-BLOQUES1 COGIDO OBJ)))
     (ASSOC-DATA (<QUITA> <PONE> <OBJ>))
     (PREDICATE (T))
     (ASSOC-STATE-OBJ ((ROBOT1 COGE)
          (MUNDO-BLOQUES1 COGE)))]

[DEFINST ACTION-OBJ (#:ACT-OBJ-FUSION257 "JA 0.bB[:.Z.156")
     (PRE-NET-RELATIONS
          ((ROBOT1 MANO-VACIA QUITA)
          (MUNDO-BLOQUES1 SOBRE OBJ1-SOBRE-OBJ2)
          (MUNDO-BLOQUES1 LIBRE OBJ1)))
     (POST-NET-RELATIONS
          ((ROBOT1 MANO-OCUPADA PONE)
          (MUNDO-BLOQUES1 LIBRE OBJ2)
          (MUNDO-BLOQUES1 COGIDO OBJ1)))
     (ASSOC-DATA (<QUITA> <PONE> <OBJ1> <OBJ2>))
     (PREDICATE (T))
     (ASSOC-STATE-OBJ
          ((ROBOT1 COGE)
          (MUNDO-BLOQUES1 DESAPILA)))]

[DEFINST ACTION-OBJ   (#:ACT-OBJ-FUSION258 "JA 0.bB[:.Z.157")
     (PRE-NET-RELATIONS
          ((ROBOT1 MANO-VACIA QUITA)
          (MUNDO-BLOQUES1 COGIDO OBJ)))
     (POST-NET-RELATIONS
          ((ROBOT1 MANO-OCUPADA PONE)
          (MUNDO-BLOQUES1 LIBRE OBJ)
          (MUNDO-BLOQUES1 SOBRE-MESA OBJ)))
     (ASSOC-DATA (<QUITA> <PONE> <OBJ>))
     (PREDICATE (T))
     (ASSOC-STATE-OBJ ((ROBOT1 DEJA)
          (MUNDO-BLOQUES1 DEJA)))]

[DEFINST ACTION-OBJ   (#:ACT-OBJ-FUSION259 "JA 0.bB[:.Z.158")
     (PRE-NET-RELATIONS
          ((ROBOT1 MANO-VACIA QUITA)
          (MUNDO-BLOQUES1 COGIDO OBJ1)
          (MUNDO-BLOQUES1 LIBRE OBJ2)))
     (POST-NET-RELATIONS
          ((ROBOT1 MANO-OCUPADA PONE)
```

```
          (MUNDO-BLOQUES1 SOBRE OBJ1-SOBRE-OBJ2)))
     (ASSOC-DATA (<QUITA> <PONE> <OBJ1> <OBJ2>))
     (PREDICATE (T))
     (ASSOC-STATE-OBJ ((ROBOT1 DEJA)
          (MUNDO-BLOQUES1 APILA)))]

[DEFINST MB-APILA    (#:MB-APILA251 "JA 0.bB[:.Z.143")
     (PRE-NET-RELATIONS
          ((MUNDO-BLOQUES1 COGIDO OBJ1)
          (MUNDO-BLOQUES1 LIBRE OBJ2)))
     (POST-NET-RELATIONS
          ((MUNDO-BLOQUES1 SOBRE OBJ1-SOBRE-OBJ2)))
         (ASSOC-STATE-OBJ ((MUNDO-BLOQUES1 APILA)))]

[DEFINST MB-COGE   (#:MB-COGE253 "JA 0.bB[:.Z.145")
     (PRE-NET-RELATIONS
          ((MUNDO-BLOQUES1 LIBRE OBJ)
          (MUNDO-BLOQUES1 SOBRE-MESA OBJ)))
     (POST-NET-RELATIONS
          ((MUNDO-BLOQUES1 COGIDO OBJ)))
     (ASSOC-STATE-OBJ ((MUNDO-BLOQUES1 COGE)))]

[DEFINST MB-DEJA   (#:MB-DEJA252 "JA 0.bB[:.Z.144")
     (PRE-NET-RELATIONS
          ((MUNDO-BLOQUES1 COGIDO OBJ)))
     (POST-NET-RELATIONS ((MUNDO-BLOQUES1 LIBRE OBJ)
          (MUNDO-BLOQUES1 SOBRE-MESA OBJ)))
     (ASSOC-STATE-OBJ ((MUNDO-BLOQUES1 DEJA)))]

[DEFINST MB-DESAPILA   (#:MB-DESAPILA250 "JA 0.bB[:.Z.142")
     (PRE-NET-RELATIONS
          ((MUNDO-BLOQUES1 SOBRE OBJ1-SOBRE-OBJ2)
     (POST-NET-RELATIONS
          ((MUNDO-BLOQUES1 LIBRE OBJ2)
          (MUNDO-BLOQUES1 COGIDO OBJ1)))
     (ASSOC-STATE-OBJ ((MUNDO-BLOQUES1 DESAPILA)))]

[DEFINST  MUNDO-BLOQUES  (MUNDO-BLOQUES1 "JA 0.bB[:.Z.141")
     (COGE (#:ACT-OBJ-FUSION256))
     (DEJA (#:ACT-OBJ-FUSION258))
     (APILA (#:ACT-OBJ-FUSION259))
     (DESAPILA (#:ACT-OBJ-FUSION257))
```

```
     (LIBRE (B3 B1))
     (SOBREMESA (B3 B2))
     (SOBRE ((B1)))]

[DEFINST  NEUTRO  (N "JA 0.bB[:.Z.148")
     ]

[DEFINST  R-COGE  (#:R-COGE255 "JA 0.bB[:.Z.154")
     (PRE-NET-RELATIONS ((ROBOT1 MANO-VACIA QUITA)))
          (POST-NET-RELATIONS ((ROBOT1 MANO-OCUPADA PONE)))
     (ASSOC-STATE-OBJ ((ROBOT1 COGE)))]

[DEFINST R-DEJA (#:R-DEJA254 "JA 0.bB[:.Z.153")
     (PRE-NET-RELATIONS
          ((ROBOT1 MANO-VACIA QUITA)))
     (POST-NET-RELATIONS
          ((ROBOT1 MANO-OCUPADA PONE)))
     (ASSOC-STATE-OBJ ((ROBOT1 DEJA)))]

[DEFINST ROBOT (ROBOT1 "JA 0.bB[:.Z.152")
     (COGE (#:ACT-OBJ-FUSION257 #:ACT-OBJ-FUSION256))
     (DEJA (#:ACT-OBJ-FUSION259 #:ACT-OBJ-FUSION258))
     (MANO-VACIA (NEUTRO))]
```

Anexo B

Listado de Objetos del Sistema de Fabricación

Listado de objetos utilizados para modelar la celda de fabricación introducida en el apartado 1.3.2. El sistema consta de seis estaciones de trabajo y un sistema de transporte formado por seis tablas. A continuación se muestran los subsiguientes objetos según el orden utilizado para su creación.

B.1 Objetos de marcado

{*estacion*
is-a : objeto-marcado }

{*tabla*
is-a : objeto-marcado }

{*pieza*
is-a : objeto-marcado }

{*sentido-piezas*
is-a : objeto-marcado }

; relaciones en los objetos de marcado
{*con-tabla*
is-a : especificacion
domain : (TYPE is-a estacion)
range : (TYPE is-a tabla)
inverse : con-estacion
description : "conexion con tabla" }

{*sig-tabla*
is-a : especificacion
domain : (TYPE is-a tabla)
range : (TYPE is-a tabla)
inverse : ant-tabla
description : "siguiente tabla" }

{*localizacion*
is-a : atribucion
domain : (TYPE is-a pieza)
range : (OR (TYPE is-a tabla) (TYPE is-a estacion))
inverse : pieza
description : "localizacion donde se encuentra una pieza" }

{*estacion-asignada*
is-a : atribucion
domain : (TYPE is-a pieza)
range : (TYPE is-a estacion)
inverse : piezas-asignadas
description : "estacion a la que es asignada una pieza" }

{*sentido*
is-a : atribucion
domain : (TYPE is-a pieza)
range : (TYPE is-a sentido-piezas)
inverse : sentido-de
description : "siguiente tabla" }

; una vez hechas las instancias:
{*estacion*
is-a : objeto-marcado
inverse+inv : S_1 S_2 S_3 S_4 S_5 S_6
con-tabla :
piezas-asignadas :
pieza : }
{S_1
instance : estacion
con-tabla : T_1 }
...
{S_6
instance : estacion
con-tabla : T_6 }

{*tabla*
is-a : objeto-marcado
inverse+inv : T_1 T_2 T_3 T_4 T_5 T_6
sig-tabla :
con-estacion :
pieza : }

{T_1
instance : tabla
sig-tabla : T_2
con-estacion : S_1 }
{T_2
instance : tabla
sig-tabla : T_3
ant-tabla : T_1
con-estacion : S_2 }
...
{T_6
instance : tabla
ant-tabla : T_5
con-estacion : S_6 }

{*sentido-piezas*
is-a : objeto-marcado
inverse+inv : *ent* *sal*
sentido-de : }

{*pieza*
is-a : objeto-marcado
inverse+inv : X Y Z R S T D V W
sentido :
estacion-asignada :
localizacion : }
...

{X
instance : pieza
sentido : *ent* }

B.2 Objetos de estado

{*transp-tablas-R*
instance : transporte-tablas
estados : tablas tablaslibres
tablas :
type : slot-estado
dato-asociado : (TYPE is-a pieza)
tablaslibres : T_1 T_2 T_3 T_4 T_5 T_6
dato-asociado : (TYPE is-a tabla)
acciones : entrada salida sigtabla carga-est descarga-est
nivel-precision : nivel-precision-celda
descripcion : }

{*celda-estaciones-R*
instance : celda-estaciones
estados : C CC P CP D CD
C :
dato-asociado : (TYPE is-a estacion)
CC : S_1 S_2 S_3 S_4 S_5 S_6
dato-asociado : (TYPE is-a estacion)
P :
CP : S_1 S_2 S_3 S_4 S_5 S_6
D :
CD : S_1 S_2 S_3 S_4 S_5 S_6
acciones : carga-st carga-pu descarga-pu descarga-st
nivel-precision : nivel-precision-celda
descripcion : }

B.3 Objetos de acción

{*carga-st*
instance : carga-estacion
pre-relaciones-red :
(transp-tablas-R tablas fPETS)
(celda-estaciones-R CC fest)
post-relaciones-red :
(transp-tablas-R tablaslibres ftabla)
(celda-estaciones-R C fPE)
dato-asociado : < *vtabla*, *vestacion*, *vpieza*, *vsentido* >
predicado : (eq < *vsentido* > 'ent)

descripcion : "Carga de una pieza en una estacion"}

{*carga-pu*
instance : carga-puesto
pre-relaciones-red :
(celda-estaciones-R C fPE)
(celda-estaciones-R CP fest)
post-relaciones-red :
(celda-estaciones-R CC fest)
(celda-estaciones-R P fPE)
dato-asociado : < *vpieza*, *vestacion* >
predicado : t
descripcion : "Carga de una pieza en un puesto"}

{*descarga-pu*
instance : descarga-puesto
pre-relaciones-red :
(celda-estaciones-R P fPE)
(celda-estaciones-R CD fest)
post-relaciones-red :
(celda-estaciones-R CP fest)
(celda-estaciones-R D fPE)
dato-asociado : < *vpieza*, *vestacion* >
predicado : t
descripcion : "Descarga de una pieza en un puesto"}

{*descarga-st*
instance : descarga-estacion
pre-relaciones-red :
(celda-estaciones-R D fPE)
(transp-tablas-R tablaslibres ftabla)
post-relaciones-red :
(celda-estaciones-R CD fest)
(transp-tablas-R tablas fPETS)
dato-asociado : < *vpieza*, *vestacion*, *vtabla*, *vsentido* >
predicado : (eq ¡*vsentido* > '*sal*)
descripcion : "Descarga de una pieza de una estacion"}

{*sig-tabla*
instance : siguiente-tabla
pre-relaciones-red :
(transp-tablas-R tablas fPETS)
(transp-tablas-R tablaslibres ftabla)
post-relaciones-red :

(transp-tablas-R tablas fPESS)
(transp-tablas-R tablaslibres fsig-tabla)
dato-asociado : < *vtabla*, *vsigtabla*, *vpieza*, *vestacion*, *vsentido* >
predicado :
(AND
(NOT (null (getvalue < *vtabla* > 'sig-tabla))
(OR
(AND
(posicion-tabla-estacion < *vtabla* > *vestacion*)
(eq (getvalue < *vpieza* > 'sentido) '*ent*))
(AND
(NOT (posicion-tabla-estacion < *vtabla* > *vestacion*)
(eq (getvalue < *vpieza* > 'sentido) '*sal*))))
descripcion : "Descarga de una pieza de una estacion"}

{*entrada*
instance : entrada-transporte
pre-relaciones-red :
(transp-tablas-R tablaslibres ftabla)
post-relaciones-red :
(transp-tablas-R tablas fPETS)
dato-asociado : < *vtabla*, *vpieza*, *vestacion* >
predicado :
(AND
(eq < *vtabla* > 'T_1)
(eq < *vsentido* > '*ent*))
descripcion : "Entrada de una pieza en la celda de fabricacion"}

{*salida*
instance : dalida-transporte
pre-relaciones-red :
(transp-tablas-R tablaslibres fPETS)
post-relaciones-red :
(transp-tablas-R tablaslibres ftabla)
dato-asociado : < *vtabla*, *vpieza*, *vestacion*, *vsentido* >
predicado :
(AND
(eq < *vtabla* > 'T_6)
(eq < *vsentido* > '*sal*))
descripcion : "Salida de una pieza de la celda de fabricacion"}

B.3.1 Condiciones de arco

Las **condiciones de arco** se definen de la siguiente forma:

fPETS ≡
{**schema-name** < *vpieza* >
estacion-asignada < *vestacion* >
localizacion < *vtabla* >
sentido < *vsentido* > }

fPESS ≡
{**schema-name** < *vpieza* >
sentido < *vsentido* >
estacion-asignada < *vestacion* >
localizacion < *vsigtabla* > }

fest ≡
{**schema-name** < *vestacion* > }

ftabla ≡
{**schema-name** < *vtabla* > }

fsig-tabla ≡
{**schema-name** < *vtabla* >
sig-tabla < *vsigtabla* > }

fPE ≡
{**schema-name** < *vpieza* >
localizacion < *vestacion* > }