D. RAMÓN I. BARBER CASTAÑO
con D. N. I. : 5.404.247

A U T O R I Z A:

Que su tesis doctoral con el título "Desarrollo de una arquitectura para robots móviles autónomos. Aplicación a un sistema de navegación topológica" pueda ser utilizada para fines de investigación por parte de la Universidad Carlos III de Madrid.

Leganés, 14 de julio de 2000

Fdo.: Ramón I. Barber Castaño


**UNIVERSIDAD CARLOS III DE MADRID**

**Departamento de Ingeniería Eléctrica, Electrónica y Automática**

# TESIS DOCTORAL

# DESARROLLO DE UNA ARQUITECTURA PARA ROBOTS MÓVILES AUTÓNOMOS. APLICACIÓN A UN SISTEMA DE NAVEGACIÓN TOPOLÓGICA.

Autor

**Ramón I. Barber Castaño**
Ingeniero Industrial


Directores

**Miguel Ángel Salichs Sánchez-Caballero**
Doctor Ingeniero Industrial

**Luis Moreno Lorente**
Doctor Ingeniero Industrial

Leganés, 2000

# TESIS DOCTORAL

## DESARROLLO DE UNA ARQUITECTURA PARA ROBOTS MÓVILES AUTÓNOMOS. APLICACIÓN A UN SISTEMA DE NAVEGACIÓN TOPOLÓGICA.


Autor: **Ramón I. Barber Castaño**

Directores: **Dr. Miguel Ángel Salichs Sánchez-Caballero**
**Dr. Luis Moreno Lorente**


Tribunal Calificador:

Presidente: CARLOS BALAGUER BERNALDO DE QUIRÓS

Vocales: RICARDO GARCÍA ROSA
RAMÓN GALÁN LÓPEZ
JAVIER GONZÁLEZ JIMÉNEZ

Vocal Secretario: CARLOS CERRADA SOMOLINOS

Calificación: SOBRESALIENTE "CUM LAUDE" (POR UNANIMIDAD)

Leganés, 14 de JULIO de 2000

*A mis padres y hermanos.*

*A toda mi familia y a todos mis amigos.*

# Agradecimientos

Tras unos largos años de trabajo, ha llegado la hora de recoger los frutos, y es justo ser agradecido con todas las personas que han contribuido de una forma u otra en la elaboración de esta tesis.

A Miguel Ángel y Luis, mis directores de tesis por su confianza y apoyo mostrado durante estos años. Gracias por haber inculcado en mí el interés por la investigación. Esta tesis es consecuencia de nuestras largas discusiones.

A Carlos Balaguer por darme la oportunidad de comenzar mi tarea docente en la Universidad Carlos III.

A Ramón Galán y los profesores de DISAM, que han contribuido en mi formación como ingeniero y me introdujeron en el mundo de la investigación.

A mi grupo de Colaboradores: Verónica, Pilar, Pablo, Juan Carlos, Sergio y José Manuel, que han colaborado activamente en los trabajos de esta tesis.

A Vicente y José María, que han hecho las veces de hermano mayor en el mundo de la robótica. Especialmente quiero agradecer a Vicente el que siempre que lo he necesitado se ha levantado de su silla para ayudarme en ese mismo momento.

Al resto de compañeros: Samuel, quien me ha sacado de más de un apuro con los ordenadores, Mª Jesús por su apreciable ayuda con los robots, Paco Pepe, Arturo, José Manuel, Antonio, Santiago, Dolores, Beatriz, Mohamed, Victor, Mario, Cristina, Ramiro, Eladio, Luis, Javier, Ángela, Eva, Khamis, Salah y Carlos, que han propiciado un entorno agradable de trabajo y que han estado disponibles en todo momento.

También quiero mostrar mi agradecimiento a Marcelo, Juan Pablo, José Manuel y Amparo, mis amigos del "cole", que han seguido con interés toda la elaboración de esta tesis.

Por último, agradecer a toda mi familia su apoyo: especialmente a mis padres, hermanos y a mi tía Mª Teresa, por el calor familiar que me han proporcionado, y sin el cual, no habría sido posible la elaboración de esta tesis.

# Resumen

El objetivo de esta tesis es el desarrollo de una arquitectura para el control de robots móviles autónomos y su aplicación al caso de navegación topológica. Para la elaboración de la arquitectura se ha tenido en cuenta como son los procesos mentales en los humanos. La arquitectura desarrollada es una arquitectura que consta de dos niveles: Uno deliberativo y otro automático. Por ello la arquitectura ha sido denominada AD (Automatic-Deliverative Architecture).

En el nivel deliberativo se encuentran aquellas habilidades que requieren tiempo de cómputo elevado como consecuencia de razonamientos a alto nivel. Estas habilidades son gestionadas por un secuenciador existente en este nivel, que activa y desactiva las distintas habilidades deliberativas.

En el nivel automático se encuentran las habilidades que interactuan con los sensores y actuadores del robot. Estas habilidades son gestionadas por habilidades del nivel deliberativo y se encargan del movimiento del robot y de activar los sistemas auxiliares necesarios para la percepción del entorno. La respuesta de las habilidades automáticas son más rápidas que la de las habilidades deliberativas.

Dicha arquitectura ha sido aplicada a un novedoso sistema de navegación topológica denominado EDN (Event Driven Navigation). Este sistema se basa en la información contenida en un tipo original de mapa topológico: la carta de navegación. En dicha carta de navegación la información se guarda en forma de nodos y arcos. Los nodos se corresponden con eventos sensoriales y los arcos con acciones.

Finalmente, se ha demostrado experimentalmente la viabilidad de la nueva arquitectura y del sistema de navegación implementado sobre ella.

## Abstract

The objective of this thesis is the development of an architecture for the control of mobile robots and its aplication to topological navigation. To create this architecture we took into account how mental processes are performed in humans. The developed architecture has two levels: one is deliberative and the other is automatic. For this reason, we have named this architecture AD. (Automatic-Deliberative Architecture)

The deliberative level includes those abilities that require a high computing time because of the high level reasoning. A sequencer located on the same level manages those abilities. The sequencer activates and desactivates the different deliberative abilities.

The automatic level includes the abilities that interact with the robot's sensors and actuators. These abilities are managed by the abilities of the deliberative level and are in charge of the movement of the robot and of the auxiliary systems that perceive the environment. The response of the automatic abilities is faster than that of the deliberative abilities.

This architecture has been applied in a new navigation system called EDN (Event Driven Navigation). This system is based on the information included in an original topological map: the navigation chart. In this navigation chart the information is kept in node and edge shapes. The nodes correspond to the sensorial events and the edges correspond to actions.

Finally, we demonstrate the feasibility of the new architecture as well as the navigation system implemented in it.

# Índice

# Índice de tablas

# Índice de figuras

# 1

# INTRODUCCIÓN

Siempre copiamos de la naturaleza, obteniendo así la civilización, deberíamos aprender de ella para conseguir nuestra naturaleza.
FRAN. J. MARTIN

El día que el hombre se dé cuenta de sus profundas equivocaciones, habrá terminado el progreso de la ciencia.
MADAME CURIE

¿Cuánto tiempo me llevará resolver mi problema?. Ni un minuto más que lo que tardes en comprenderlo, dijo el maestro.
ANTHONY DE MELLO

# Introducción 1

## 1.1 Motivación

Esta tesis está enmarcada dentro de las tesis dedicadas a las arquitecturas de control dentro de la robótica móvil.

El objetivo de la misma es el de aportar una arquitectura que sea capaz de conseguir que un robot autónomo se mueva de forma coherente para conseguir desempeñar una determinada tarea.

En esta tesis, como ocurre siempre que se trata de realizar una arquitectura sobre robots móviles, aparece el dilema de la planificación frente a la reactividad. En la

arquitectura presentada en esta tesis conviven la reactividad con la deliberación en una arquitectura en dos niveles.

Como todas las arquitecturas de robots basadas en la inteligencia artificial, la arquitectura considerada busca emular el comportamiento humano. Cuando nosotros nos movemos, buscamos hacerlo de forma natural. Primero pensamos dónde tenemos que ir y cómo tenemos que hacerlo. Normalmente pensamos en el lugar al que nos dirigimos y en los sitios por los que tenemos que pasar. A su vez, pensamos cómo tenemos que ir. Por ejemplo, para ir de un despacho a otro dentro de un edificio, primero debemos conocer donde se está, y a continuación establecer una lista de los sitios por donde tenemos que pasar (Pasillos, descansillos, etc.). Esta información es únicamente topológica. También nos hacemos una idea de cuán lejos está, siendo esta información más o menos difusa. De ello deriva que el planificador de trayectorias deba ser fundamentalmente topológico, aunque sin olvidarse del componente geométrico.

A su vez, los humanos somos capaces de evitar obstáculos, de modificar el camino previsto si surge algún problema, y de replanificar el camino si nos hemos perdido totalmente. Ello es posible fundamentalmente a la capacidad de razonar, a la precisión de nuestro sistema sensorial y a nuestra capacidad de relacionar los sitios por los que vamos pasando con los lugares planeados. Esto es lo que se pretende emular con la presente arquitectura. Dentro de un plan, el robot debe de conservar su capacidad reactiva frente a obstáculos, pero una parte de la arquitectura debe de ser capaz de darse cuenta si el robot evoluciona de forma adecuada o no. Deberá existir, por lo tanto, algún mecanismo para monitorizar la evolución del robot y detectar si el robot sigue el plan previsto. De no ser así, habrá que tomar las medidas oportunas, actualizando su posición y volviendo a planificar de nuevo si es necesario.

En la tesis se consideran, de acuerdo con el modelo humano, dos niveles en la arquitectura: uno deliberativo y otra automático. El nivel deliberativo de la arquitectura se corresponde con el nivel deliberativo de las personas. Dicho nivel está relacionado con la capacidad de razonar y de tomar decisiones que tienen las personas. El nivel automático de la arquitectura está relacionado con el nivel automático humano. Este nivel es el encargado

de realizar las acciones que se ejecutan de forma automática, como por ejemplo andar. Emulando estos dos niveles, en el caso del robot tendremos dos niveles: uno donde se realizan las actividades que necesitan tiempo para procesar la información y otro donde se llevan acabo los movimientos básicos del robot.

## 1.2 Objetivos

Los objetivos de la presente tesis están orientados a conseguir que el robot se mueva de forma eficiente para ser capaz de realizar una tarea concreta. Los objetivos más concretos que motivan esta tesis son:

- Construcción de una arquitectura completa para el control de vehículos autónomos inteligentes.

- Construcción de un sistema de navegación topológica. El robot debe ser capaz de moverse en un entorno dado en forma de eventos sensoriales y acciones motoras.

La arquitectura debe incorporar los mecanismos necesarios para trabajar con información tanto topológica como geométrica, siendo capaz de integrar la planificación con los comportamientos reactivos más básicos.

## 1.3 Contribuciones

La principal aportación de esta tesis consiste en la definición de una arquitectura basada en habilidades y que es capaz de trabajar con información topológica y geométrica.

En la tesis se presenta un mapa topológico totalmente novedoso. El mapa está constituido por una combinación de eventos sensoriales a captar y acciones motoras a realizar. Este mapa topológico puede considerarse como una carta de navegación, en la que se indican los lugares por donde debe pasar el robot y cómo ha de llegar hasta ellos.

En la tesis se utiliza como ejecutor del plan un Diagrama Funcional. En él las etapas están asociadas a las acciones motoras a realizar y las transiciones a los eventos sensoriales a captar.

A su vez, en esta tesis se definen los conceptos de habilidades deliberativas y automáticas, así como se describen algunas de ellas.

## *1.4 Estructura de la tesis*

El contenido de la tesis se encuentra distribuido en varios capítulos, los cuales se describirán a continuación.

En el capítulo 2 de esta tesis se hace una descripción de las principales arquitecturas de control de robots desarrolladas hasta el momento, haciendo hincapié en las arquitecturas por capas actuales.

En el capítulo 3 se realiza la descripción de la arquitectura desarrollada en la presente tesis. Se presentará la filosofía que motiva la arquitectura y se hará la descripción general de la arquitectura y de sus dos niveles.

En el capítulo 4 se describe el nivel superior de la arquitectura o nivel deliberativo. Se explicarán sus componentes, destacando las habilidades deliberativas y los secuenciadores.

En el capítulo 5 se describe el nivel inferior de la arquitectura o nivel automático. Se explicarán sus componentes, destacando las habilidades automáticas y las acciones reflejas.

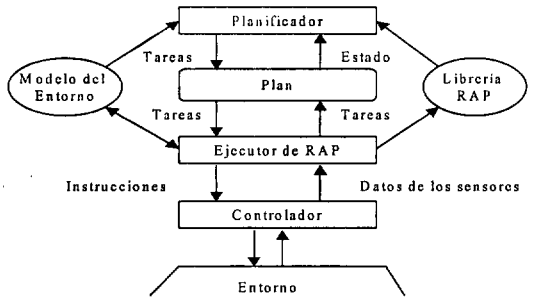
En el capítulo 6, se muestran unos ejemplos de cómo serían vistas otras arquitecturas bajo el enfoque de la nueva arquitectura.

En el capítulo 7, se hace una introducción a la navegación topológica, haciéndose un estudio sobre las tendencias actuales que utilizan información topológica para navegar.

En el capítulo 8, se presenta la carta de navegación como modelado del entorno. En dicha carta de navegación la información se guarda en un grafo topológico. Se comenta como puede tenerse en cuenta la información geométrica para navegar. Por último, se comenta la arquitectura propuesta para el caso de navegación topológica.

En el capítulo 9 se muestran los resultados experimentales obtenidos tras implementar la arquitectura presentada en la tesis en un robot real.

Por último, en el capítulo 10 se establecen las conclusiones y se apuntan las siguientes líneas de investigación.




# 2

# ARQUITECTURAS DE CONTROL DE ROBOTS

El hombre con una idea nueva es un mentecato hasta que la idea triunfa.
MARK TWAIN

Con el conocimiento, se acrecientan las dudas.
GOETHE

El primer paso hacia la filosofía es la incredulidad.
DIDEROT

# Arquitecturas de control de robots 2

## 2.1 Introducción

A la hora de considerar una nueva arquitectura de control en el mundo de la robótica, la primera cuestión es ver como se distribuye la capacidad de razonar y la de actuar dentro de la arquitectura. La cuestión que se planteó al inicio de las investigaciones en el campo de la robótica móvil es la de escoger entre planificación o reactividad. Esta es una cuestión ampliamente debatida en el mundo de la robótica móvil. Aunque cada una de las filosofías tiene sus partidarios y sus detractores, es difícil hacer prevalecer a la una sobre la otra. Más bien se diría que la tendencia actual es juntar ambas filosofías en los denominados sistemas híbridos.

Aunque a priori se pueda pensar que los sistemas puramente reactivos sean menos eficientes a la hora de alcanzar un objetivo que los sistemas que contienen un planificador de tareas, es difícil demostrarlo de forma analítica. La demostración depende enteramente de la comparación de ambas filosofías bajo el marco de una tarea determinada, del entorno y de las capacidades computacionales y sensoriales del robot.

En principio, las arquitecturas planificadas pueden parecer más propicias para realizar un estudio analítico de sus capacidades y su respuesta a diversas configuraciones del entorno. Sin embargo, la existencia de condiciones imprecisas e impredecibles en el entorno, hace que estos sistemas sean tan difíciles de validar como aquéllos que se basan en una arquitectura reactiva.

Por otro lado, un sistema de control basado en una arquitectura reactiva, permite construir robots móviles cuya respuesta sea en tiempo real, e incorporar nuevas funcionalidades al sistema. Sin embargo, este tipo de arquitectura presenta una serie de inconvenientes como son:

- Dificultad de alcanzar objetivos globales.
- A medida que se van incorporando nuevos niveles de competencia, la complejidad del sistema va aumentando.
- Se hace difícil a un usuario definir de una manera flexible la serie de tareas que deba realizar el robot.

Si seguimos buscando en la literatura, cada vez aparecen más líneas donde se intente una integración de ambos, como propone esta tesis. El ser humano siempre combina una necesaria planificación para saber como llegar donde se propone con una necesaria reactividad para conseguir evitar obstáculos e imprevistos. Dicho de otra forma, las personas vamos cotejando siempre si estamos haciendo lo que teníamos que hacer de forma correcta. Esto lo hacemos de forma continua y a todos los niveles, es decir, no solo comprobamos si el camino seguido es el correcto, si no que nos aseguramos que todos los pasos los demos correctamente.

La arquitectura propuesta en esta tesis puede ser englobada con las arquitecturas híbridas. Dicha arquitectura tendrá un planificador de tareas flexible que no impedirá la necesaria reactividad de las capas inferiores.

En este capítulo se va a hacer un breve recorrido por algunas de las arquitecturas utilizadas a lo largo de la historia de la robótica, haciendo hincapié en cómo se distribuye la capacidad de razonar y la capacidad de actuar.

## 2.2 Arquitecturas planificadas

La propuesta de la Inteligencia Artificial clásica era la de realizar una planificación a largo plazo basándose en el modelo del mundo (mapa) desarrollado en un instante dado. Si el robot se encontraba ante un evento imprevisto, se modifica el mapa y se vuelve a planificar, generándose un nuevo plan de actuación. Uno de los principales objetivos era el de generar un modelo simbólico del mundo (escenario cartesiano) a partir de la información sensorial, de tal forma que la máquina de inferencia pudiera razonar y planificar a priori los movimientos del robot [Fikes71]. Primero se especificaban el formato de la base simbólica de conocimiento y la máquina de inferencia, y más tarde se desarrollaban los módulos de percepción que deberían proveer los símbolos necesarios.

La mayor parte de las arquitecturas toman como base los tradicionales enfoques de sistemas jerárquicos. Si nos referimos al control de un robot móvil autónomo, este enfoque divide el problema en módulos funcionales como los indicados en la figura 2.1.

Las arquitecturas tradicionales planificadas se caracterizan por dos fundamentos muy determinados:

1. La descomposición funcional de la arquitectura.
2. La generación de un modelo del mundo.

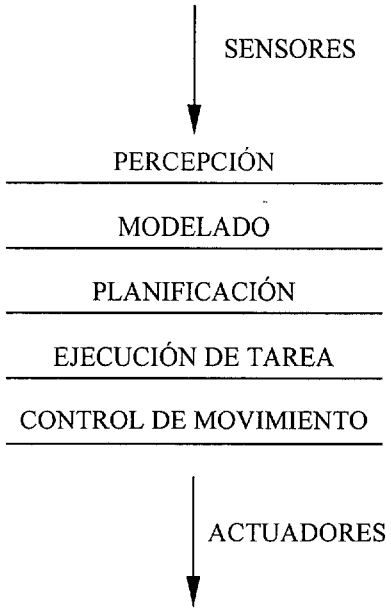



**Figura 2.1** Descomposición funcional en las arquitecturas clásicas de control.

**Descomposición funcional**: El método tradicional de diseño de un sistema de navegación se basa en que las tareas a realizar por el robot móvil se encuentran en un nivel de abstracción más elevado que el conjunto de variables de control del robot La solución que se aporta es la de descomponer la tarea en un refinamiento progresivo de la solución, de forma que se definan una serie de funciones que reduzcan progresivamente el nivel de abstracción del problema para acercarse al nivel de las variables de control del robot. Desde el punto de vista del flujo de la información, esta metodología realiza una descomposición en una serie de unidades funcionales como la percepción, la generación de un modelo de lo percibido, la planificación de una trayectoria sobre este modelo, la ejecución de la tarea, y finalmente el control de los motores del robot móvil (figura 2.1). Este flujo es cíclico, con lo que se repite para cada una de las acciones del robot móvil.

**Modelo del Mundo**: Los sistemas construidos siguiendo las líneas marcadas por la Inteligencia Artificial clásica tratan de modelar el entorno y pueden responder a cuestiones relacionadas con él. Por lo general son sistemas "cerrados", en el sentido de que no hay una interacción directa con el entorno del que se ha pre-codificado un cierto conocimiento. El tiempo de respuesta del sistema no es una de las prioridades, ya que el entorno no va a

cambiar mientras se está calculando la solución. Por otro lado, los defensores de este tipo de filosofía siempre han abogado por un incremento en las prestaciones de los ordenadores y sistemas sensoriales en un futuro próximo. Pero, en definitiva, lo que realmente caracteriza a este tipo de sistemas es la necesidad de generar un modelo simbólico del mundo a partir de la información suministrada por los sensores. Este modelo del mundo está lleno de incertidumbres, sujeto a errores, y su cómputo requiere una gran cantidad de tiempo y recursos. Independientemente de la tarea que se vaya a realizar, el modelo del mundo se construye siempre, a pesar de que gran parte de la información contenida en él no sea necesaria para el objetivo que trata de alcanzar el sistema.

### 2.2.1 Arquitectura NASREM

Como ejemplo representativo de este tipo de arquitecturas se puede tomar la propuesta por Janes Albus [Albus85][Albus87] como parte de su trabajo en el NIST (National Institute of Standards and Technology) en los Estados Unidos. Esta arquitectura llamada NASREM (Nasa Standard Reference Model), ha sido adoptada como sistema de control para manipuladores en las estaciones espaciales. La figura 2.2 ilustra esta arquitectura de control.

Albus propone una arquitectura jerárquica como resultado de la descomposición de tareas necesaria para alcanzar un objetivo. La arquitectura se basa en los conceptos de descomposición espacial, descomposición temporal y ejecución. La jerarquía de niveles se define por:

- Descomposición temporal y espacial de tareas en distintos niveles de detalle
- Integración temporal y espacial de los datos de los sensores en niveles de abstracción
- Resolución espacial de objetos y mapas para la representación del conocimiento del entorno.

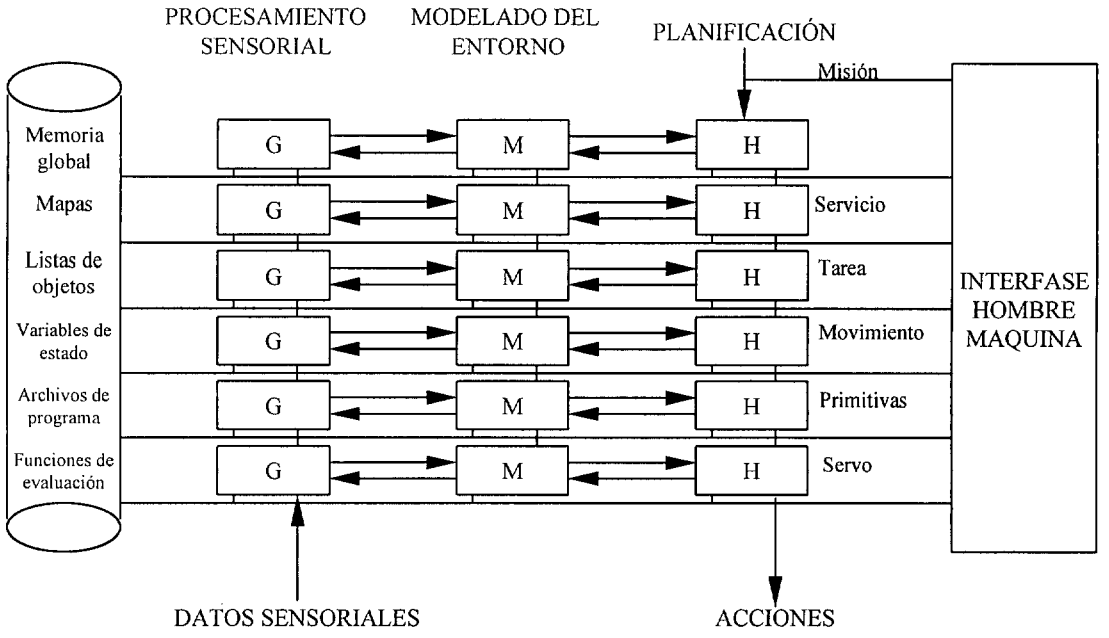



**Figura 2.2** Arquitectura de control NASREM

En la literatura aparecen otros ejemplos de arquitecturas planificadas. Algunos de ellos son:

Saridis crea una arquitectura jerárquica de acuerdo al principio de incremento de inteligencia con decremento de precisión [Saridis83]. La arquitectura está basada en un modelo de tres niveles de organización, coordinación y ejecución que se acomoda a las funciones de control.

La Ford Motor Company desarrolla en 1982 una arquitectura jerárquica, implementándose en un robot móvil.[Kuan86][Kuan87][Kuan88]

Chatila [Chatila92] propone una arquitectura que incorpora algunos esfuerzos en el sentido de añadir robustez y control reactivo, pero no de una manera específica.

Buttazzo [Buttazzo92], propone la implementación de una arquitectura jerárquica (Harems) que se basa en la transferencia de mensajes operando sobre un sistema de tiempo real (Hartik). La arquitectura está muy bien detallada considerando los siguientes niveles:

aplicación, acción, comportamiento, comunicación y dispositivo. Esta propuesta tiene como novedad el uso de un sistema distribuido para su implementación.

Como conclusión general, todas ellas se basan en modelos multicapa y siguen la aproximación sensor-plan-actuador, donde la deliberación lleva el peso más importante dentro de cada una de las arquitecturas mencionadas.

## 2.3 Arquitecturas reactivas

A mediados de los años ochenta, Brooks [Brooks86][Brooks91a], en respuesta a esta visión tradicional de la Inteligencia Artificial aplicada a la robótica móvil, introdujo una nueva filosofía de diseño que él definió como Sistemas Reactivos. Reemplazó la modularización funcional por módulos generadores de comportamientos. Una jerarquía de procesos se hacía cargo del sistema. Cada uno de estos procesos era extremadamente simple y dotado de una funcionalidad completa (evitar obstáculos, bordear una pared, etc.). El corto camino entre los sensores y los actuadores (desde un punto de vista computacional) proporcionaba una respuesta en tiempo real del robot ante cambios en el entorno. La implementación del sistema se realizaba de abajo a arriba, donde primero se trataba de asegurar la supervivencia del robot, mediante comportamientos básicos como evitar obstáculos, y después se le iban asignando nuevas y más complejas aptitudes, como la confección de mapas del entorno [Mataric92a], o la manipulación de objetos [Connell89].

Las arquitecturas reactivas basadas en comportamientos [Mataric92b] se fundamentan en la acción inmediata ante el estímulo percibido. Los objetivos del sistema son múltiples, y las condiciones del entorno dictan cual de ellos es el prioritario.

Los robots diseñados mediante este tipo de técnicas se asemejan en su comportamiento a los insectos [Webb93], y son utilizados como herramienta de estudio en el campo de la Vida Artificial [Levy92], que trata de emular los procesos básicos de la vida en los ordenadores. Asimismo, las arquitecturas multi-agente se han utilizado en otros

campos de la inteligencia artificial y en entornos complejos [Genes94], como la cooperación de un grupo de robots móviles [Mataric94].

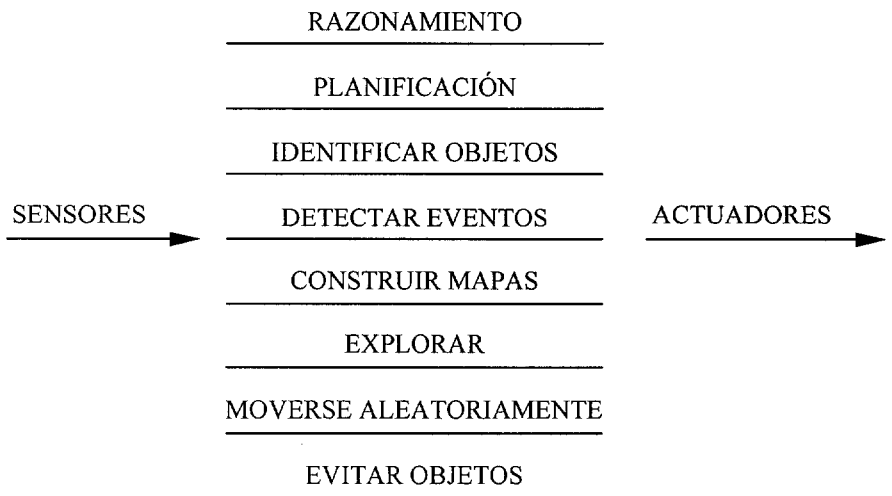



**Figura 2.3** Descomposición de una tarea en comportamientos

Las arquitecturas de control reactivo tienden a reemplazar el clásico ciclo de sensación, planificación, acción por un ciclo de sensación, acciones elementales, combinación de acciones elementales (figura 2.3), manteniendo por supuesto relación con niveles superiores necesarios para llevar a cabo una tarea.

La idea básica en la que se sustentan las arquitecturas reactivas es la descomposición jerárquica del problema en niveles de competencia horizontales. Se construyen niveles del control para cada uno de los niveles de competencia, añadiendo nuevas capas a las ya existentes. Se comienza construyendo un robot con una capa lo más simple posible, cuyo único objetivo sea evitar objetos, tal y como se puede observar en la figura 2.3. Dicha capa se encarga de asegurar la supervivencia del robot en el entorno. Es decir, un nivel que reaccione a la señal sensorial de tal forma que cuando el robot se encuentra muy cercano a un obstáculo, lo evite. Sobre esta capa de control se puede añadir una segunda capa, en la que el objetivo de la misma sea el de vagar por el entorno, sin tener en principio un objetivo claro. De esta forma, el control del robot está estratificado de modo que los niveles inferiores toman el control del robot cuando eventos del entorno así

lo deciden. El sistema se puede cortar en cualquiera de los niveles implementados, y los niveles inferiores formarán un sistema de control operativo completo.

Los sistemas basados en arquitecturas reactivas no necesitan de entornos muy definidos. Por lo general, estos sistemas tienen integrados en su repertorio múltiples competencias de bajo nivel. A su vez, están situados en su entorno, es decir, están directamente conectados a él a través de sus sensores y actuadores. En este tipo de sistemas se pone un especial énfasis en la autonomía del sistema. El sistema ha de observar el entorno y decidir por si mismo cómo se ha de resolver el problema y cómo ha de elegir entre un número elevado de metas simultáneas. Sus estructuras internas son activas[Mataric94] y generan un "comportamiento" determinado, al contrario que las estructuras estáticas que utilizan los sistemas basados en el conocimiento. En particular, el sistema se considera integrado en su entorno y está situado en un espacio determinado. Esto implica que hay una necesidad menor de tener que modelar el entorno porque el mundo es su mejor modelo [Brooks91b] y se puede utilizar como una memoria externa [Connell89].

El Laboratorio de Inteligencia Artificial del Massachusetts Institute of Technology (M.I.T.) fue el pionero en el diseño e implementación de robots móviles que utilizaban una arquitectura reactiva como sistema de control. Estos modelos se caracterizaban por ser capaces de realizar toda una serie de tareas de una forma robusta y fiable en tiempo real.

Dentro del conjunto de este tipo de arquitecturas, la arquitectura Subsumption desarrollada por R. A. Brooks es de gran interés ya que en alguna manera constituye la base del desarrollo del control reactivo.

### 2.3.1 Arquitectura Subsumption

La idea básica de la arquitectura de Brooks [Brooks86] es descomponer el sistema no sobre la base de las operaciones internas de éste (descomposición funcional), si no sobre la base de su comportamiento externo. Trabaja además con comportamientos elementales

que producen acciones de diferente tipo siendo necesaria una coordinación de ellas para producir acciones complejas.

Brooks define la arquitectura de Subsumption o inclusión como una alternativa a la utilización de técnicas de IA clásica para planificación y control en robots [Brooks86]. La arquitectura de inclusión tiene las siguientes propiedades:

- Se define un conjunto de comportamientos que operan asíncronamente y en paralelo sin una función central que los coordine.
- Cada comportamiento incorpora las funciones de percepción, planificación y ejecución de tareas que son necesarias para alcanzar la conducta externa deseada.
- La inhibición o inclusión de las entradas y salidas de un comportamiento se realiza a través de otro comportamiento.

La arquitectura propuesta por Brooks puede utilizarse a través de las siguientes observaciones:

- Los comportamientos del sistema no resultan necesariamente de estructuras internas complejas.
- El diseño del sistema debería ser simple, cuando un sistema se compone de varios elementos las interfases entre ellos no deberían ser complejas o la descomposición es incorrecta
- Los sistemas deberían ser capaces de mantener su autonomía durante el mayor tiempo posible sin la intervención humana.

Esta arquitectura se basa en una idea de evolución donde la ejecución de niveles superiores (comportamientos complejos), implica la ejecución de los de nivel más bajo (comportamientos primitivos).

Si esta arquitectura se aplica al caso de la robótica móvil la descomposición de las tareas del sistema en comportamientos tal y como lo propone Brooks puede observarse en la figura 2.3.

Algunas consecuencias importantes pueden ser extraídas de este enfoque:

- Cada capa puede tratar con objetivos diferentes, por lo que el sistema entero puede tratar con una variedad de los mismos.
- Se pueden utilizar varios tipos de sensores de acuerdo con las necesidades de cada capa sin construir necesariamente una representación central.

El sistema global es robusto, y la depuración es bastante fácil debido a que cada capa puede ejecutarse independientemente antes de que se añadan otras. Este es un punto importante a tener en cuenta ya que evita la depuración y corrección de un único sistema grande y complejo. La arquitectura es fácilmente extensible y puede corresponder a una implementación simple.

De la figura 2.3 se desprende que los niveles de competencia para el control de un robot móvil propuestos por Brooks son:

1. Escape de colisiones.
2. Movimiento aleatorio.
3. Exploración del entorno.
4. Construcción de un mapa del entorno y planificación de trayectorias.
5. Tomar en consideración cambios que ocurren en el entorno.
6. Razonar sobre objetos y realizar tareas.
7. Generación y ejecución de planes que transformen el entorno a un estado deseado.
8. Razonamiento sobre los comportamientos de los objetos y adaptación a los planes.

Un nivel dado se realiza mediante procesos que combinan la acción y la información sensorial a través de módulos simples. Los módulos que Brooks propone son máquinas de estado finito y están fuertemente acopladas, éstos se ejecutan asíncronamente intercambian mensajes sin ningún protocolo.

La base de esta arquitectura de Subsumption es que un módulo puede suprimir o inhibir las entradas (salidas) de otro por un determinado periodo de tiempo reemplazándolas por otras con lo que capas altas de control pueden ser integradas fácilmente con las capas previas a través de este mecanismo.

## 2.4 Arquitecturas híbridas

Las arquitecturas híbridas aparecieron como un compromiso entre las arquitecturas tradicionales y las arquitecturas reactivas. Este tipo de arquitecturas comparten con las arquitecturas tradicionales el concepto fundamental de la necesidad de una planificación en un modelo del mundo generado a partir de la información sensorial. Por otro lado, reconocen la necesidad de que el robot reaccione y sea robusto ante cambios en tiempo real en el entorno. Por tanto, este tipo de arquitecturas se suelen componer de un subsistema reactivo que se encarga de la supervivencia de robot en todo momento, y un sistema de planificación de alto nivel.

Estas arquitecturas intentan integrar los modelos jerárquicos y de comportamiento. Las arquitecturas híbridas son una consecuencia bien de una evolución de las arquitecturas de comportamiento o bien de la definición de un nuevo tipo de arquitectura que incluye características relevantes de ambos métodos.

Atendiendo al intervalo temporal que abarcan los dos tipos de planificación que acabamos de describir, se puede realizar la siguiente clasificación:

- *Planificación inmediata*: En un sistema reactivo las acciones a tomar por el robot están precodificadas: El sistema está compuesto por un conjunto de agentes, cada uno de los cuales tiene una meta diferente, a la que trata de llegar en cada instante. El entorno, las metas de cada agente, junto con el arbitraje del control precodificado, establecen el comportamiento global del sistema.

- *Planificación a largo plazo*: Los sistemas tradicionales realizan una planificación a largo plazo de un conjunto de acciones del robot, basándose en un modelo simbólico del mundo completo y preciso.

Las arquitecturas híbridas han intentado aunar los dos tipos de planificación en un único sistema jerárquico. De tal forma, el subsistema reactivo se encarga de las tareas básicas de supervivencia y de navegación, mientras que un subsistema simbólico de alto nivel planifica las tareas y trayectorias de un futuro a más largo plazo.

Sin embargo, las arquitecturas híbridas presentan dos problemas en su utilización, tanto de índole práctica como conceptual:

1. Desde un punto de vista práctico, en el momento en el que el plan inicial no sea correcto, se debe volver a planificar un nuevo conjunto de acciones. Por tanto, el robot (en principio) se debe parar hasta que se genere el nuevo plan. Es decir, utilizar un control reactivo para que el robot reaccione en tiempo real a eventos imprevistos en el entorno, y un sistema simbólico estático que planifique las tareas, es un contrasentido.

2. Desde un punto de vista conceptual, la transición entre el sistema reactivo y el de planificación es brusca y no natural. Es decir, en este tipo de arquitecturas se ha tratado de dar un salto cualitativo en las prestaciones de las arquitecturas reactivas dotándolas de un componente simbólico tradicional. Sin embargo, no se ha tratado de aumentar las prestaciones de los robots reactivos tratando de seguir la misma filosofía empleada en las primeras "capas" cognitivas del sistema [Brooks91b].

Las arquitecturas híbridas buscan integrar las arquitecturas anteriores en una arquitectura que utilice planificación pero que permita la reactividad en los niveles inferiores. Algunos de estos ejemplos son Aura [Arkin92] y AFREB [Gachet93] [Armingol97]

### 2.4.1 Arquitectura Aura

La principal aportación de esta arquitectura está en que utiliza una navegación basada en campos de potencial, de carácter inminentemente reactivo. Sin embargo, va a ser considerada como una arquitectura híbrida ya que presenta módulos de planificación.

Arkin [Arkin92] incorpora, en cambio, el concepto de múltiples comportamientos concurrentes que él denomina "esquemas" y que producen acciones del mismo tipo que finalmente se combinan utilizando una metodología de campos de potenciales.

Arkin define la arquitectura (AuRA) para un robot autónomo [Arkin89] siguiendo el modelo de navegación animal propuesto por Arbib y House para explicar el comportamiento de las ranas. AuRA tiene las siguientes propiedades:

- Utiliza un régimen de control motriz o esquemas motrices como mecanismos de comportamientos reflexivos o percepción orientada a la acción.
- No se utiliza un conocimiento de alto nivel para el entorno o para la selección de estrategias motrices y de percepción.
- No hay una planificación a priori, en su lugar se activa un esquema del tipo 'mueve hacia objetivo' o 'mueve hacia adelante'.
- No existen capas de esquemas como en el modelo de Brooks.
- Los esquemas son activados dinámicamente mediante mecanismos basados en percepción según el robot avanza a través del mundo externo.
- Cada vector de salida de los esquemas motrices que fueron activados se añade y normaliza para conseguir la velocidad y dirección en las cuales el robot debería moverse.

Esta arquitectura ha sido desarrollada por R.C Arkin [Arkin 92] en la Universidad de Massachusetts. En su desarrollo este investigador introduce el concepto de varios comportamientos elementales que él denomina esquemas (Un término derivado de las ciencias biológicas).

Arkin, ha implementado esta arquitectura para el control de un robot móvil autónomo. En este caso los comportamientos elementales que mencionan son por ejemplo: "evitar objetos estáticos", "moverse hacia adelante", "permanecer en un camino", etc. Todos estos comportamientos están pensados para realizar tareas de tipo navegacional y se implementan como procedimientos parametrizados.

En esta arquitectura se distinguen cuatro módulos:

- **Planificador de Misiones:** El planificador de misiones tiene la responsabilidad de la planificación de alto nivel. Este incluye capacidad de razonamiento espacial, determinación de los parámetros necesarios para el módulo de navegación y de pilotaje, así como modos de operación y criterios de optimización. Los comandos de misión los introduce un operador a través de una interfase de usuario. La salida del módulo de planificación se dirige hacia el módulo de navegación y consiste en modos de operación, que determinan el comportamiento global del robot así como una lista de parámetros, adicionalmente se proveen submetas para la tarea que se va a realizar.

- **Módulo de Navegación:** Este módulo diseña un camino punto a punto desde la posición actual del robot hasta el punto de meta, en base a un mapa del entorno que se tiene almacenado a priori. El nivel de representación usado por este módulo es un modelo de grafo de espacios libres donde los obstáculos se aproximan como polígonos. La trayectoria se calcula en base a la ubicación de los vértices de éstos.

  Constantemente el módulo de navegación reporta al planificador de misión el estado de la misma.. Se representa el entorno inmediato al vehículo, donde se mantiene información acerca de las marcas visibles, características de los obstáculos conocidos, atributos del terreno, etc. Estos datos están disponibles para la predicción del subsistema de percepción o para ser usados por el módulo de pilotaje en la selección y ejecución de comportamientos.

La salida de este módulo se dirige al subsistema piloto, ésta consiste en una trayectoria punto a punto así como algunos parámetros que afectan el comportamiento de este bloque.

Esencialmente el módulo de navegación se basa en un modelo mientras que el piloto se basa en la información sensorial. El sistema piloto reporta constantemente al módulo superior el cumplimiento o no de las submetas establecidas. Si ha existido un fallo del piloto, el módulo de navegación puede iniciar una replanificación sin invocar al planificador de misión.

- **Módulo de Pilotaje:** El módulo de pilotaje acepta una trayectoria punto a punto desde el módulo de navegación y provee al robot de comportamientos de movimiento que le permitan un desplazamiento seguro.

Este módulo selecciona los comportamientos (esquemas) apropiados de un repertorio de ellos y se los envía parametrizados hacia el módulo que se encarga de instanciarlos.

A partir de este punto se ejecuta un ciclo a través del "administrador de esquemas". Durante el movimiento, el módulo cartógrafo construye una representación temporal del entorno a partir de los datos provenientes de los sensores. Si por algún motivo el administrador de comportamientos falla en alcanzar un punto durante el período de tiempo determinado entonces se reinvoca al piloto para encontrar un camino alternativo en base a los mapas a priori y temporal. Polígonos aproximados representando objetos detectados se insertan en el mapa y se corre otra vez el algoritmo de búsqueda de espacio libre.

Los comportamientos de movimiento con los que trabaja Arkin son:

- Moverse en una dirección
- Moverse a un punto.

- Evitar obstáculos estáticos.
- Parada de emergencia..
- Mantenerse en una trayectoria identificable (camino, sendero, etc.).

Se incluyen también comportamientos de percepción como por ejemplo:

- Encontrar un obstáculo.
- Encontrar una referencia.
- Encontrar un camino.

A fin de proveer una acción de movimiento apropiada para recorrer una trayectoria, el módulo piloto accede a la información que se encuentra presente tanto en el mapa a priori como en el mapa temporal, que representa los alrededores del robot y conjuntamente con las submetas provistas por el módulo de navegación las almacena en una base de hechos al inicio de cada etapa.

Arkin utiliza un sistema de reglas para seleccionar los comportamientos en base a las submetas de navegación. El piloto mantiene también una base de reglas que se aplican al contexto actual (base de hechos). El resultado es una lista de comportamientos movimiento y de percepción parametrizados y optimizados para una particular etapa navegacional. Algunos comportamientos se instancian siempre como el de evitar obstáculos estáticos mientras otros se ejecutan de acuerdo al contexto.

- **Administrador de Comportamientos:** El control distribuido para la ejecución de un movimiento de un punto a otro ocurre dentro de los confines del módulo administrador de comportamientos. Múltiples comportamientos están activos durante el movimiento del robot a fin de lograr una correcta transición de la trayectoria. Arkin usa una metodología de campo de potencial para calcular los comandos de velocidad y curvatura que se darán a los actuadores del robot. Se calcula un vector de velocidad total a partir de las contribuciones vectoriales individuales de cada comportamiento.

Cuando el robot se mueve, el sistema cartógrafo, utilizando los datos sensoriales construye un modelo del mundo percibido en la zona de memoria temporal. Si la trayectoria actual del robot se desvía demasiado de la especificada inicialmente por el módulo de navegación debido a la presencia de obstáculos o errores posicionales se vuelve a invocar al módulo de navegación y se calcula una trayectoria global diferente. Si la trayectoria del vehículo está dentro de límites aceptables (lo que se determina en los módulos superiores) el subsistema de pilotaje y el administrador de comportamientos intentan evitar el obstáculo, seguir un camino o afrontar otros problemas.

## *2.5 Arquitecturas en tres capas*

La tendencia actual, como ya se ha comentado antes, es la de incorporar deliberación y reactividad en una misma arquitectura. Dentro de estas arquitecturas, se van a destacar las arquitecturas en tres capas desarrolladas por Firby, Gat y Bonasso, ya que marcan la tendencia actual y pueden considerarse como antecesoras de la arquitectura AD (Automatic-Deliberative Architecture) propuesta en esta tesis.

### 2.5.1 Arquitectura propuesta por Firby

James Firby [Firby92][Firby94] pertenece al laboratorio de Inteligencia Artificial del Departamento de Informática de la Universidad de Chicago, estudia el control de agentes autónomos y específicamente los problemas que supone integrar el razonamiento simbólico y el control continuo. Desarrolló un sistema de ejecución de planes llamado RAP que se basa en "programas-paquete de acciones reactivas" que es lo que denomina RAPs. Los planes consisten en tareas definidas mediante esos RAPs y cada RAP genera acciones básicas en el robot en el momento de ejecución, utilizando el mejor método posible dentro de una variedad, dependiendo del entorno y las circunstancias que rodeen a la tarea a realizar. Considera dichos RAPs ya no solo como programas que corren en el tiempo de ejecución, sino como bloques de construcción jerárquicos para la realización de

planes. El sistema de ejecución de RAP que describe incluye una memoria sensorial, un lenguaje y un intérprete.

La arquitectura completa se muestra en la figura 2.4. En ella se observa como su arquitectura posee tres capas: una *capa planificadora* que produce otros planes según las metas que le lleguen, el *ejecutor de RAP* que profundiza en los detalles del plan anterior en el tiempo de ejecución, y un *sistema de control* que actualmente lleva a cabo acciones en el entorno.

El elemento clave es el que sirve de nexo entre el plan y el controlador: el *ejecutor de RAP*. El controlador necesita una información mucho más precisa de la que el planificador le puede ofrecer ya que esta última es vaga e incompleta. El sistema RAP permite convertir esas vagas instrucciones en instrucciones detalladas, adecuadas a la situación real, sacadas de una librería preexistente. El papel del *ejecutor* es el de asegurar que no existen problemas por haber elegido métodos incorrectos para llevar a cabo metas, chequeando que cada método alcance su meta y si no lo hace, eligiendo otro e intentándolo de nuevo.

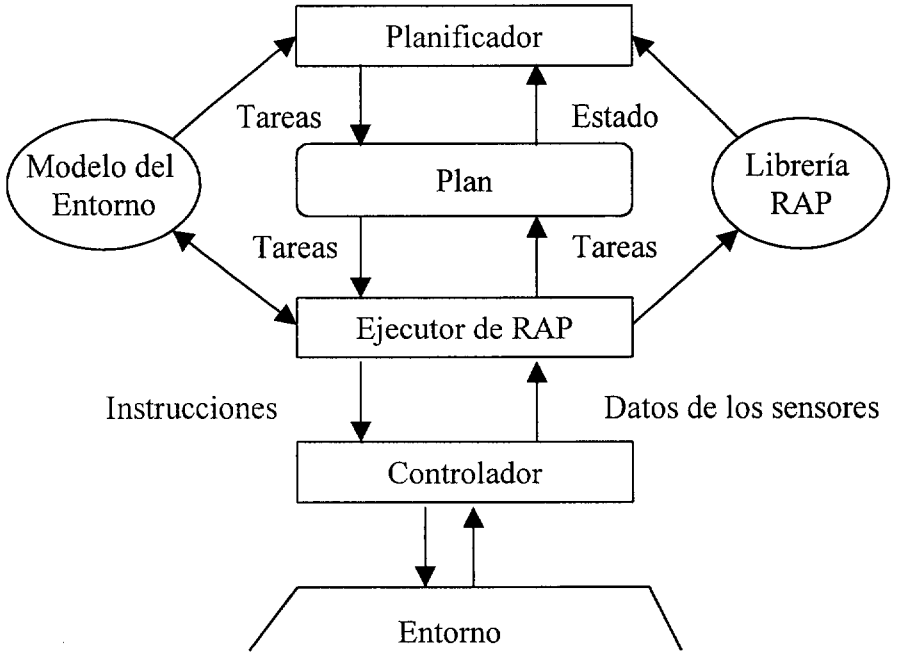



**Figura 2.4** Arquitecura completa propuesta por Firby

El papel del sistema RAP puede ser visto de dos modos fundamentales. Por una parte el sistema RAP es reactivo y puede conseguir metas complejas empleando sus propios métodos y creando una librería que le dará suficiente tiempo para la ejecución. Por otra sirve de nexo para proveer al planificador de acciones concretas que puedan ser entendidas por el controlador.

La meta que busca es la de construir un sistema de control que sea capaz de tener una gran variedad de acciones básicas manteniéndose en dimensiones relativamente pequeñas. Su aproximación es construir un sistema de control basado en comportamientos que pueda ser reprogramado por el sistema de ejecución RAP. Permitiendo al sistema de control que se pueda reconfigurar se permite que los mismos comportamientos puedan ser usados de muchas maneras diferentes sin tener que codificarlas con antelación.

La arquitectura diseñada por Firby para poder acoplar el sistema RAP con el control continuo es la representada en la figura 2.5. Esta parte de la arquitectura tiene tres componentes: El sistema RAP, una colección de rutinas de activación de sensores, y una colección de rutinas de control mediante comportamiento. Las dos últimas permiten el sistema de control en tiempo real. Considera rutinas para los sensores y actuadores que pueden ser activadas, desactivadas y reconfiguradas individualmente.

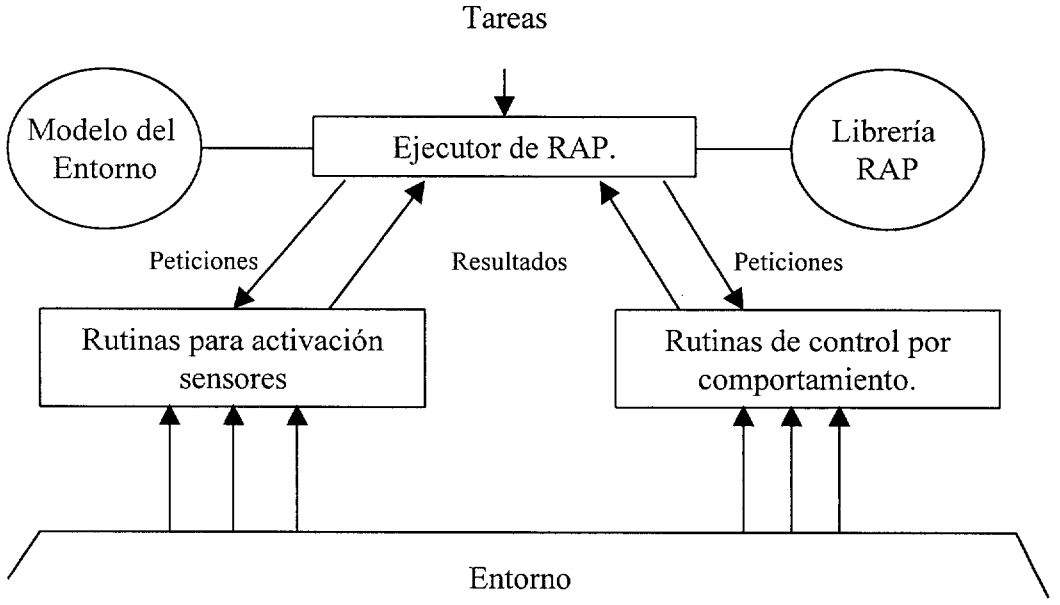



**Figura 2.5** Control continuo de la arquitectura propuesta por Firby

RAP no generará actividades, generará instrucciones para habilitar, deshabilitar y reconfigurar rutinas. RAP convierte en instrucciones de control esas actividades. Una vez que se activa todo el mecanismo, las rutinas se van produciendo de manera continua por lazos cerrados y sin necesidad de intervenir.

Las tareas definidas por RAP tienen la ventaja de ser discretas y finitas. Cada acción debe tener un comienzo y un final bien definidos. La mayor dificultad de las tareas radica en que tienen un comienzo muy bien definido pero su final no está tan claro. Debido a la imprecisión de los sensores en muchas ocasiones no se llegan a alcanzar las metas deseadas. Por ello considera acciones determinadas como fallos. Por ejemplo si el robot se ha perdido se tiene que establecer la acción que considere que la tarea que provocó esa situación ha fallado.

### 2.5.2 Arquitectura 3T

Peter Bonasso [Bonasso97][Schreck98] investiga maneras de combinar el control deliberativo y el reactivo para programar robots que lleven a cabo tareas de manera robusta en distintos entornos. Un elemento no tiene solo que ajustarse a cambios en una situación, debe también ser capaz de sintetizar planes. La complejidad del mundo real hace que intentar tener en cuenta la totalidad de acciones que pueden ocurrir con antelación sea impracticable.

La arquitectura diseñada permite a un robot, por ejemplo, planear una serie de actividades en varias localizaciones, moverse entre esas localizaciones realizando actividades y simultáneamente evitar peligros, mantener niveles determinados y aceptar la guía de un supervisor humano.

Utiliza una arquitectura que permite programar muchos robots móviles en entornos reales y ofrece un sistema unificado para el control de sistemas inteligentes. Esta arquitectura separa el problema de dotar de inteligencia al robot en tres capas que denomina 3T. Esta subdivisión consiste en:

- Un sistema dinámico reprogramable formado por un conjunto de habilidades reactivas coordinadas por un gestor de habilidades.
- Un secuenciador que activa y desactiva estos conjuntos de habilidades para que, en función del entorno, desempeñen tareas específicas. Para ello utilizan el sistema RAP.
- Un planificador deliberativo que estudia en profundidad las metas a alcanzar y las restricciones de tiempo. Para esto usan un sistema conocido como el "Adversarial Planner".

La estructura de esta arquitectura queda reflejada en la figura 2.6.

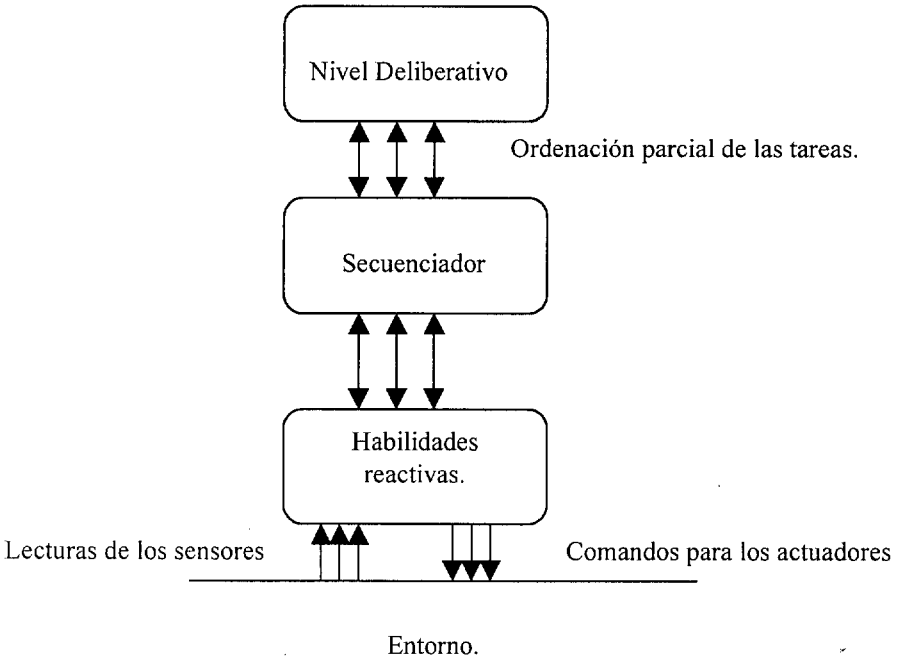



**Figura 2.6** Arquitectura 3T

Las habilidades representan la conexión con el entorno. El sistema de control en la práctica se crea estableciendo un grupo de habilidades que trabajan juntas en un contexto dado. Es necesario dadas las características de las habilidades crear una interfaz con el secuenciador, que sea independiente de las características físicas del robot y de sus sensores. Ello conlleva:

1. Establecer las entradas y salidas de esas habilidades. Cada habilidad debe tener una descripción de las entradas que espera y una lista de las salidas que genera. Esto permite que las tareas aparezcan en una red, de tal forma que las salidas de una habilidad pueden ser tomadas como entradas de otra, de manera inmediata.

2. Una transformación computacional. Este es el punto en que la habilidad hace su trabajo. Cuando se establece una habilidad, utiliza su transformación para computar de nuevo sus salidas basándose en sus nuevas entradas.

3. Una rutina de inicialización. A cada habilidad se le da la posibilidad de iniciarse por si misma una vez que el sistema se pone en marcha.

4. Una función de habilitación. El secuenciador puede habilitar y deshabilitar habilidades. Dependiendo del contexto, a la habilidad se le da la oportunidad de variar comenzando con un procedimiento especial.

5. Deshabilitar una función. Cuando no se necesita una habilidad, el secuenciador la deshabilitará y la función deshabilitada realizará todos los cambios de limpieza necesarios.

Desde la perspectiva del secuenciador, las habilidades pueden ser habilitadas y deshabilitadas dependiendo de la situación. Excepto para entradas y salidas habituales, cada habilidad es totalmente independiente del resto. Para dotar al secuenciador con una interfaz uniforme y permitir a las habilidades comunicarse, el entorno de desarrollo de las habilidades encapsula las habilidades dentro del *manejador de habilidades*. El determinar cual es la mejor manera de configurar las habilidades para la situación que se tiene es tarea del secuenciador.

Para llevar a cabo las tareas que el robot debe realizar de manera rutinaria, la arquitectura tiene un sistema secuenciador. En su caso el secuenciador es el sistema RAP. RAP es simplemente una descripción de cómo llevar a cabo tareas en el entorno bajo una gran variedad de circunstancias usando pasos discretos.

La manera de realizar una tarea depende del conocimiento que el robot tiene de su situación. El secuenciador contiene una librería de tareas RAPs, cada una para una situación diferente y activa un diferente conjunto de habilidades para realizar una tarea. Lo que se utiliza es el comando "wait-for" que implica una parada hasta que ocurra algo que se denomina "evento". Los eventos toman entradas de otras habilidades y notifican al secuenciador cuando un estado ha sido alcanzado. Así, el secuenciador utiliza los eventos para determinar cuando un conjunto de habilidades ha completado su trabajo y cuando unos estados particulares en el entorno han cambiado.

Bonasso señala que todavía la combinación del secuenciador y la reactividad no está estructurada para realizar razonamientos complicados de localización. Mientras que el secuenciador tiene la habilidad de manejar situaciones rutinarias, no puede organizar secuencias nuevas de éstas tareas para manifestar un comportamiento determinado. La habilidad para considerar la implicación global de las acciones es la tarea para la que los planificadores reactivos se diseñan.

Desde su punto de vista, existe un modo de realizar un planificador basado en estados dentro de la inteligencia robótica, pero no tiene que manejar tareas que se han especificado como secuencias de habilidades usuales del robot. Cuando se planifica es necesario que el planificador opere al mayor nivel de abstracción posible de tal forma que haga que el problema de espacio sea el menor posible.

El objetivo es controlar en tiempo real los comportamientos. El papel del secuenciador es generar series de comportamientos en tiempo real bien conocidos. En el proceso, el secuenciador alcanzará niveles de abstracción de las actividades con los que el planificador trabajará. Esto simplificará los problemas en la planificación porque se utilizan pocas operaciones para una gran familia de acciones a ejecutar. La ventaja del sistema RAP es el manejo de comportamientos iterativos, que simplificará mucho la representación del planificador, permitiendo una representación de estados parecida a la de los sistemas de planificación clásicos. Es importante destacar que los tres elementos estudiados, *habilidades, secuenciador y planificador* trabajarán juntos y de manera asíncrona.

El planificador que emplea es el AP [Elsaesser91], que tiene una serie de características que le hacen adecuado para la planificación con robots. AP fue diseñado para utilizar coordinación multiagente extendiendo su planificación basada en estados con los razonamientos que se dan durante las acciones. Esta capacidad permite a AP planificar actividades tales como que dos robots transporten un objeto. AP puede razonar sobre agentes que no controla como resultado de su desarrollo original. Un agente no controlado puede ser una persona que opera en un entorno. AP utiliza un modo *counterplanning* para razonar sobre como las condiciones iniciales. Un plan podrían negarse por un agente no controlado. Estos problemas se solucionan aumentando el plan con operaciones que pueden neutralizar los efectos negativos de una acción incontrolada. AP puede utilizar la capacidad de razonamiento como un mecanismo que considera la probabilidad de interacciones peligrosas con otros agentes.

### 2.5.3 Arquitectura ATLANTIS

Erann Gat[Gat91][Gat92][Gat97] propone una arquitectura heterogénea y asíncrona para el control de robots móviles, que es capaz de controlar la ejecución de múltiples tareas en tiempo real en entornos ruidosos e impredecibles. La arquitectura integra planificación y reactividad usando elementos de arquitectura heterogénea y usando los resultados de la planificación como guía pero no para hacer control directamente.

ATLANTIS combina un control reactivo con un sistema de planificación tradicional de acuerdo con el modelo sensor-planificador-actuador.

La arquitectura que presenta tiene tres componentes:

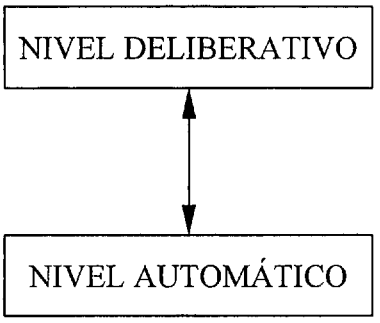
- El controlador: es un mecanismo de control reactivo que controla actividades primitivas (actividades sin computación de toma de decisión). Emplea el lenguaje ALFA, diseñado para tal propósito, como soporte para programar mecanismos de control reactivo.

- El secuenciador: es un sistema operativo de propósito especial que controla la iniciación y la terminación de actividades primitivas. El control de las secuencias es difícil porque el secuenciador debe ser capaz de tratar con fallos inesperados. Ello conlleva un cuidadoso mantenimiento de la información interna del estado ya que el secuenciador debe ser capaz de recordar que acciones han sido realizadas en el pasado en orden a decidir que acción debería hacerse en este momento. El secuenciador inicia y termina actividades primitivas por activación y desactivación de los módulos del controlador. El secuenciador también puede enviar parámetros al controlador. El secuenciador fue modelado en un principio usando el paquete RAP de Firby, aunque posteriormente se utilizó ELS. ELS es un lenguaje para codificar conocimiento en agentes autónomos embebidos y ha sido diseñado para ser el substrato implementado en el secuenciador en la arquitectura de tres capas considerada.

  El secuenciador debe ser capaz de responder rápidamente a eventos mientras procesa grandes cantidades de información. Es también capaz de tratar con una variedad de estrategias diferentes para asignar responsabilidades a las otras capas (desde reactiva hasta planificador).

  El manipulador de contingencias del secuenciador está basado en el concepto de "fallo cognitivo", el cual es una filosofía de diseño para detectar los fallos cuando estos ocurren y que así el sistema pueda responder apropiadamente. Esta aproximación presume que las múltiples posibilidades de acciones pueden ser fácilmente clasificadas en éxito o fracaso.

- El deliberador: es el responsable de ejecutar las tareas que consumen tiempo computacional como pueden ser la planificación y el modelado del entorno. El deliberador ejecuta los cálculos bajo el control del secuenciador. Todos los cómputos deliberativos son iniciados y terminados por el secuenciador. Los resultados de los cómputos deliberativos son situados en una base de datos la cual es accesible por el secuenciador.




# 3

# ARQUITECTURA AD

Ninguna época ha sabido tanto y tan diversas cosas del hombre como la nuestra... Pero ninguna otra época supo, en verdad, menos, qué es el hombre.
HEIDEGGER

Todas las teorías son legítimas y ninguna tiene importancia. Lo que importa es lo que se hace con ellas.
J. L. BORGES

Así como perfeccionamos las ciencias, debemos perfeccionar la moralidad, sin la cual el saber se destruye.
J. NEWTON

# Arquitectura AD 3

## 3.1 Introducción

Tal y como se comentó en el capítulo anterior, a la hora de considerar una nueva arquitectura de control en el mundo de la robótica, la primera cuestión es ver cómo se distribuye la capacidad de razonar y la de actuar dentro de la arquitectura. La nueva arquitectura pretende aunar planificación y acción reactiva, considerándose pues como una arquitectura híbrida.

Para la realización de la nueva arquitectura de control se va a considerar la forma de organizar los procesos mentales los humanos. Una vez más, la naturaleza va a proporcionar modelos a imitar. Es por tanto, la capacidad de razonar y de actuar los seres humanos lo

que va a motivar la nueva arquitectura. En este capítulo se hace una reflexión sobre la forma de organizar los procesos mentales los humanos, que van a servir de motivación para desarrollar la nueva arquitectura. Como consecuencia de estos razonamientos, se presenta la nueva arquitectura AD (Automatic-Deliberative Architecture).

## *3.2 Motivación*

Una metodología tradicional, empleada para el desarrollo de nuevas tecnologías, se basa en la imitación de la naturaleza: aviones con forma de ave, manipuladores antropomórficos, etc. Esta metodología puede ser especialmente útil cuando se trata de crear máquinas con capacidades semejantes a las de los seres vivos. Si algo caracteriza a los animales es su autonomía, y en particular a los seres humanos su inteligencia. Por lo tanto, no parece descabellado, si queremos desarrollar máquinas autónomas e inteligentes, intentar imitar, en la medida de lo posible, los procesos mentales de los seres humanos. Para ello habrá que definir, de manera meramente funcional, cómo el ser humano organiza las actividades mentales relacionadas con sus movimientos. Nos centraremos tan sólo en el movimiento, puesto que son estas capacidades, y no muchas otras del ser humano, las que en principio se pretenden imitar en los robots móviles.

### 3.2.1 Actividades mentales del ser humano

En el ser humano cabe diferenciar dos niveles de actividad mental:

- Nivel deliberativo: Se caracteriza porque en él se realizan actividades de forma consciente. Otra propiedad importante de los procesos mentales que se llevan a cabo en este nivel es la dimensión temporal. Una actividad puede haber sido consecuencia de un proceso anterior de planificación, y tras realizarse ir seguida de un proceso posterior de evaluación. A diferencia del nivel automático, las actividades del nivel deliberativo se llevan a cabo secuencialmente, es decir, una después de otra y no es posible realizar más de una actividad deliberativa al mismo tiempo.

- Nivel automático. Se caracteriza porque en él se controlan movimientos de forma automática, y sin necesidad de que se tenga consciencia de los procesos mentales encargados de controlar dichos movimientos. Ejemplos de este tipo de actividades serían el movimiento del corazón, el movimiento de la mano al escribir o el de las piernas cuando se anda. Una actividad automática puede llevarse a cabo en paralelo con otras actividades automáticas y con una actividad deliberativa. Por ejemplo, una persona puede simultáneamente estar conduciendo un vehículo, parpadeando y manteniendo una conversación con otra persona. El nivel de complejidad de las actividades automáticas puede ser muy variable y puede ir desde la "sencillez" de mover un dedo, hasta la complejidad de interpretar al piano una sonata previamente memorizada.

El tiempo que transcurre desde que se recibe un estímulo hasta que se realiza una acción a consecuencia del mismo es muy superior cuando el proceso se lleva a cabo en el nivel deliberativo que cuando se lleva a cabo en el nivel automático. En los animales un primer grupo de capacidades automáticas transmitidas genéticamente lo constituyen aquellas que son necesarias desde el nacimiento, por lo que no es posible adquirirlas por aprendizaje; y que requieren un tiempo de respuesta muy corto, por lo que no se pueden realizar de forma deliberativa. Un ejemplo sería la habilidad de alejar la mano de algo que quema o produce dolor.

### 3.2.2 Actividades automáticas

Las actividades automáticas elementales pueden ser de dos tipos: perceptivas y sensimotoras. Las habilidades complejas se forman fusionando habilidades elementales, tanto sensimotoras como perceptivas. Como se ha comentado antes, las actividades automáticas, a diferencia de las deliberativas, pueden realizarse de forma concurrente. No hay que confundir la ejecución simultánea e independiente de varias habilidades con la fusión de varias de estas habilidades para dar lugar a una única habilidad más compleja.

### 3.2.2.1 Habilidades perceptivas

Las habilidades perceptivas sirven para interpretar la información sensorial. Hay que resaltar la complejidad de este proceso (no es lo mismo el lenguaje que un conjunto de ondas sonoras.) Una de las preguntas que se planteó la filosofía griega estaba relacionada con la generalización que implica la interpretación de la información sensorial. La palabra caballo no se refiere a un caballo en particular, sino a cualquiera. Platón afirmaba que, de alguna manera, existe un caballo ideal fuera del espacio y del tiempo, y que la idea es lo real, lo particular es sólo aparente. Sin llegar a tanto, hoy en día podemos afirmar que la interpretación de patrones sensoriales son habilidades que pueden adquirirse genéticamente o por aprendizaje al igual que las habilidades motoras. El concepto de caballo es algo que un ser humano aprende generalizando los elementos característicos de todos los caballos que ve en su infancia o etapa de aprendizaje básico. Este proceso no puede separarse del de la adquisición del lenguaje. Cuando un padre enseña a su hijo lo que significa la palabra caballo, repitiéndola cada vez que éste ve un caballo, de hecho está enseñando al niño a asociar la visión de un caballo particular con el concepto general de caballo. Este tipo de interpretación de la información sensorial consiste en esencia en un proceso de clasificación.

Se pueden considerar habilidades perceptivas más complejas obtenidas por fusión de varias habilidades perceptivas. Estas habilidades permiten obtener información del entorno de una manera mucho más precisa. La complejidad inherente de estas habilidades requiere una capacidad de procesado de información muy elevada. Los seres vivos aprenden a sacar el máximo rendimiento de la fusión de la información sensorial de forma subconsciente; es un hecho que el cerebro diseña, depura y aplica complejos heurísticos sobre las diversas informaciones sensoriales que recibe. Estos heurísticos son fruto del extremadamente sofisticado sistema de aprendizaje que los seres vivos emplean de forma intuitiva.

### 3.2.2.2 Habilidades sensimotoras

Las habilidades sensimotoras permiten realizar movimientos de forma coordinada. Son habilidades que llevan a cabo acciones de movimiento, recibiendo realimentación de

los sistemas sensoriales. Pueden consistir en acciones muy simples como el movimiento de un brazo, o en acciones más complejas como el conducir. Las habilidades sensimotoras más simples son de tipo continuo y se llevan a cabo normalmente con realimentación sensorial propioceptiva. La habilidad de posicionar a voluntad un miembro de nuestro cuerpo es un ejemplo de habilidad sensimotora elemental.

Las habilidades sensimotoras pueden combinarse con habilidades perceptivas, dando lugar a comportamientos en forma de eventos discretos.

Por ejemplo, la habilidad de llenar un vaso con agua de una jarra consiste en:

Mover la mano abierta hacia la jarra (habilidad sensimotora) hasta que...
se produce el contacto (evento detectado mediante una habilidad perceptiva).
Agarrar el asa (habilidad sensimotora) hasta que...
se detecta que el asa está fuertemente sujeta (evento detectado mediante una habilidad perceptiva).
Mover la jarra hacia el vaso (habilidad sensimotora) hasta que...
se detecta que la jarra está encima del vaso (evento detectado mediante una habilidad perceptiva).
Inclinar la jarra (habilidad sensimotora) hasta que...
empiece a caer agua (evento detectado mediante una habilidad perceptiva).
Mantener un determinado caudal de agua (habilidad sensimotora) hasta que...
el vaso esté lleno (evento detectado mediante una habilidad perceptiva).
Enderezar la jarra (habilidad sensimotora) hasta que...
deje de caer agua (evento detectado mediante una habilidad perceptiva).
Mover la jarra hacia la mesa (habilidad sensimotora) hasta que...
esté sobre la mesa (evento detectado mediante una habilidad perceptiva).

### 3.2.3 Actividades deliberativas

Las actividades mentales deliberativas tienen cuatro características que las diferencia claramente de las habilidades automáticas.

- Carácter secuencial. Las actividades mentales deliberativas se llevan a cabo secuencialmente. Es decir, no es posible estar reflexionando sobre más de una cosa al mismo tiempo.

- Carácter temporal. Reflexionar conlleva utilizar los conceptos pasado y futuro. Las acciones pasadas son analizadas, a fin de descubrir las causas tanto de los éxitos como de

los errores y poder explotar los primeros y no repetir los segundos. Es decir, adquirir experiencia. Deliberar es también imaginarse posibles acciones futuras, para prever las consecuencias de cada una de ellas y de esa forma elegir la que mejor se adapte a la consecución de unos determinados objetivos: planificar.

- Adaptabilidad. La capacidad de análisis de los seres humanos es el catalizador de los procesos de adaptación que se requieren para poder responder adecuadamente a las situaciones con las que se va encontrando a lo largo de su existencia. Los seres vivos se encuentran en innumerables ocasiones con situaciones nuevas; el análisis de forma deliberativa permite determinar la respuesta adecuada para superarlas y, si no se elige la respuesta más conveniente, se podrá tener en cuenta para la siguiente vez que se deba reaccionar ante algo similar, adaptándo la forma de actuar según el historial adquirido, es decir, el aprendiendo.

- Poca rapidez de respuesta. El gran inconveniente de las actividades deliberativas es que requieren una cantidad de tiempo, dedicado al análisis, mucho mayor que las automáticas.

## *3.3 Arquitectura AD*

La arquitectura desarrollada en la presente tesis tiene como precedentes las arquitecturas por capas propuestas por Firby, Gat y Bonasso. En la tesis se desarrolla una arquitectura que integra los niveles deliberativos y reactivos.

La arquitectura ha sido construida teniendo en cuenta las motivaciones anteriores. De acuerdo con las teorías de la psicología moderna de Shiffrin y Schneider [Shiffrin77][Shiffrin88] hay dos mecanismos de procesamiento de la información: los procesos automáticos y los controlados. Aunque no parece existir un único criterio capaz de diferenciar un tipo de procesos del otro, puede decirse que el rasgo principal de los procesos automáticos es que apenas requieren la intervención de la atención. Por el contrario, los procesos controlados son aquellos que requieren atención. Se trata de

procesos que por lo general consumen recursos de procesamiento, por lo que suelen mediar las interferencias entre tareas.

Por este motivo, una ventaja de las tareas automatizadas es la posibilidad de que se lleven a cabo en presencia de otras tareas sin que aparezcan interferencias entre ellas. A la ausencia de inferencias con otras tareas hay que sumar la rapidez, gracias a la cual se pueden lograr niveles óptimos de ejecución. Estas características hacen que los procesos automáticos sean de gran valor adaptativo. No obstante, es preciso señalar que aquellas tareas con un fuerte componente de procesos automáticos presentan una dificultad de modificación que en circunstancias determinadas pueden convertirlas en desadaptativas.

De acuerdo con este criterio, establecemos únicamente dos niveles (figura 3.1). El nivel deliberativo lleva asociado procesos controlados y el nivel automático procesos automáticos. El nivel deliberativo estará compuesto por aquellos procesos que requieran un tiempo elevado de cálculo como consecuencia de un razonamiento. El planificador de trayectorias, el modelado del entorno, el supervisor de tareas son módulos que se encuentran dentro de este nivel. El nivel automático esta compuesto por los módulos que interactúan con los sensores y los actuadores y que requieren un tiempo mínimo para procesar la información con la que trabajan. Entre ellos se tendrán los módulos de información sensorial, los módulos de actuación sobre los distintos elementos mecánicos del robot.

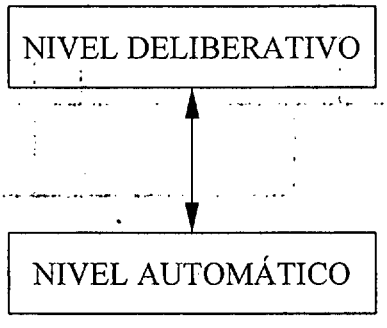



**Figura 3.1** Niveles de la arquitectura AD

La distribución de los módulos dentro de cada nivel y las relaciones entre ellos se puede apreciar en la figura 3.2.

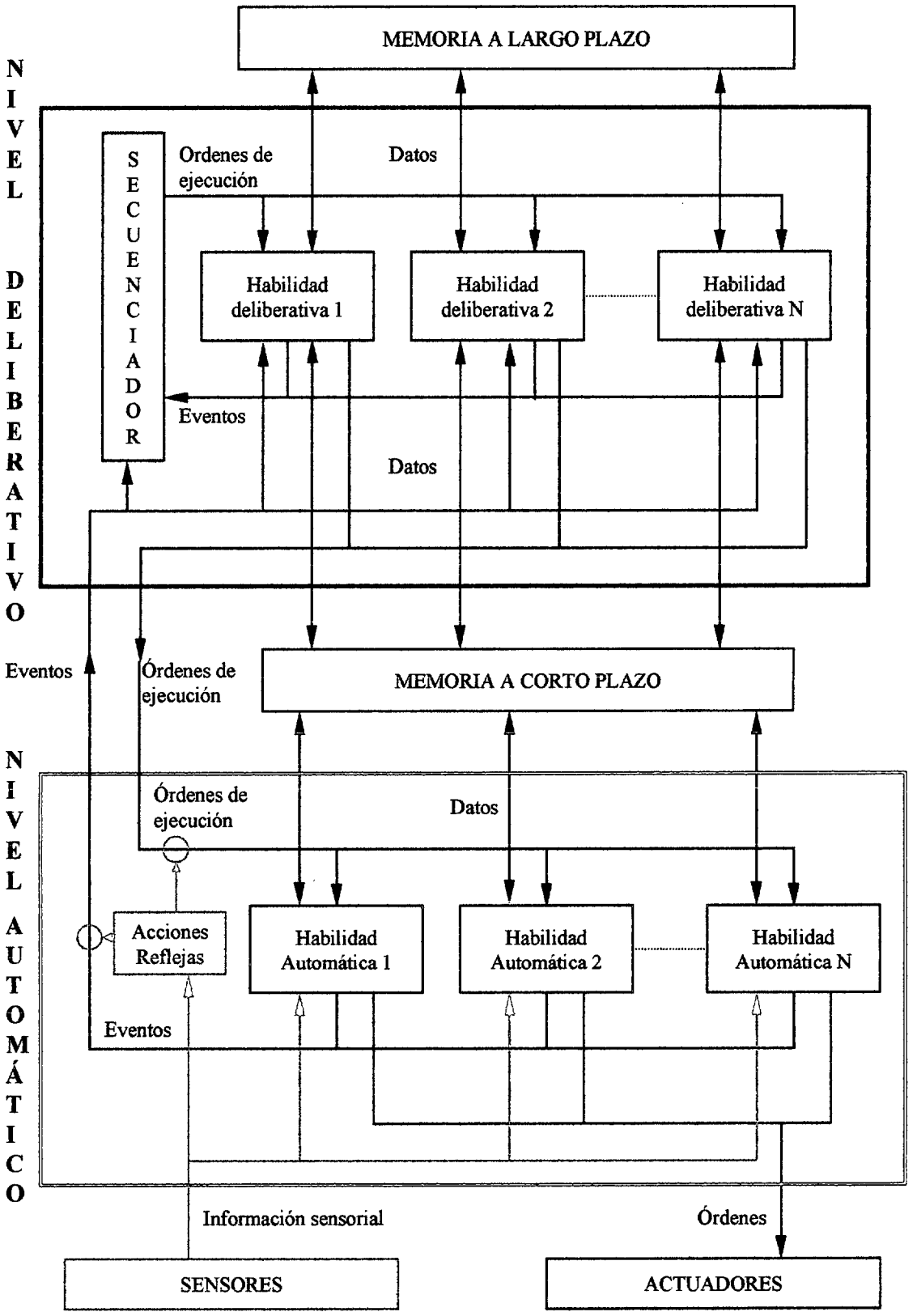



**Figura 3.2** Esquema general de la arquitectura AD

Los dos niveles presentan una característica en común: ambos niveles están formados por habilidades. Las habilidades consisten en las diferentes capacidades de realizar un razonamiento o llevar a cabo una acción. Dichas habilidades son activadas por órdenes de ejecución de otras habilidades o de un secuenciador, y devuelven datos y eventos a las habilidades o secuenciadores que las han activado. Dichas habilidades dan nombre a la arquitectura: AD (Automatic-Deliberative Architecture).

En este esquema se puede apreciar los elementos que forman parte de cada uno de los niveles:

### 3.3.1 Nivel Deliberativo

Este nivel está formado por una serie de habilidades llamadas habilidades deliberativas, una memoria a largo plazo de donde se obtiene información y un secuenciador que activa y desactiva las habilidades deliberativas.

- **Habilidades deliberativas:** Son cada una de las capacidades de razonamiento y aprendizaje de que dispone el sistema autónomo. Ejemplos de estas habilidades son los planificadores y los sistemas de relocalización. Conllevan la necesidad de un elevado tiempo de cálculo. Se activan y desactivan una a una, no pudiendo haber concurrencia.

- **Memoria a largo plazo:** Contiene información que puede ser considerada más estable a lo largo del tiempo, es decir, no dependiente del estado del robot. A dicha memoria sólo tiene acceso las habilidades deliberativas, las cuales pueden realizar razonamientos sobre dicha información, modificando la información cuando sea necesario. En ella se incluirán la información a priori, como pueden ser los mapas, e información procedente del razonamiento o aprendizaje de las distintas habilidades deliberativas.

- **Secuenciador:** El secuenciador se encarga de gestionar las habilidades deliberativas, dando la orden de ejecución de cada una de ellas en el momento

oportuno. Este secuenciador viene dado a priori y es el que define el comportamiento del sistema. El secuenciador va decidiendo que habilidades debe activar en función de su secuencia y de los eventos que le vayan llegando tanto de cada una de las habilidades de este nivel como del nivel automático.

Dicho nivel será desarrollado de forma más explícita en el capítulo 4 de la presente tesis.

### 3.3.2 Nivel Automático

En este nivel se encuentran las habilidades automáticas que pueden ser ejecutadas o activadas por el nivel deliberativo, por las acciones reflejas o bien se ejecutan de forma continua desde que es inicializado el sistema (un ejemplo de ello es el corazón, que funciona continuamente sin que la persona tenga control sobre él). Estas habilidades reciben directamente información del sistema sensorial y actúan directamente sobre el sistema locomotor del robot. En este nivel se encontrarán los módulos de control a bajo nivel, que actúan directamente sobre los accionadores, así como los módulos que recogen datos de los distintos sensores del sistema.

En la figura 3.2 se aprecian los distintos elementos que forman el nivel automático:

**Habilidades Automáticas:** Son las capacidades tanto sensoriales como motoras de que dispone el sistema. Son activadas mediante los secuenciadores de las habilidades del nivel deliberativo, al que le devuelve información en forma de datos y eventos, o mediante las interrupciones producidas por las acciones reflejas. Estas habilidades toman información directamente de los sensores y ejecutan ordenes directamente sobre los actuadores, ya sean al sistema de locomoción del robot o sobre los actuadores que manejan los dispositivos sensoriales y sus elementos auxiliares.

**Acciones Reflejas:** Las acciones reflejas proceden directamente de los sensores y consisten en información de carácter prioritario. Dichas acciones son tratadas a modo de

interrupciones. Producen una señal que deshabilitan las órdenes del nivel deliberativo, teniendo un tratamiento prioritario. Un ejemplo de estas acciones reflejas puede ser el camarero que lleva una bandeja y es picado por un mosquito. Aunque su nivel deliberativo le comunique que lleve la bandeja recta, el picotazo del mosquito genera una interrupción prioritaria que hace que mueva el brazo, contraponiéndose a su nivel deliberativo, con riesgo de que se le caiga la bandeja. La interrupción llega a modo de eventos tanto a las habilidades del nivel automático como a las habilidades y al secuenciador del nivel deliberativo. Dicha interrupción deshabilita momentáneamente las órdenes del nivel deliberativo y será tratada por un módulo que indique que hacer en el caso de que llegue la interrupción.

Dicho nivel será desarrollado de forma más explícita en el capítulo 5 de la presente tesis.

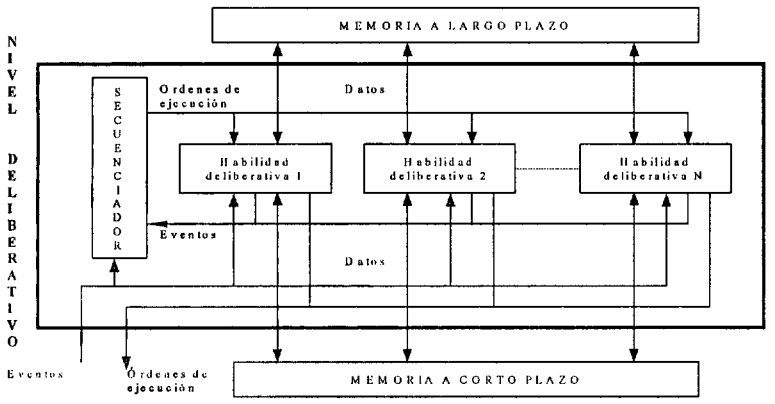
### 3.3.3 Comunicación entre los niveles deliberativo y automático

Entre los niveles deliberativo y automático existe una comunicación bidireccional. El nivel deliberativo comunica al nivel de reactivo las órdenes de ejecución de las habilidades automáticas, así como pasa datos al nivel inferior mediante la memoria a corto plazo, como por ejemplo los parámetros necesarios para la ejecución de alguna de las habilidades automáticas.

El nivel automático envía al nivel deliberativo eventos que son interpretados por los secuenciadores de las habilidades deliberativas para dar por concluida una habilidad automática o lanzar otra, así como también pueden ser recogidos por el secuenciador de este nivel y provocar activaciones y desactivaciones de habilidades deliberativas. El nivel automático también puede enviar datos al nivel deliberativo a través de la memoria a corto plazo, como por ejemplo información proveniente de los sensores y que deban ser utilizada por las habilidades deliberativas o almacenados en la memoria a largo plazo.

La memoria a corto plazo, aparece pues, como elemento de intercambio de información entre los dos niveles de la arquitectura. Esta se diferencia de la memoria a

largo plazo en que la información que se almacena en ella es la información acerca del estado del robot, mientras que en la memoria a largo plazo se almacenan datos que pueden considerarse permanentes y no dependientes del estado del robot.




# 4

# EL NIVEL DELIBERATIVO

Querer pensar es una cosa y otra es tener talento para pensar.
LUDWIG WIYYGENSTEIN

El pensamiento es grande, rápido y libre; la luz del mundo y la gloria principal del hombre.
BERTRAND RUSELL

Me dan miedo los hombres demasiado inteligentes que se convierten en máquinas de pensar, porque, como máquinas, no tienen corazón.
S. MILL

# El Nivel Deliberativo 4

## *4.1 Introducción*

Cuando se habla de deliberación, siempre viene asociada a la inteligencia. En el mundo de la robótica pocos son los autores que describen una capa deliberativa completa, aunque muchos son los que utilizan la inteligencia artificial para resolver problemas puntuales. Así, es abundante la literatura que describe agentes inteligentes empleados en robótica como numerosas técnicas de inteligencia artificial que resuelven algún problema en concreto.

Estas técnicas normalmente son utilizadas tanto en módulos deliberativos como en automáticos, encargados de realizar una determinada tarea en la que se necesite de aprendizaje o de razonamiento. Así, módulos deliberativos que aparecen frecuentemente en robótica son los planificadores, los módulos de monitorización y los módulos encargados de modelar el entorno. Los planificadores deben de establecer una trayectoria o un plan, a partir de la información del mapa, utilizando un determinado criterio. Los monitores o supervisores deben revisar que el plan se ejecute correctamente, tomando decisiones en caso de que no sea así. Los módulos de modelado del entorno tienen que ser capaces de analizar la información proveniente de los sensores y tomar decisiones para construir el modelo del entorno.

Sin embargo estas técnicas son sólo empleadas para solventar problemas muy concretos, sin llegarse a hablar de una arquitectura que sea inteligente o que agrupe la inteligencia en una sola capa, como se propone en esta tesis. Una excepción son las arquitecturas tradicionales planificadas, en las que toda decisión debe ser tomada en el nivel de deliberación.

En dichas arquitecturas planificadas o jerárquicas, el plan se establece previamente a la ejecución por el planificador. Una vez que se ha comenzado a realizar la tarea, el robot deberá seguir completamente el plan. Si aparece un imprevisto, habrá que comunicarlo al nivel deliberativo para que establezca un nuevo plan o tome una decisión de cómo actuar ante dicho imprevisto. Estas arquitecturas mostraban una capa deliberativa muy importante, pero adolecían de una capa reactiva que evitara el excesivo razonamiento de la capa deliberativa. Un excesivo deliberar hace que el sistema evolucione de manera más lenta, y además no es acorde con la realidad. El hombre tiene una capacidad de deliberación muy grande, pero hay una gran cantidad de tareas que se dejan para el nivel automático, permitiendo a la capacidad de razonar que atienda otros menesteres que sí necesiten su atención. Así, cuando queremos ir de un sitio a otro, podemos dar los pasos uno a uno de forma consciente, o podemos dar los pasos de forma automática, sin ocupar la inteligencia en ello y para poder razonar otras cosas.

Esto mismo es lo que se pretende en esta tesis. Plantear una capa deliberativa donde se agrupen todas las habilidades que requieran capacidad de razonamiento, pero que no evite la necesaria reactividad del nivel inferior.

## *4.2 Nivel Deliberativo*

En este nivel se encuentran los módulos que requieren capacidad de razonamiento o decisión. Éstos módulos no proporcionan respuestas inmediatas, ya que requieren procesar la información con la que trabajan. Estos módulos formarán las habilidades deliberativas, y serán activados por un secuenciador, que se encargará de gestionar el correcto funcionamiento de estas habilidades. El esquema de este nivel es el que aparece en la figura 4.1.

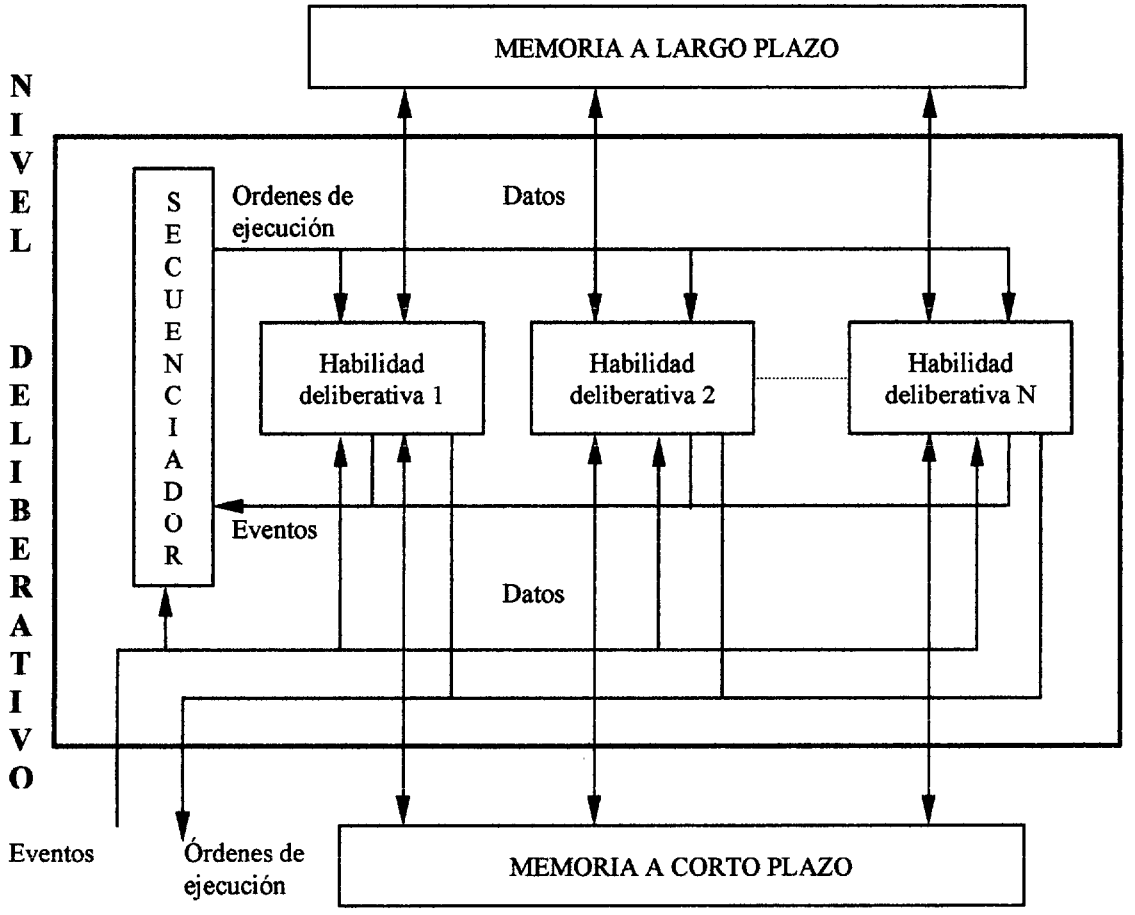



**Figura 4.1** Nivel Deliberativo

En dicho esquema se puede apreciar los elementos que forman parte de este nivel:

- **Habilidades deliberativas:** Son cada una de las capacidades de razonamiento y aprendizaje de que dispone el sistema autónomo. Ejemplos de estas habilidades son los planificadores y los sistemas de relocalización. Conllevan la necesidad de un elevado tiempo de cálculo. Se activan y desactivan una a una, no pudiendo haber concurrencia. Algunas de estas habilidades tienen a su vez un secuenciador, que es utilizado para secuenciar habilidades del nivel inferior. Por lo tanto, algunas de las habilidades deliberativas van a tener capacidad de gestionar habilidades del nivel automático.

- **Memoria a largo plazo:** Contiene información que puede ser considerada más estable a lo largo del tiempo, es decir, no dependiente del estado del robot. A dicha memoria sólo tiene acceso las habilidades deliberativas, las cuales pueden realizar razonamientos sobre dicha información, modificando la información cuando sea necesario. En ella se incluirán la información a priori, como pueden ser los mapas, e información procedente del razonamiento o aprendizaje de las distintas habilidades deliberativas.

- **Secuenciador:** El secuenciador se encarga de gestionar las habilidades deliberativas, dando la orden de ejecución de cada una de ellas en el momento oportuno. Este secuenciador viene dado a priori y es el que define el comportamiento del sistema. El secuenciador va decidiendo que habilidades debe activar en función de su secuencia y de los eventos que le vayan llegando tanto de cada una de las habilidades de este nivel como del nivel automático.

Es importante hacer notar que una característica fundamental dentro del nivel deliberativo frente al nivel automático es que este primero es secuencial. Esto quiere decir que no se pueden ejecutar dos habilidades deliberativas en paralelo. No se puede intentar resolver dos problemas o realizar dos tareas en las que haya que pensar exactamente a la vez. Una debe preceder a la otra. Sin embargo, en el nivel automático esto si que puede

ocurrir. Podemos andar mientras comemos algo y el corazón bombea la sangre a todo el cuerpo. Esta característica se ha tenido en cuenta a la hora de diseñar la capa deliberativa.

El nivel deliberativo se comunica con el nivel automático mediante dos líneas y a través de la memoria a corto plazo Por la primera línea las habilidades del nivel deliberativo da las ordenes de ejecución de las habilidades del nivel automático, Por la segunda de ellas, le llegan los eventos generados por las habilidades automáticas. El intercambio de datos se realiza a través de la memoria a corto plazo. En ella se guardarán los datos que afecten al estado del robot y a los que deban tener acceso tanto el nivel deliberativo como el automático.

## *4.3 Habilidades deliberativas*

Son cada una de las capacidades de razonamiento y aprendizaje de que dispone el sistema autónomo. Estas habilidades se encargan tanto de gestionar y modificar la memoria a largo plazo como de gestionar y secuenciar las habilidades del nivel automático. Las habilidades son gestionadas por un secuenciador. Dicho secuenciador se encarga de dar la orden de ejecutar la habilidad deliberativa y gestiona la información en forma de eventos que genera la habilidad considerada.

Ejemplos de estas habilidades pueden ser los planificadores, los supervisores, los módulos que gestionan el mapa, los módulos que realizan exploración, los módulos de modelado del entorno, los módulos de relocalización y los que gestionan las comunicaciones con el usuario. Algunos de ellos se describen a continuación.

### 4.3.1 Habilidad de monitorización

Es una habilidad que se encarga de supervisar la información proveniente de otras habilidades y que se encarga de actualizar la memoria a largo plazo. Recoge la información en forma de eventos y datos procedentes tanto del nivel automático como del resto de componentes del nivel deliberativo. Analiza dicha información y genera los eventos

oportunos hacia el secuenciador, o simplemente actualiza la memoria a largo plazo con la nueva información. Su esquema es el que aparece en la figura 4.2.

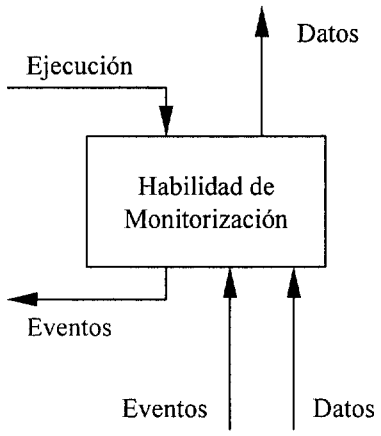



**Figura 4.2** Habilidad de monitorización

Por ejemplo, esta habilidad puede recopilar la información de los sensores durante la navegación, analizar dicha información y, en caso de que exista un obstáculo que permanezca permanentemente en el mismo lugar, actualizar el mapa para tenerlo en cuenta en planificaciones sucesivas. De la misma forma, si se detecta que una puerta está permanentemente cerrada, puede actualizarse el mapa para que el planificador tenga constancia de este hecho.

### 4.3.2 Planificador clásico

Esta habilidad se encarga de, a partir de los datos provenientes de la memoria a largo plazo, en forma de mapa principalmente, extraer la trayectoria en el caso de un planificador geométrico o de generar la lista de tareas en el caso de un planificador de tareas. Dentro del grupo de planificadores se pueden encontrar el planificador de tareas, el planificador de trayectorias o los planificadores que se encarguen de seleccionar los módulos de información sensorial más adecuados en cada situación. El planificador de tareas se encargará de generar la secuencia de acciones a realizar por el robot a partir del mapa. La secuencia de operaciones quedará representada en forma de una secuencia de tareas a realizar por el robot. Otro tipo de planificador es el planificador de percepción

desarrollado por Armingol en [Armingol95]. Estos planificadores tienen como característica principal que deben trabajar a priori, generando el plan a realizar por los módulos del nivel reactivo. Como resultado generará un plan que será almacenado en la memoria a largo plazo, donde puede ser obtenido a su vez por otra habilidad deliberativa para ordenar su ejecución. En la figura 4.3. se muestra un ejemplo de planificador clásico.

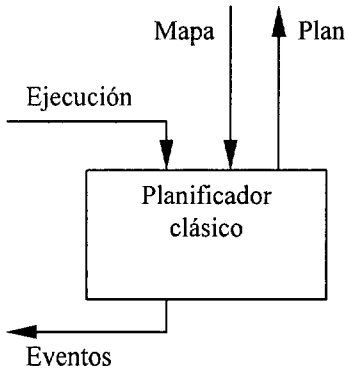



**Figura 4.3** Planificador clásico

### 4.3.3 Navegador clásico

Esta habilidad lee el plan almacenado en la memoria por el planificador y se encarga de dar las órdenes de ejecución de las habilidades del nivel automático. Dentro de esta habilidad se encuentra un secuenciador que gestiona las habilidades del nivel automático, dando las órdenes de ejecución de las diversas habilidades y recibiendo la información en forma de datos y eventos de los niveles inferiores. Este secuenciador es construido a partir del plan establecido. El esquema es el se representa en la figura 4.4.

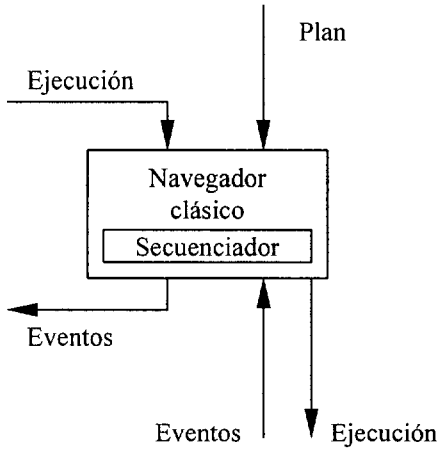



**Figura 4.4** Navegador clásico

### 4.3.4 Planificador basado en sensores

Estos planificadores van generando el plan a partir de la información procedente de los sensores. El plan generado es local, pues solo puede planificar hasta donde alcancen sus sensores. Esta habilidad recibe los datos procedentes del nivel automático y como resultado produce un plan que almacena en la memoria a largo plazo. El esquema aparece representado en la figura 4.5.

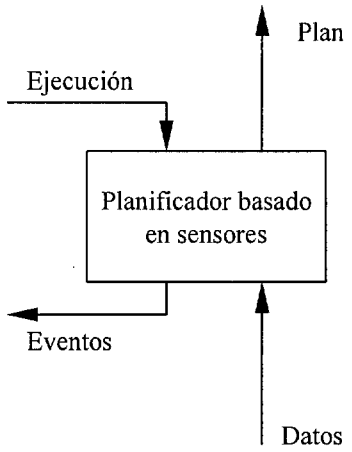



**Figura 4.5** Planificador basado en sensores

### 4.3.5 Navegador sin plan

Este navegador se corresponde con las arquitecturas reactivas más clásicas. Este planificador es prácticamente un secuenciador que recibe información directamente del nivel automático y genera órdenes para ejecutar las distintas habilidades automáticas. El esquema correspondiente es el representado en la figura 4.6.

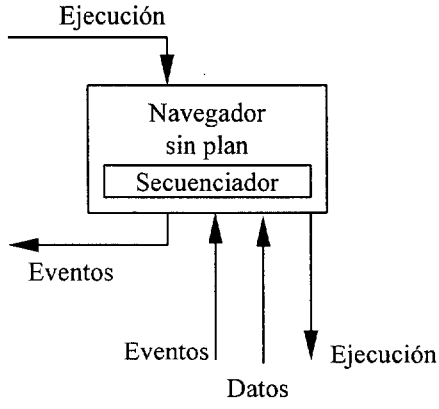



Figura 4.6 **Navegador sin plan**

### 4.3.6 Sistemas de relocalización

Estas habilidades se encargan de actualizar o corregir la posición del robot dentro del mapa (geométrico o topológico) en el caso de que el robot no tenga seguridad en cuanto a su posición. Los sistemas de relocalización estiman la incertidumbre de la posición del robot. Si esta incertidumbre sobrepasa un determinado umbral, se deben corregir los datos de la posición del robot, asegurando así que el robot tenga conocimiento de estar donde realmente está. El esquema de esta habilidad aparece en la figura 4.7.

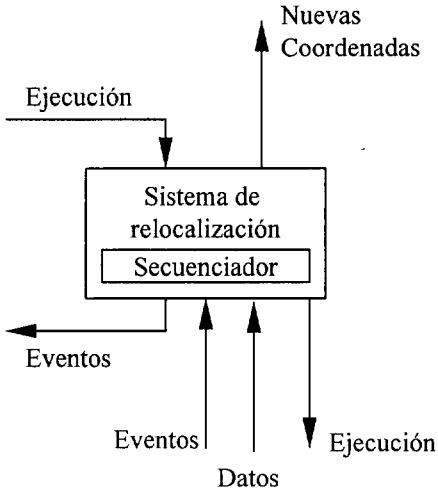



Figura 4.7 **Sistema de relocalización**

A partir de datos procedentes de los supervisores o directamente a través de la información sensorial y de algoritmos de decisión, resitúa al robot dentro de un mapa o un plan. Dicho relocalizador puede tener un secuenciador en el caso de que deba ordenar activar algún sistema sensorial como puede ser una óptica motorizada.

### 4.3.7 Exploración

La secuencia de acciones y decisiones que debe tomar el robot para navegar por un entorno desconocido e ir extrayendo información de él, definen esta habilidad deliberativa. El robot debe poseer un algoritmo que le permita moverse por un entorno desconocido mientras que va recopilando información sensorial, a partir de la cual debe ir construyendo el mapa. Dicha habilidad tendrá un secuenciador que ordene que habilidades del nivel

inferior deben ser activadas, tanto para navegar como para detectar el entorno. El esquema de esta habilidad se muestra en la figura 4.8.

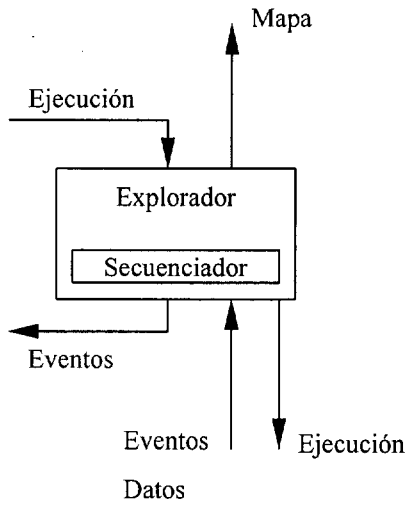



**Figura 4.8** Habilidad de exploración

### 4.3.8 Modelado del entorno

Entre estas habilidades se incluyen todas aquellas habilidades que requieran procesamiento de los datos provenientes de la información sensorial para, a partir de ellos, generar un modelo del mundo que rodea al robot. Esta habilidad estará ligada a las habilidades sensoriales del nivel inferior. La figura 4.9 nos muestra esta habilidad.

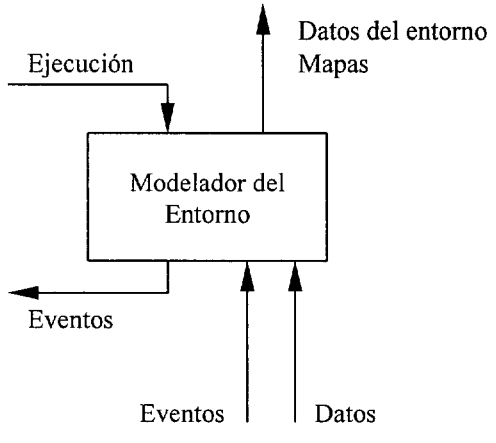



**Figura 4. 9** Habilidad de modelado del entorno

Este modelado puede hacerse mediante el tratamiento de los datos mientras el robot ejecuta una misión encomendada o bien procesando la información que se va generando durante la exploración

En este nivel pueden existir otras habilidades deliberativas. Pueden ser consideradas todas aquéllas que requieran tiempo de cálculo, aquéllas que gestionen las habilidades del nivel automático o aquéllas que procesen datos de la memoria a largo plazo.

## *4.4 La memoria a largo plazo*

En la memoria a largo plazo se encuentra toda la información sobre la que se va a razonar o tomar decisiones, y que pueda considerarse más estable en el tiempo. La información puede venir dada a priori o puede ser generada por habilidades deliberativas como producto de un aprendizaje o simplemente como resultado de procesar la información tanto interna como de la proveniente de los sensores, pasando información de la memoria a corto plazo (temporal) a la memoria a largo plazo (cuasi-permanente).

Una de las formas en que puede venir codificada la información es en forma de mapa. En los mapas viene codificada la información del entorno por el cual se va a mover el robot. Esta información, tanto si es geométrica como topológica, será utilizada por los planificadores. El ejemplo más clásico es el mapa geométrico, formado por las coordenadas geométricas de los distintos elementos del entorno, sobre el que un planificador establece el camino a seguir por el robot entre dos puntos. En los mapas también puede existir información utilizada para la relocalización, como por ejemplo la posición de marcas artificiales. A su vez, el mapa puede ser utilizado para hacer fusión sensorial o para hacer modelado del entorno.

Cabe destacar que el mapa puede venir dado a priori o ser producto de una exploración llevada a cabo por el robot. Existen mapas que son generados a priori y otros

que o bien son resultado de una exploración o aunque sean generados a priori, permiten una actualización dinámica como consecuencia de sucesivas navegaciones.

Otra de las formas en que puede venir dada esa información es en forma de reglas que puedan ser utilizadas por mecanismos de razonamiento como pueden ser los sistemas expertos o la lógica borrosa. También puede venir codificada la información en forma de pesos en el caso de las redes neuronales.

Cabe contemplar otros datos que pueden existir en esta memoria como pueden ser parámetros utilizados por las distintas habilidades, parámetros de configuración del robot o parámetros resultados de la calibración de los distintos sensores que posea el robot.

## *4.5 Los secuenciadores*

Dentro de este nivel existen varios secuenciadores. Uno de ellos es el secuenciador principal del nivel deliberativo, que se encarga de gestionar las habilidades deliberativas. Los otros secuenciadores son los que aparecen dentro de las habilidades deliberativas, que se encargan de gestionar las habilidades del nivel automático.

El secuenciador principal del nivel deliberativo, viene dado a priori y es fijo. Dicho secuenciador debe poseer la secuencia correcta de cómo activar todas y cada una de las habilidades deliberativas. Los secuenciadores de las habilidades deliberativas se encargan de gestionar la secuencia de habilidades automáticas representada por los planes. Dicha secuencia no tiene por que ser fija, dependiendo del plan generado por los correspondientes planificadores. Es más, la secuencia de operaciones a realizar por dichos secuenciadores puede modificarse dinámicamente debido a distintas alternativas basadas en situaciones con que se encuentre el robot.

En las arquitecturas por capas comentadas, el secuenciador forma una capa por sí solo, tomando un protagonismo excesivo dentro de dichas arquitecturas. En la arquitectura

propuesta, el secuenciador es un mecanismo que se encarga de lanzar o activar las distintas habilidades, pero no forma un nivel por sí sólo.

En la literatura aparecen distintos lenguajes para representar las secuencias. Ejemplos de ellos son el lenguaje utilizado en los Scripts por Murphy [Murphy96], el RAP creado por Firby [Firby89], ESL, creado por Gat [Gat97] y los lenguajes para modelar los sistemas de eventos discretos, como puede ser el Diagrama Funcional descrito por la norma internacional IEC 848, que es el lenguaje escogido en la presente tesis para representar la secuencia de actividades de un determinado plan.

### 4.5.1 Representación de secuencias mediante SCRIPTS

Murphy[Murphy96] utiliza scripts para coordinar y controlar una colección de comportamientos necesarios para ejecutar una tarea. Los scripts son secuencias de tareas que facilitan la planificación ya que representan explícitamente los tipos de situaciones, de comportamientos y de planes de actividad. Los scripts son especialmente útiles en el caso de la navegación topológica.

Los Scripts se basan en el lenguaje natural y son usados para representar secuencias de eventos. Una secuencia de eventos en un script se denomina *cadena casual*. Un script puede contener subsecuencias y tener punteros a otros subscripts.

Un script tiene un concepto principal u *objetivo* que se intenta alcanzar. También incluye *lugares* donde la cadena de secuencias es válida. Los *actores* especifican que eventos son ejecutados o inicializados. Los *props* son los objetos usados por los actores para llegar al objetivo. Los *actores* y los *props* se actualizan en tiempo de ejecución.

Los componentes de los scripts y las funciones que ellos habilitan, son idénticos a una colección de comportamientos. En primer lugar la cadena casual puede ser utilizada para representar la secuencia esperada de comportamientos (por ejemplo: busca la puerta, atraviesa la puerta y párate en el otro lado). El robot puede obviar un comportamiento (por ejemplo, si percibe la puerta, no la busca). La interpretación explícita de una secuencia

permite a un planificador deliberativo anticiparse a las demandas motoras y sensoriales y programar los recursos adecuadamente. La cadena casual puede incluir tareas de monitorización.

Un caso práctico para evaluar la utilidad de los scripts puede ser la navegación topológica. En la navegación topológica es frecuente el uso de directivas para guiar al robot por una ruta, frente a la exactitud métrica de la navegación geométrica. Un ejemplo de este tipo de navegación puede ser el guiado de un robot desde una habitación a otra, como ocurre en la figura 4.10.

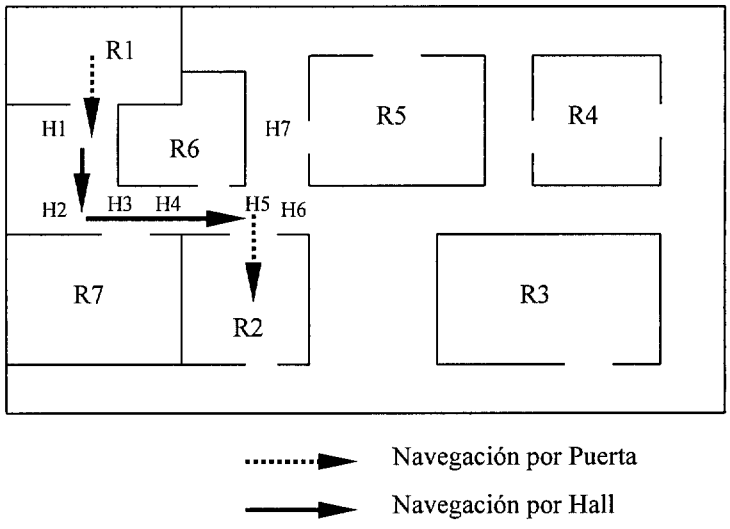



**Figura 4.10** Mapa que muestra las directivas topológicas desde R1 a R2

Para ir de la habitación R1 a la R2, de acuerdo con este mapa se puede expresar la siguiente secuencia de acciones: salir por la puerta, seguir el hall y doblar en la segunda puerta a la derecha. La cadena de comportamientos resultante podría ser: NavegaPorPuerta(), NavegaPorHall(terminar=HallIzquierdo), NavegaPorHall (terminar=PuertaDerecha,2), NavegaPorPuerta().

En este ejemplo, el comportamiento NavegaPorPuerta y NavegaPorHall pueden ser a su vez una colección de comportamientos y no un simple comportamiento atómico. Los comportamientos NavegaPorPuerta y NavegaPorHall consisten en dos scripts. El script NavegaPorPuerta es usado para pasar las puertas y tiene el siguiente pseudo código:

switch(puerta)
caso puerta-no-encontrada
    //fase de inicialización
    //sigue la pared hasta encontrar puerta
    si la pared es encontrada
        **seguirpared** hasta puerta
    sino
        **avanzar** hasta encontrar pared
caso puerta-encontrada
    //fase de actividad
    **muevete-hacia-puerta(localiza-puerta)**

El script NavegaPorHall es usado para recorrer pasillos y tiene el siguiente pseudo código:

switch(hall)
caso hall-no-encarado
    //fase de inicialización
    si empezamos en un VESTIBULO
        si hall-no-encontrado
            **seguirpared** hasta encontrar el hall
        sino
            si no hall encarado
                girar hasta encarar hall
    si empezamos en un HALL
        si no hall encarado
            girar hasta encarar hall
caso hall-encarado
    //fase de actividad
    **seguirhall** hasta siguiente paso

El script completo para ir desde la habitación R1 a la R2 se muestra a continuación. En este ejemplo se muestra la flexibilidad de los scripts.

R1->R2

R1 - H1 - H2 - H5 - R2

Moverse desde R1 hasta H1, yendo al SUR

En el comportamiento NavegarPuerta

buscar puerta hacia el: SUR

MOVER ADELANTE

Puerta encontrada – Inicialización terminada

MOVER A TRAVÉS PUERTA

Movido a través puerta – Comportamiento terminado

Moverse de H1 hacia H2, yendo al sur

En el comportamiento NavegarHall

girar hacia: SUR

Girado hacia el hall – Inicialización terminada

buscar hall hacia: ESTE

SEGUIR HALL

Hall encontrado – Comportamiento terminado

Moverse desde H2 hasta H5, yendo al ESTE

En el comportamiento NavegarHall

girar hacia: ESTE

Girado hacia el hall – Inicialización terminada

buscar (visión) puerta a 90 (lado derecho)

SEGUIR HALL

Puerta encontrada – Comportamiento terminado

Moverse desde H5 hasta R2, yendo al SUR

En el comportamiento NavegarPuerta

buscar puerta hacia el: SUR

MOVER ADELANTE

Puerta encontrada – Inicialización terminada

MOVER A TRAVÉS PUERTA

Movido a través puerta – Comportamiento terminado

Objetivo alcanzado!

### 4.5.2 Representación de secuencias mediante el lenguaje RAP

James Firby [Firby89], en su tesis de 1989 "Adaptative Execution in Complex Dynamic Worlds" propone la utilización de un lenguaje para secuenciar tareas llamado RAP. RAP aparece como una interfase entre la ejecución reactiva y el control continuo ilustrado en la figura 2.5. El sistema RAP toma tareas y las convierte en comandos para posibilitar los procesos adecuados de sensado y acción para la situación que encuentre en el tiempo de ejecución. Típicamente, los procesos se basan en "pasos" que llevarán acabo de manera segura una acción en el entorno durante un periodo de tiempo. Slack [Slack92] ha llamado a ese tipo de procesos "niveles reactivos". El sistema RAP produce un comportamiento dirigido a metas usando esta idea, definiendo planes de pasos abstractos en secuencias de configuraciones diferentes para un control del sistema basado en procesos.

Una vez que el conjunto de procesos ha comenzado, el sistema RAP cuenta con señales que le indican cuándo una actividad deseada está completa y cómo se llevó a cabo. Desde el momento en que la razón para invocar un conjunto de procesos no se conoce en los procesos mismos, (no se sabe que es lo que se tiene que hacer durante un proceso por estar en ese proceso sino que depende de acciones externas o de las percepciones sensoriales), el sistema RAP debe interpretar las señales en el contexto. Las mismas señales podrían significar diferentes cosas y diferentes planes. Por ejemplo, para alcanzar un determinado objetivo no se sabe si se tendrá que levantar un objeto que está fijo o si se tiene que seguir el objeto por todo el entorno. Una señal que indique que el objetivo ha

sido alcanzado puede significar bien que la tarea está completa o bien que el objeto está demasiado cerca cuando se le está siguiendo.

Así la descripción de una tarea RAP debe:

1. Permitir llevar a cabo ejecuciones concurrentes de tal forma que varios procesos puedan ser empezados a la vez.

2. Representar cuándo se debe proceder a la siguiente subtarea en un método dado y cuándo las tareas deben esperar ciertas señales.

3. Describir métodos para una tarea que permitan diferentes pasos cuando se perciban en el entorno cambios o se le den señales explícitas.

El RAP es un sistema diseñado para la ejecución reactiva de planes simbólicos. Un plan debe incluir metas, o tareas, en una variedad de diferentes niveles de abstracción. El sistema RAP logra llevar acabo cada tarea en orden usando métodos diferentes en situaciones diferentes y manejando problemas comunes e interrupciones simples.

En el sistema RAP las tareas se describen mediante RAPs (Reactive Action Packages) que constituyen representaciones que agrupan y describen todas las formas conocidas de llevar a cabo una tarea en diferentes situaciones. Los aspectos importantes de la descripción de tareas RAP son las secciones SUCEED y METHOD.

Por ejemplo la siguiente RAP describe como agarrar algo en un mundo simulado usado en el sistema de desarrollo de RAP inicial.

```
(define-rap(arm-pickup ?arm ?thing)
        (succed (ARM-HOLDING ?arm ?thing))
        (method
                (context (not (TOOL-NEEDED ?thing ?tool true)))
                (task-net
```

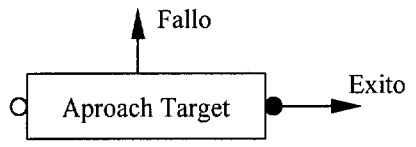
```
            (t1 (arm-move-to ?arm ?thing) (for t2))
            (t2 (arm-grasp-thing ?arm ?thing))))
    (method
        (context (TOOL-NEEDED ?thing ?tool true)))
        (task-net
            (t1 (arm-pickup ?arm ?tool) (for t2))
            (t2 (arm-move-to ?arm ?thing) (for t3))
            (t2 (arm-grasp-thing ?arm ?thing)))))
```

Esta tarea RAP tiene dos métodos para alcanzar la meta. La sentencia SUCCEED es un predicado que se comprueba en la memoria para ver si la tarea ha sido completada. Cada método especifica un plan, o una red de tareas, para alcanzar la condición de SUCCEED en un contexto dado. Como la sentencia SUCCEED, cada contexto es un predicado que será comprobado en memoria. De esta forma, se pueden escribir redes de tareas que conllevan subtareas, que dan lugar a la invocación de procesos e interpretación de señales para la ejecución de planes coherentes.

El sistema RAP lleva a cabo tareas usando el algoritmo siguiente: Primero, una tarea se selecciona para la ejecución y si representa una acción básica, se ejecuta directamente, de otra forma su RAP correspondiente se busca en la librería. Después de utilizar RAP para comprobar el SUCCEED de la situación actual, si se satisface, la tarea se considera completa y la siguiente tarea puede ejecutarse. Sin embargo, si la tarea no ha sido satisfecha, la aplicabilidad del método se revisa y uno de los métodos que satisfagan el test es seleccionado. Es decir, primero se comprueba la situación y de acuerdo a esa situación se elige un método y se intenta ejecutar la tarea de ese método y si no es posible realizarla se vuelve a revisar por si el entorno ha cambiado y hay que elegir otro método. Finalmente, las subtareas del método elegido se ponen a la cola para su ejecución en el lugar de la tarea prevista, que se suspende hasta que se elija el nuevo método. Cuando todas las subtareas del método se hayan ejecutado, la tarea se reactiva y su test de completo se examina de nuevo. Si todo fuera bien la condición de completo se satisfaría y el proceso de ejecución puede seguir con la siguiente tarea. Si no, la selección del método se repite y otro método se lleva a cabo.

Un aspecto importante en la representación y ejecución de un plan es el significado de una submeta o una subtarea. El sistema RAP se escribió originalmente asumiendo que las subtareas de RAP podían ser ejecutadas atómicamente, es decir, de forma no interrumpible. Desde el punto de vista del método que utiliza subtareas, podría tener éxito o fallar, y no se completará hasta que se conozca si ha tenido éxito o ha fallado.

Por ejemplo, la red de tareas mostrada en la figura 4.11 contiene una subtarea y una vez que la subtarea es tratada por el intérprete, el resto del proceso se parará hasta que la subtarea se complete. Se asume que la subtarea tendrá éxito, en cuyo caso el interprete continuaría procesando la red de tareas que van después de esa tarea o fallará, en cuyo caso el método completo fallará y todas sus subtareas se terminarían.




```
(task-net
        (t1 (approach-target ?target)))
```

**Figura 4.11** Una tarea simbólica discreta

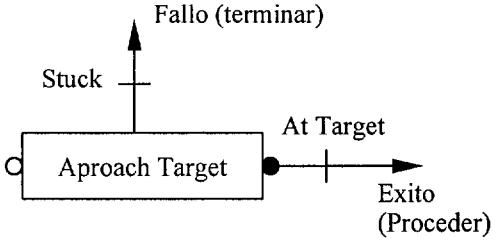
Esta representación y semántica de una subtarea (o método, o plan) asume que está preparada para ejecutar el siguiente paso en un método tan pronto como una subtarea se complete. Firby afirma que es una ventaja el hecho de que las tareas se lleguen a considerar como elementos atómicos a la hora de considerar si una ha tenido éxito o no.

Desafortunadamente, dado el bajo nivel del sistema de control que se utiliza en la arquitectura de Firby, ninguna de estas presunciones se puede mantener. Las metas son llevadas a cabo por conjuntos de procesos que deben ser habilitados de manera independiente y la detección de una meta completa depende de la interpretación apropiada

de señales que provienen de esos procesos. Así, los métodos RAP deben definir explícitamente qué señales significan que una subtarea ha tenido éxito y qué señales significan que ha fallado.

Cuando la actividad del robot se controla habilitando o deshabilitando conjuntos de procesos, el tiempo transcurre mientras que la actividad se está produciendo. Si el sistema RAP realiza la habilitación y deshabilitación de manera explícita, los métodos deben tener una manera de dejar que el tiempo pase y sincronizar la expansión de las tareas con el progreso del proceso.

Se adopta una anotación en la red de tareas que le dice al interprete que debe esperar a una señal dada antes de proceder con la siguiente subtarea en un método (esta idea se aproxima a la noción de McDermott [McDer92] de bloquear una tarea mientras se espera al flujo. Por ejemplo, un método para aproximar un objetivo fijo podría ser el de la figura 4.12.




```
(task-net
        (t1 (approach-target ?target)
        (wait-for (at-target) :proceed)
        (wait-for (stuck) :terminate)))
```

**Figura 4.12** Esperando una señal para proceder

El método que aparece ejecuta la subtarea approach-target y entonces espera a una señal bien *at-target* o bien *stuck*. Si recibe *at-target* entonces se ejecuta la subtarea succeeds y si se recibe *stuck*, la subtarea falla y otra subtarea en el método se encarga de terminar.

La red de tareas RAP sustenta métodos no lineales con tratamientos paralelos de ejecución. Esta habilidad de mantener tareas concurrentes es crítica cuando el sistema RAP se usa para posibilitar y deshabilitar procesos en casos concurrentes. Por ejemplo, se puede considerar el siguiente método:

```
(t1 (approach-target ?target)
    (wait-for (at-target) :proceed)
    (wait-for (stuck) :terminate))
(t2 (track-target ?target)
    (wait-for (lost-target) :terminate)
    (wait-for (camara-problem) :terminate)
    (until-end t1)))
```

Si dos acciones deben ser ejecutadas a la vez, se elige una para ser ejecutada la primera. El intérprete comienza el proceso indicado y bloquea ese procedimiento hasta que reciba una de las señales indicadas. Mientras que esta rama esté bloqueada, el intérprete sigue la otra rama, habilitando el otro proceso y bloqueando esa rama mientras espera una de sus señales. La anotación de until-end le dice al intérprete que la subtarea t2 se terminará cuando la tarea t1 se complete. La única manera de que el proceso se complete es recibiendo el *at-target* de la primera tarea.

Es muy importante que se sepa definir de manera explícita y estudiada si se quieren tareas que se ejecuten de manera síncrona o concurrente de acuerdo a lo que se va a encontrar en el entorno.

Cuando una actividad requiere habilitar una variedad de procesos y posteriormente esperar a que se dé algún evento en el entorno, se hace muy difícil codificar los pasos en el método para detectar cuándo ocurre un fallo.

Sería fácil en el caso anterior agregar la condición de que la cámara se encienda pero la situación se complica si queremos que se apague. Si se pierde la señal *at-target* y se produce la señal *lost-target* que implique el apagado de la cámara el método fallará.

La solución a la que llega es no considerar una situación de fallo o de éxito de manera aislada, la opción SUCCEED se convierte en una señal más que se espera para ejecutar otras acciones y no como una meta en sí. El *wait-for* toma un papel importante y el cambio es considerable.

En definitiva, la meta de RAP es definir un lenguaje y un interprete que permita planificar procesos arbitrarios tal que el mismo lenguaje se utilice para describir tareas y planes y también para describir lazos de realimentación de bajo nivel

### 4.5.3 Representación de secuencias mediante Diagramas Funcionales

Entre los secuenciadores utilizados en el mundo de la robótica también se encuentran los secuenciadores basados en el control de sistemas discretos. Así Pascoal y Oliveira[Pascoal97] utilizan una red de Petri para secuenciar las tareas que se han de realizar a la hora de llevar a cabo una misión. Buss y Scmidt [Buss96] utilizan un sistema híbrido basado en estados para que el robot sea capaz de alcanzar su objetivo.

Otro ejemplo de secuenciador puede venir en forma de Diagrama Funcional, a través del cual se le indica al robot cuando debe activar las distintas habilidades. Este es el secuenciador utilizado en la presente tesis. En él se indicará cuando deben activarse las distintas habilidades deliberativas, por ejemplo, cuando es necesario replanificar o cuando el robot debe relocalizarse.

El Diagrama Funcional es un modelo de representación gráfica para describir el comportamiento de cualquiera de las partes de ese sistema. El modelo Diagrama Funcional se define por la unión de tres partes (figura 4.13):

- los **elementos gráficos** de base, que definen la representación estática del modelo:

  - *Etapas*
  - *Transiciones*
  - *Enlaces orientados*

- la **interpretación**, traducción del comportamiento del sistema en función de sus entradas y salidas. Define el aspecto funcional del diagrama y se caracteriza por:

  - *Acciones*, asociadas a las etapas.
  - *Receptividades*, asociadas a las transiciones. Representan la condición de transición.

- las **reglas de evolución**, que definen formalmente el comportamiento dinámico del sistema descrito.

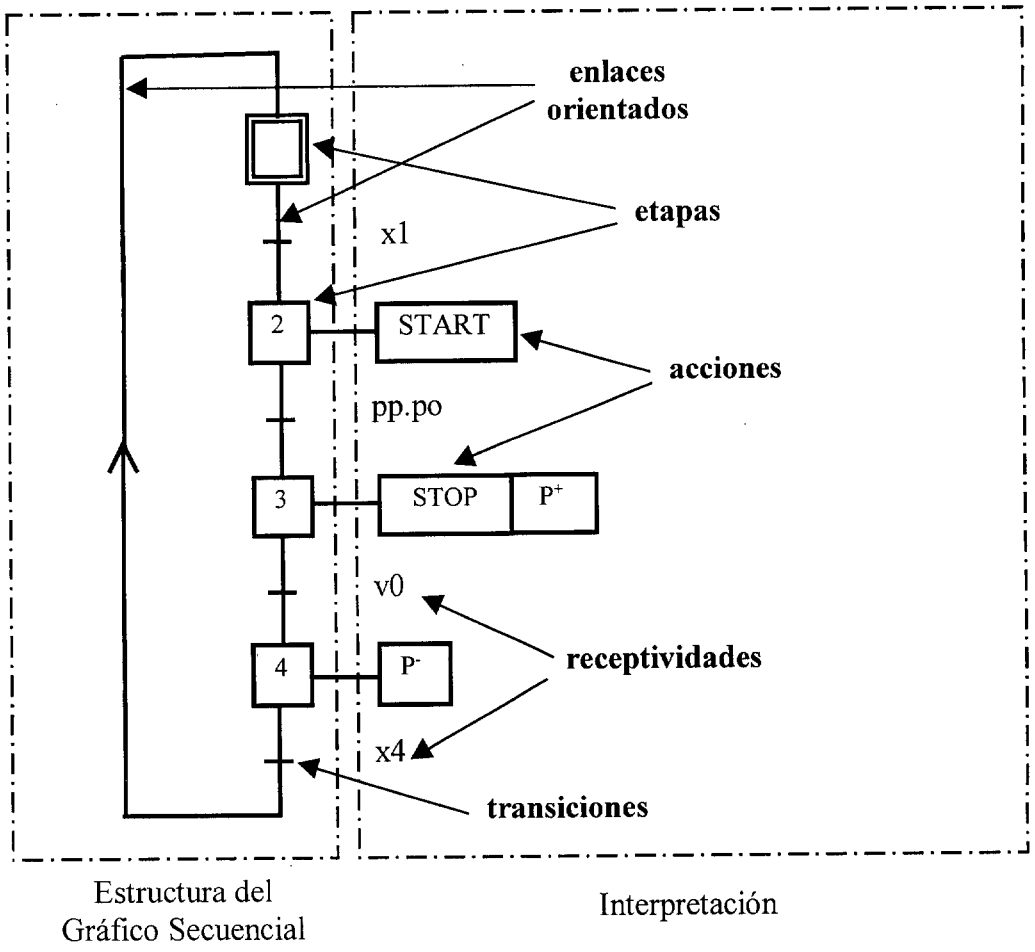



**Figura 4.13** Ejemplo de Diagrama Funcional

*4.4.3.1 Elementos del Diagrama Funcional*

- **Etapas**

Una etapa corresponde a un posible estado del sistema considerado, es decir, representa uno de los pasos del esquema secuencial que define el comportamiento global de ese sistema.

El símbolo normalizado para representar una etapa es un rectángulo (figura 4.14), de altura y longitud arbitrarias, aunque se recomienda la utilización de cuadrados.

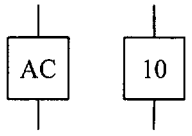



**Figura 4.14** Ejemplos de etapa

Para su identificación, a cada etapa se le asigna una etiqueta alfanumérica, que se escribe dentro del símbolo correspondiente.

En un instante de tiempo dado, cada etapa puede estar activa o inactiva. Un estado interno del sistema, o **situación,** se define por el conjunto de las etapas activas en ese instante de tiempo. Para visualizar dicha situación, se marcan las etapas activas dibujando un punto en el interior de su símbolo, como se aprecia en la figura 4.15. Este punto no pertenece al símbolo de la etapa, sino que sólo se utiliza con fines explicativos. También se pueden utilizar otros medios de discriminación visibles, por ejemplo cambiar el color del símbolo cuando se utiliza un monitor de vídeo.

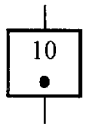



**Figura 4.15** Etapa activa

La **etapa inicial** del esquema es aquella que caracteriza el comportamiento inicial del sistema, es decir, la que está activa en el instante inicial de su funcionamiento. En algunos

casos se puede tener más de una etapa inicial. Para distinguirlas se representan con un símbolo similar al del resto de las etapas pero con doble raya (figura 4.16).

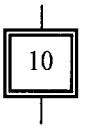



**Figura 4.16** Etapa inicial

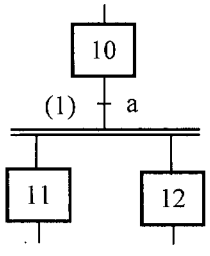
A cada etapa se le asocia una variable binaria representada por "Xi" donde la letra "i" corresponde a la etiqueta que identifica a la etapa en cuestión, por ejemplo X1 para la etapa 1, XAC para la etapa AC, etc. Esta variable indica la actividad o inactividad de dicha etapa en cada instante, de forma que:

Xi = 0, si la etapa i está inactiva.

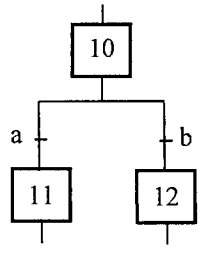
Xi = 1, si la etapa i está activa.

- **Transiciones**

Una transición indica la posibilidad de evolución de una situación a otra, es decir, la evolución de la actividad de una etapa a otra a través de un enlace directo.




Una transición:
- transición (1) ó 10/11,12 ó 10-11,12 ó 10°11,12
- receptividad 'a'




Dos transiciones:
- transición 10/11 con receptividad 'a'
- transición 10/12 con receptividad 'b'

**Figura 4.17** Ejemplos de transiciones

Las transiciones se representan mediante una pequeña raya transversal que aparece en el enlace directo entre las etapas correspondientes a esa evolución. Además, se pueden identificar asignándoles una etiqueta, o relacionándolas con las etapas que separan, como se indica en la figura 4.17.

- **Enlaces Orientados**

Definen las rutas de evolución entre etapas, conectando etapas con transiciones y transiciones con etapas.

Los enlaces pueden ser horizontales o verticales, aunque se permite la utilización de enlaces oblicuos cuando contribuyen a mejorar la claridad del diagrama.

Por convención, la dirección de evolución es de arriba hacia abajo. Pero, se pueden utilizar flechas para indicar la dirección en los casos en los que no se cumple con esta convención, por ejemplo en los lazos de retorno que implican evoluciones de abajo hacia arriba, o en aquellos en los que dichas flechas facilitan su comprensión.

El cruce de enlaces verticales y horizontales se permite cuando no existe relación entre dichos enlaces, pero debe ser evitado cuando los enlaces pertenecen a la misma transición.

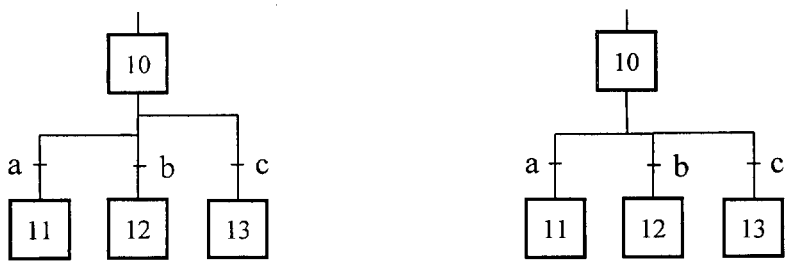



**Figura 4.18** Ejemplos de enlaces

Cuando un enlace debe ser interrumpido, por ejemplo en representaciones que ocupan varias páginas, hay que indicar el número de la siguiente etapa así como el número de la siguiente página, como se indica en la figura 4.19.

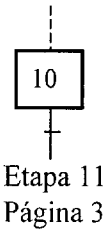



**Figura 4.19** Ejemplo de enlace interrumpido

- **Acciones**

Representan las acciones que se realizan cuando las etapas correspondientes están activas. Pueden ser:

- uno o más **comandos**, si el diagrama describe un sistema controlador,
- una o más **operaciones**, si el diagrama describe un sistema operativo.

Además, se pueden utilizar como indicaciones de estado, para informar del estado de alguna de las partes del sistema en el intervalo de tiempo en que dicha etapa está activa.

Las acciones se representan mediante un mensaje escrito, o un símbolo, dentro de un rectángulo que se conecta a la etapa asociada (figura 4.20). Si hay más de una acción por etapa se conectan a ella unidas horizontalmente o en vertical. Estas acciones correspondientes a una misma etapa no tienen porqué tener relación entre sí, aunque se representen juntas.

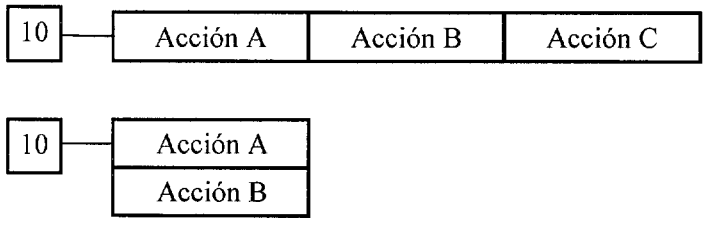



**Figura 4.20** Ejemplo de etapas con varias acciones

Si la definición de las acciones resulta confusa, se pueden introducir notas aclaratorias en el diagrama.

En función de su forma de terminación, las acciones pueden ser de dos tipos:

- **no mantenidas** : aquellas que terminan cuando la etapa correspondiente deja de estar activa. Si no se indica lo contrario las acciones se considerarán no mantenidas.

- **mantenidas** : aquellas que permanecen en su estado al desactivarse la etapa, siendo necesaria para su terminación otra etapa posterior que resetee dicha acción.

En función de la temporización podemos tener acciones:

- **retardadas** : aquellas que se realizan con un cierto tiempo de retardo respecto al instante de activación de la etapa correspondiente.
- **limitadas en el tiempo** : su ejecución tiene un límite máximo de tiempo.
- **en forma de pulso** : cuando son limitadas en el tiempo con un tiempo límite muy pequeño.

También podemos diferenciar entre acciones:

- **continuas**: las que solo dependen de la actividad o inactividad de la etapa asociada. Si no se indica lo contrario las acciones se consideran de este tipo.

- **condicionales**: las que dependen además del cumplimiento de ciertas condiciones.

- **Receptividades**

También se les llama **condiciones de transición**, pues representan la condición que debe cumplirse para que se dispare la transición y con ello se produzca la evolución entre las etapas que separa.

Las receptividades son una función lógica de las entradas al sistema, o de variables auxiliares, y de la actividad de las etapas. Se les puede asignar una variable lógica, que será igual a 1 cuando la condición de transición sea verdadera y 0 cuando sea falsa. Se representan al lado de la transición asociada mediante un texto que explique la condición de transición, una expresión booleana, un símbolo gráfico, etc.

*4.4.3.2 Reglas de evolución*

Se definen ciertas reglas que caracterizan la evolución de un Diagrama Funcional:

- Regla 1: Situación inicial.

La situación inicial se caracteriza por las etapas iniciales que, por definición, estarán activas al comienzo de la operación. En todo Diagrama Funcional debe haber al menos una etapa inicial.

- Regla 2: Disparo de transiciones.

Las transiciones pueden estar habilitadas o deshabilitadas. Una transición está habilitada si todas las etapas precedentes conectadas a ella están activas. En otro caso estará deshabilitada.

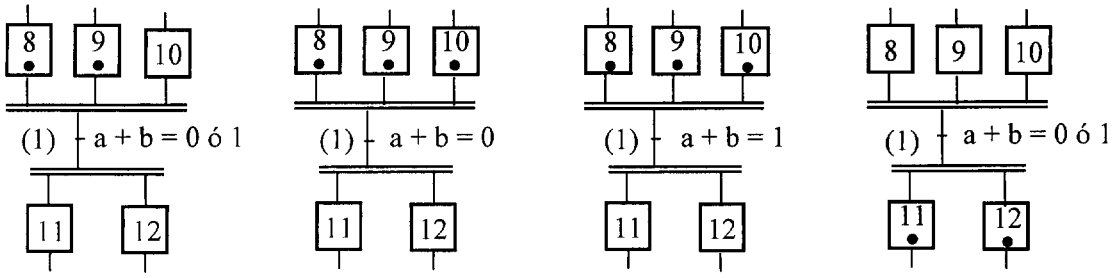
El disparo de la transición se produce cuando:

- la transición está habilitada, y
- la transición está activada, es decir, la condición de transición asociada a ella es cierta.

- Regla 3: Evolución de las etapas activas.

El disparo de la transición produce simultáneamente la activación de todas las etapas inmediatamente posteriores conectadas directamente a dicha transición, y la desactivación de todas las inmediatamente anteriores. Esto provoca la evolución de una

situación a otra en el diagrama. En la figura 4.21 se representan varios ejemplos de evolución de la actividad entre diversas etapas.




**Transición no habilitada:**

- la etapa 10 no está activa
- la condición es indiferente

**Transición habilitada pero no activada:**

- etapas anteriores activadas
- condición de transición no es verdadera

**Transición habilitada y activada:**

- etapas anteriores activadas
- condición de transición verdadera

**Disparo de la Transición:**

- evolución a la situación siguiente

**Figura 4.21** Evolución entre etapas

➢ Regla 4: Evolución simultánea.

Las transiciones que se disparan simultáneamente se representan mediante doble rayado (figura 4.22). Si dichas transiciones se encuentran separadas, por estar en distintas posiciones del esquema o en esquemas diferentes, se indica con un asterisco y una referencia adecuada.

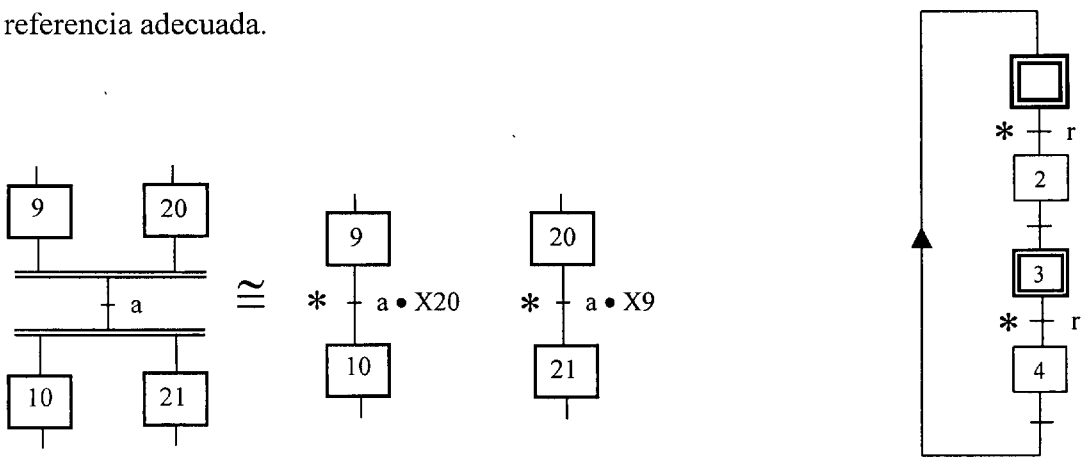
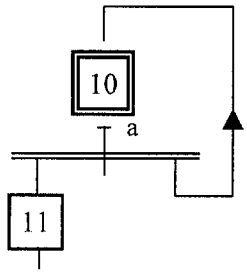



**Figura 4.22** Ejemplos de evolución simultánea

➢ Regla 5: Activación y desactivación simultáneas.

Si, durante la operación, una etapa es activada y desactivada simultáneamente, se dará prioridad a la activación (figura 4.23).




La transición 'a' desactiva la etapa 10 por ser inmediatamente anterior y la activa por ser inmediatamente posterior

**Figura 4.23** Ejemplo de transición que activa y desactiva simultáneamente una etapa

➢ Regla 6: Tiempos de disparo de transiciones y de activación de etapas.

El tiempo de disparo de una transición puede ser teóricamente considerado tan pequeño como se quiera, pero nunca cero. En la práctica vendrá impuesto por la tecnología utilizada para implementar el sistema en cuestión.

De la misma manera, el tiempo de activación de una etapa tampoco se considerará nunca nulo.

*4.4.3.4 Regla de sintaxis:* Regla de alternancia etapa - transición.

Debe respetarse siempre la alternancia de etapa-transición y transición-etapa en toda la secuencia de evolución del esquema. Nunca podrán aparecer dos etapas ni dos transiciones unidas directamente por un enlace.

*4.4.3.5 Estructuras básicas*

Todas las posibles evoluciones de un Diagrama Funcional se pueden representar mediante la combinación de las siguientes estructuras básicas.

- **Secuencia simple**

Está formada por una serie de etapas conectadas una detrás de otra, como se muestra en la figura 4.24. Cada etapa va seguida por una sola transición y cada transición es habilitada por una sola etapa.

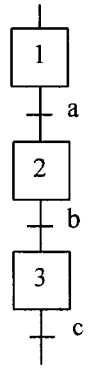



**Figura 4.24** Ejemplo de secuencia simple

- **Secuencia de selección**

Se utiliza en los casos en los que se puede elegir entre más de un camino de evolución dentro del Diagrama Funcional. Sólo se recorre el camino elegido en cada ocasión, sin tener en cuenta los demás.

➢ **Comienzo de secuencia de selección**

Se representa mediante tantas transiciones como caminos de evolución haya, situadas bajo una línea horizontal. No se permite ningún símbolo de transición por encima de dicha línea horizontal, ya que no pueden aparecer dos símbolos de transición seguidos, y cada símbolo debe representar una única posibilidad de evolución. En la figura 4.25 se muestra un ejemplo.

Para elegir una única secuencia es necesario que las condiciones de transición sean excluyentes entre sí, es decir, que no puedan ser ciertas más de una a la vez. También se puede asignar cierto orden de prioridad.

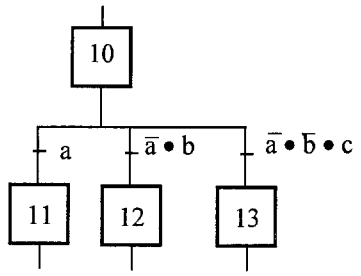



**Figura 4.25** Ejemplo de comienzo de secuencia de selección

➢ **Final de secuencia de selección**

La convergencia entre varias secuencias de selección se representa por tantos símbolos de transición como secuencias converjan, por encima de una línea horizontal. Como en el caso anterior, no se permite ningún símbolo de transición por debajo de dicha línea horizontal. En la figura 4.26 se muestra un ejemplo.

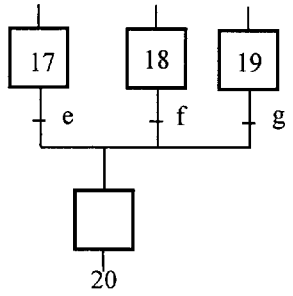



**Figura 4.26** Ejemplo de final de secuencia de selección

En éste caso, las condiciones de transición no tienen por qué ser excluyentes entre sí, ya que, como se indicó en el caso anterior, sólo se recorre una de las secuencias y la transición a disparar vendrá dada por la etapa anterior activada.

- **Secuencias simultáneas**

También se denominan secuencias en paralelo. Aparecen en los casos en los que se ejecutan varias secuencias a la vez. La activación de todas éstas secuencias es simultánea, pero tras dicha activación la evolución de cada secuencia es independiente de la evolución de las otras.

- **Comienzo de secuencias simultáneas.**

  También se denomina divergencia de secuencias simultáneas. Como aparece en la figura 4.27, se representa mediante una única transición por encima de una doble raya horizontal, que es el símbolo de sincronización (ISO 5807, símbolo nº 9.2.2.5).

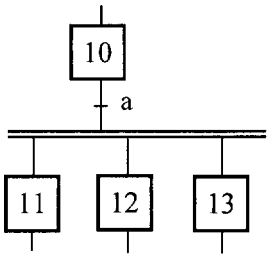



**Figura 4.27** Ejemplo de comienzo de secuencias simultáneas

- **Final de secuencias simultáneas.**

  Se utiliza para sincronizar la convergencia de varias secuencias simultáneas. Se representa mediante una única transición bajo una doble línea horizontal que, como habíamos indicado es el símbolo de sincronización, y significa que hasta que no están activas todas las etapas inmediatamente anteriores a la doble línea no se habilita la siguiente transición (figura 4.28).

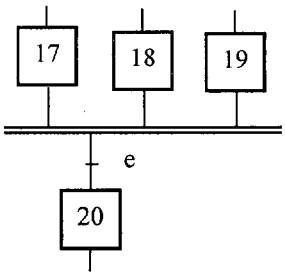



**Figura 4.28** Ejemplo de convergencia de secuencias simultáneas

Tanto la divergencia como la convergencia de secuencias simultáneas se puede realizar en varios pasos, divergiendo o convergiendo primero algunas de las secuencias y luego otras.

*4.4.3.6 Macro-representación.*

La principal ventaja de la macro-representación es que permite una aproximación progresiva a la descripción del sistema, estableciendo diferentes niveles de detalle en dicha descripción. Además, proporciona un método de representación separada que facilita la elaboración y el análisis de la documentación así como la realización de modificaciones. Hay dos métodos que utilizan el concepto de la macro-representación: las macro-etapas, y las tareas.

- **Macro-etapas**

Una macro-etapa, ME, representa una secuencia de etapas y transiciones llamada "expansión de macro-etapa". Esta expansión puede contener cualquiera de las estructuras del Diagrama Funcional. Su utilización debe respetar cuatro normas:

1.- La expansión de la ME debe contener una etapa de entrada y una de salida, identificadas mediante las letras E y S, respectivamente.

2.- El disparo de la transición anterior a una macro-etapa debe activar la etapa de entrada, E, de su expansión.

3.- La etapa de salida, S, participa en la habilitación de la transición posterior a la macro-etapa.

4.- No debe tener relación estructural con una etapa o transición de la expansión de la ME, ni con otro gráfico de la representación.

El símbolo general de una macro-etapa es similar al de una etapa normal, pero dividido en tres partes, mediante dos líneas horizontales (figura 4.29). Se pueden identificar mediante un número, como cualquier otra etapa, un número precedido de la letra M, o una etiqueta.

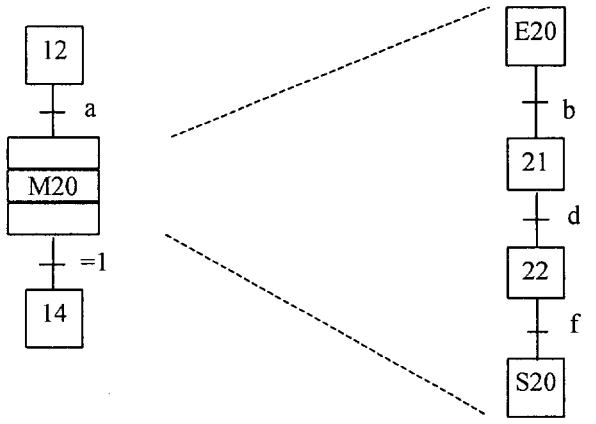



**Figura 4.29** Ejemplo de macro-etapa

Por definición, no se puede utilizar la misma expansión para dos macro-etapas distintas. Esta restricción es importante para evitar situaciones de conflicto si se activan simultáneamente varias macro-etapas que comparten una misma expansión.

- **Tarea**

El concepto de tarea, no introduce una extensión del Diagrama Funcional sino que corresponde simplemente a una representación progresiva estructurada. Permite la reutilización de secuencias en distintos puntos del diagrama.

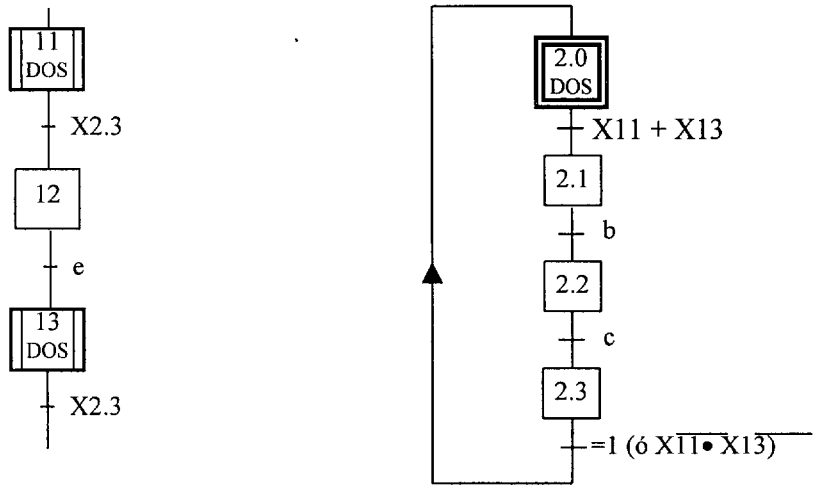



**Figura 4.30** Ejemplo de tarea

Las tareas se representan mediante un símbolo similar al de una etapa, pero dividido en tres partes mediante dos líneas verticales. Se pueden identificar mediante un número o una etiqueta, como el resto de las etapas (figura 4.30).

*4.4.3.8 Ejemplos de Diagrama Funcional*

A continuación va a mostrarse algunas secuencias que aparecen en el nivel deliberativo correspondientes tanto al secuenciador principal como los secuenciadores de las habilidades delibertivas. Las secuencias de ambos secuenciadores se van a representar mediante Diagramas Funcionales.

Un ejemplo de la secuencia del secuenciador principal del nivel deliberativo es el que aparece en la figura 4.31. Las habilidades secuenciadas en este caso son las habilidades deliberativas. Como ya se comentó, esta secuencia viene dada a priori y tiene la peculiaridad de que no pueden realizarse dos tareas deliberativas en paralelo, tal y como se explicó anteriormente.

En la etapa inicial, tras recibir la orden de realizar una misión por el usuario, se hace la planificación de los movimientos a realizar por el robot. Una vez que el planificador ha terminado, construye la lista de tareas que puede ser expresada a su vez en forma de Diagrama Funcional. Posteriormente se activa la etapa encargada de ejecutar el plan. Este plan consistirá en una secuencia de tareas a realizar. Si las tareas son realizadas correctamente, el secuenciador se quedará a la espera de recibir otra orden.

Si el robot encuentra un camino obstruido o una puerta cerrada, se volverá a activar la etapa de planificación. Por otro lado, si la incertidumbre de su posición es grande, el secuenciador activará una nueva etapa, que se encargará de relocalizar al robot. Si el robot se encuentra en una posición desde la cual puede continuar el plan, se activará la etapa correspondiente a la ejecución del plan. En el caso de que el robot se encuentre en una situación donde no pueda continuar, se volverá a llamar al planificador.

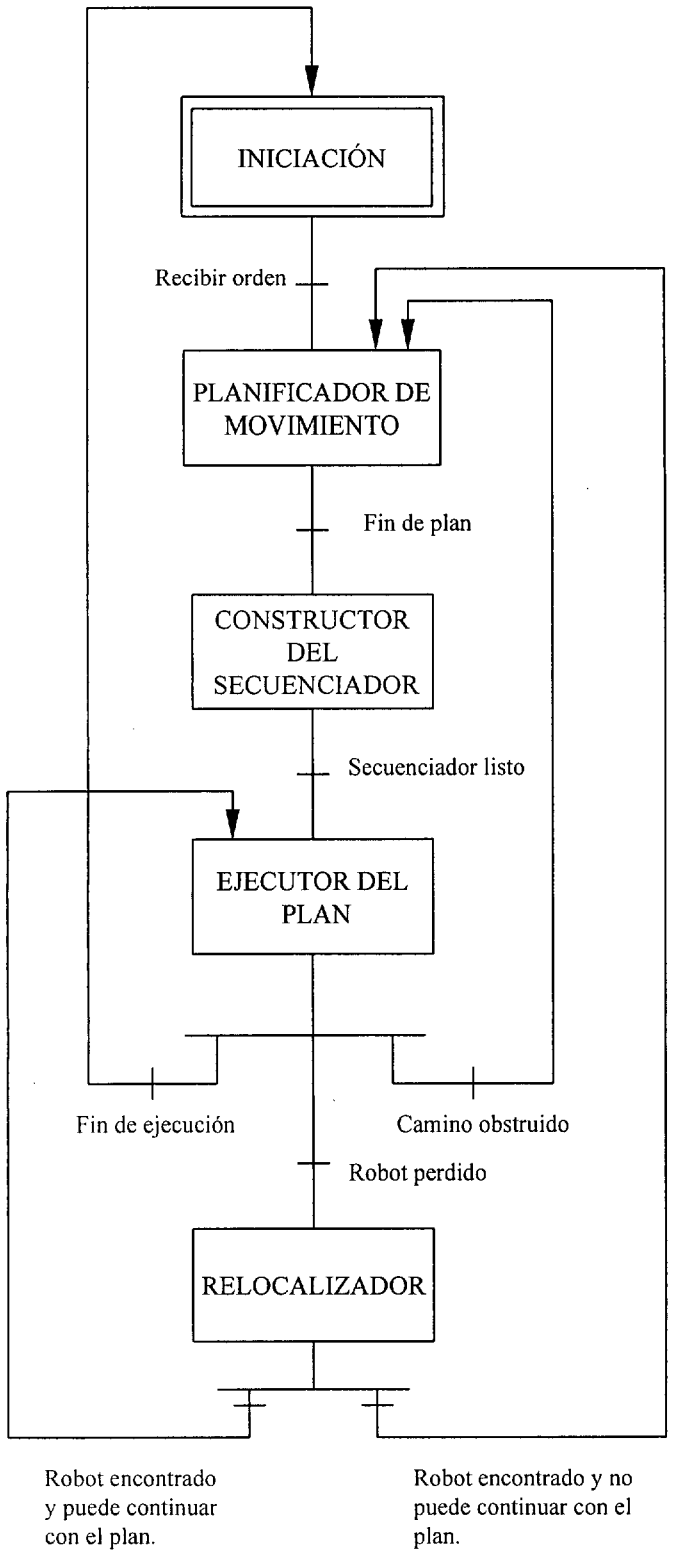



**Figura 4.31** Secuencia de la capa deliberativa. Implementación en Diagrama Funcional

El secuenciador que se encuentra dentro de las habilidades deliberativas se encarga de gestionar las habilidades del nivel automático. Dicho secuenciador contiene la secuencia de acciones a realizar por el robot. Al ejecutarse el Diagrama Funcional, se activará la habilidad automática correspondiente y se quedará en espera de que el evento correspondiente active la transición de cambio de etapa. Este secuenciador se encarga, por lo tanto, de hacer de puente entre el plan establecido y las habilidades que actúan a más bajo nivel. Un ejemplo de esta secuencia puede ser el Diagrama Funcional que se muestra en la figura 4.32.

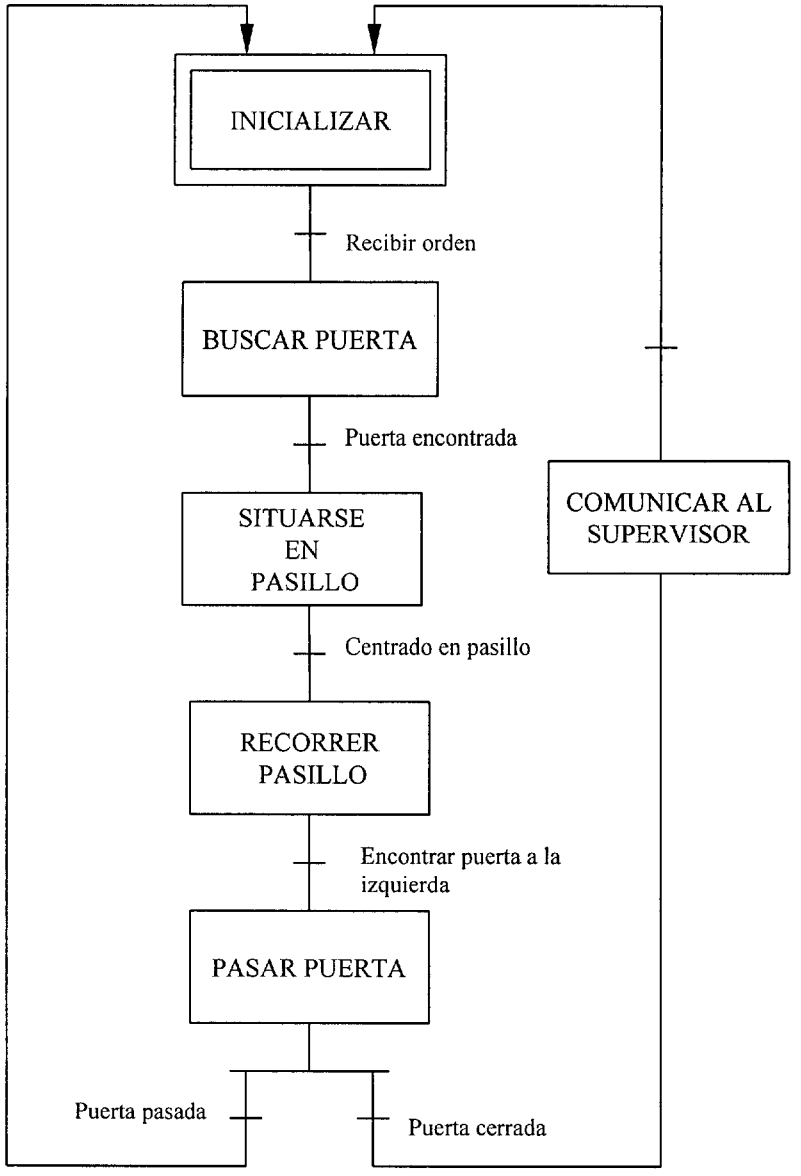
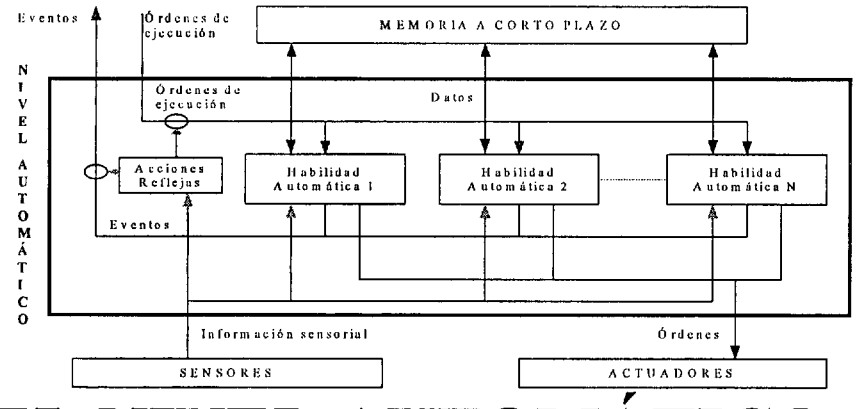



**Figura 4.32** Secuenciador de las habilidades automáticas

En él se le da la orden al robot de ir de una sala a otra. Se parte de una etapa en la que se inicializa la secuencia de tareas. Cuando recibe la orden de comenzar, se dirige a la puerta de la sala. Una vez encontrada la puerta, el robot se dirige al pasillo y se sitúa en el centro del mismo. Posteriormente empieza a recorrer el pasillo, hasta que encuentra la primera puerta a la izquierda. A continuación trata de pasar la puerta. Si la puerta está abierta, entra en la nueva sala, volviendo a quedarse a la espera de una nueva orden. Si la puerta está cerrada, lo comunica al supervisor y se queda a la espera de una nueva orden.

En este caso, el ejemplo presentado es una secuencia lineal de operaciones que se realizan una después de otra. Sin embargo, cabe destacar que el planificador puede presentar secuencias de planes con diferentes alternativas, así como se pueden realizar tareas en paralelo, ya que dos habilidades automáticas si que pueden ser ejecutas en paralelo. En el capítulo 6 se muestran otros ejemplos de secuencias de otras habilidades deliberativas.




# 5

# EL NIVEL AUTOMÁTICO

El hombre no está hecho para meditar sino para actuar
JEAN-JAQUES ROUSSEAU

¡Cuántas cosas deben ignorarse para actuar!
PAUL VALÉRY

El esfuerzo por unir sabiduría y acción se logra pocas veces y dura poco.
A. EINSTEIN

# El Nivel Automático 5

## 5.1 Introducción

El nivel automático de la presente arquitectura está relacionado con el control a bajo nivel de los elementos del robot. Las habilidades de este nivel interactuarán directamente con los sensores y actuadores del robot. Este nivel es el encargado de dotar a la arquitectura de la necesaria reactividad.

Este nivel ha sido ampliamente desarrollado en otras arquitecturas. De hecho, es el nivel principal de las llamadas arquitecturas reactivas[Brooks86][Arkin89].

En este nivel se introduce el concepto de habilidad automática. Dentro de este concepto están incluidas tanto los llamados comportamientos elementales de otros autores[Brooks86][Gachet93] como los llamados comportamientos complejos o emergentes [Gachet93]. Este concepto de comportamiento o conducta ha sido ampliamente desarrollado en la literatura de la robótica móvil. Algunos autores [Bonasso97] también utilizan el concepto de habilidad.

### 5.1.1 Conductas simples

Muchos son los autores que basan la reactividad de sus robots en la ejecución de conductas simples.

Brooks define la arquitectura de Subsumption o inclusión como una alternativa a la utilización de técnicas de IA clásica para planificación y control en robots [Brooks86]. La arquitectura de inclusión tiene las siguientes propiedades:

- se define un conjunto de comportamientos que operan asíncronamente y en paralelo sin una función central que los coordine,
- cada comportamiento incorpora las funciones de percepción, planificación y ejecución de tareas que son necesarias para alcanzar la conducta externa deseada,
- la inhibición o inclusión de las entradas y salidas de un comportamiento se realiza a través de otro comportamiento,
- los comportamientos del sistema no resultan necesariamente de estructuras internas complejas.

La base de esta arquitectura de Subsumption es que un módulo puede suprimir o inhibir las entradas (salidas) de otro por un determinado período de tiempo reemplazándolas por otras con lo que capas altas de control pueden ser integradas fácilmente con las capas previas a través de este mecanismo.

En AFREB [Gachet93], el nivel de control reactivo recibe información de niveles superiores, que consisten en el comportamiento principal, los comportamientos auxiliares y

los parámetros para el supervisor de fusión e incluso algunos parámetros de los comportamientos auxiliares.

En la arquitectura AFREB los comportamientos primitivos son homogéneos, es decir, todos ellos producen una salida del mismo tipo, en nuestro caso un valor de velocidad lineal $v$ y un valor de curvatura $k$.

Cada uno de los comportamientos primitivos tiene acceso directo al sistema sensorial, y tomará de él la información que necesite. Son estos comportamientos los que hacen que el sistema se comporte de forma reactiva ante el entorno. La salida de cada comportamiento en un instante de tiempo t+1 depende exclusivamente de la información de entrada en el instante t, por lo que no tienen memoria. Además son completamente independientes entre sí.

En esta arquitectura, las habilidades automáticas vienen dadas en forma de conductas o comportamientos elementales. El comportamiento final o emergente, se realiza por fusión mediante superposición de las conductas elementales. Al final, lo que generan es una habilidad que se encarga de recorrer un camino dado en forma de puntos geométricos que es capaz de evitar obstáculos de forma reactiva.

Gracias a la modularidad del sistema se pueden utilizar los comportamientos primitivos necesarios en cada aplicación concreta, aunque muchos de ellos son comunes a casi todas las aplicaciones. En nuestro caso utilizamos los siguientes:

*Atracción a un punto*: Este comportamiento se limita a dirigir el robot en línea recta hacia el punto prefijado sin preocuparse de si hay o no obstáculos en el camino. Sólo necesita conocer las coordenadas del punto de destino y la velocidad máxima que puede alcanzar y no utiliza en absoluto la información de los sensores de ultrasonidos.

*Seguimiento de un contorno*: Este comportamiento hace que el robot siga el contorno de un obstáculo inmóvil a una distancia prefijada del mismo. En realidad se implementan dos comportamientos distintos, uno sigue el contorno a la izquierda del robot

y el otro a su derecha. La velocidad se controla de forma que el movimiento sea estable, manteniéndose siempre entre dos valores prefijados.

*Alejarse de objetos cercanos*: Este comportamiento devuelve los parámetros adecuados para que el robot se mueva en la dirección opuesta al objeto más cercano, que se determina por el sensor que dé la mínima distancia. Sólo tiene sentido cuando existen objetos cercanos al robot.

*Búsqueda de zonas libres*: Este comportamiento dirige al robot hacia la zona donde los sensores de ultrasonidos detectan mayores distancias, es decir donde no hay obstáculos o se encuentran más alejados.

### 5.1.2 Conductas complejas

Arkin[Arkin98] explica el concepto de comportamiento complejo o emergente. Entre las diversas definiciones se encuentran:

- Un comportamiento emergente es la aparición de nuevas propiedades en sistemas complejos[Moravec88].
- Son los que provienen de la interacción en paralelo de comportamientos locales[Steels90].
- Son los que proceden de la interacción de los componentes de un sistema[Brooks91b]
- Aparecen como la interacción entre componentes no diseñados por sí mismos con una función en particular y concreta [McFarl93].

De estas definiciones se seduce que la emergencia es una propiedad basada en una colección de componentes o comportamientos interactivos. La generación de comportamientos emergentes es necesaria en cuanto no conocemos completamente el entorno y no podemos seleccionar correctamente los comportamientos simples.

Hay dos clases predominantes de coordinación: competitiva y cooperativa.

5.1.2.1 Métodos competitivos

Los conflictos surgen cuando dos o más comportamientos están activos. Los métodos competitivos proveen un medio de coordinación de las respuestas de los comportamientos para resolver el conflicto.

El coordinador puede ser visto como un elemento externo que selecciona el comportamiento ganador. La respuesta simple del comportamiento ganador anula todas las otras y es la que rige la acción del robot. Este tipo de estrategia competitiva puede ser realizado de diferentes formas:

- La *arbitración* requiere que la función de coordinación pueda tomar la forma de una red fija de priorización en la cual existe una dominancia jerárquica estricta de comportamientos, normalmente implementada haciendo uso de supresiones e inhibiciones de manera descendiente. Esto es la base de la arquitectura de Subsumption [Brooks86]. En la figura 5.1 se muestra un ejemplo en el que aparece la jerarquía de comportamientos.

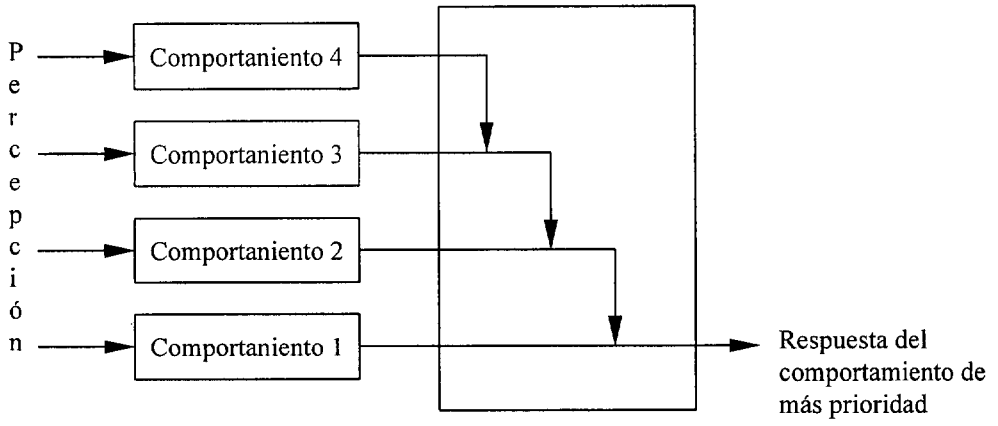



**Figura 5.1** Coordinación basada en prioridades

- Los métodos *acción-selección*[Maes90] seleccionan arbitrariamente la salida de un comportamiento simple, pero es hecho de una manera autocrática. Aquí, la actividad de cada uno de los comportamientos compite con las demás. No se establece ninguna jerarquía fija: más bien los comportamientos compiten con

los demás por el control en un instante dado. La arbitración ocurre por la selección del comportamiento más activo, pero no se presenta ninguna jerarquía predefinida. La figura 5.2 muestra este caso.

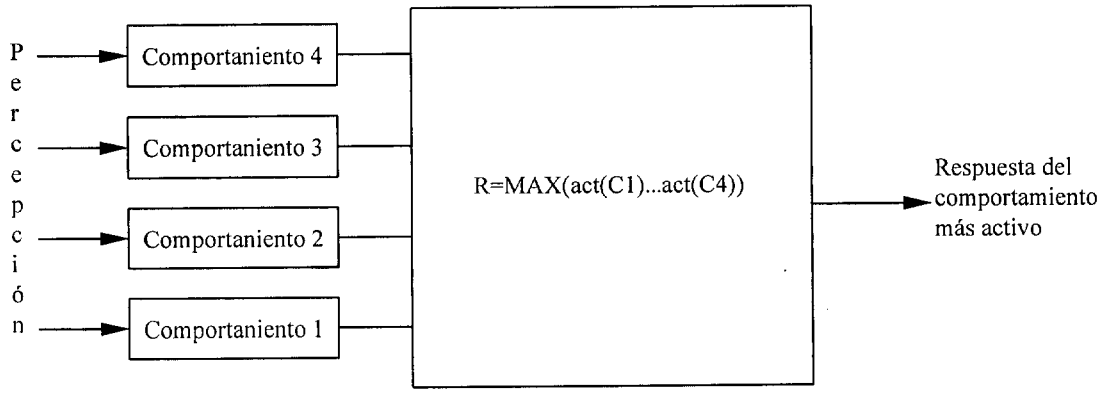



Figura 5.2 **Coordinación basada en acción-selección**

Existen otros métodos competitivos más democráticos en los cuales los comportamientos generan votos sobre las acciones. El comportamiento que tenga más votos será escogido para llevar a cabo esa acción[Rosen95].

5.1.2.1 Métodos cooperativos

Los métodos cooperativos aparecen como una alternativa a los métodos competitivos. La fusión de comportamientos permite la habilidad de usar concurrentemente la salida de más de un comportamiento a la vez. Esta coordinación de comportamientos puede hacerse de varias formas:

- La *fusión de comportamientos* es una de ellas. En ella se intenta ponderar las acciones de cada una de los comportamientos simples con una ganancia para cada conducta. En la figura 5.3 se muestra in esquema de fusión de comportamientos.

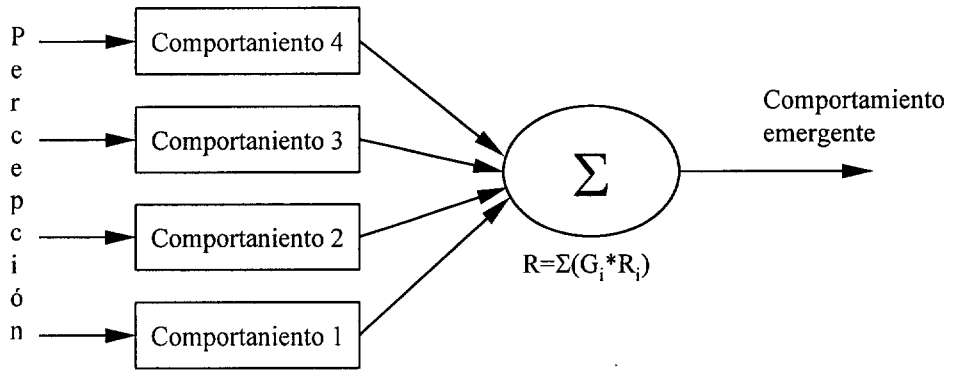



**Figura 5.3** Fusión de comportamientos

Una forma de representar las salidas de los comportamientos es mediante los campos de potencial. Se basan en la adición o superposición de vectores. La ganancia de cada conducta es usada como un multiplicador de los vectores de movimiento correspondientes a cada conducta. En las figuras 5.4 y 5.5 aparecen dos ejemplos utilizando campos de potencial. En la primera de ellas el peso multiplicador de la conducta atracción al objetivo es mayor que el de evitar obstáculos. En la segunda, el peso multiplicador de evitar obstáculos es el mayor. En ambos casos consigue llegar al objetivo, pero sin embargo en el segundo caso pasa más lejos de los obstáculos.

Un ejemplo de fusión de comportamientos es la arquitectura AFREB presentada por Gachet [Gachet93]. En la arquitectura AFREB, se plantea generar un comportamiento complejo a partir de los comportamientos primitivos anteriores mediante el llamado supervisor de fusión. El módulo de comportamientos primitivos contiene todos los comportamientos elementales necesarios para llevar a cabo una tarea. El módulo supervisor de fusión obtiene comportamientos más complejos mediante la correcta combinación de comportamientos elementales. El módulo ejecutor transforma el vector de salida obtenido en el supervisor de fusión a comandos adecuados para el sistema a controlar.

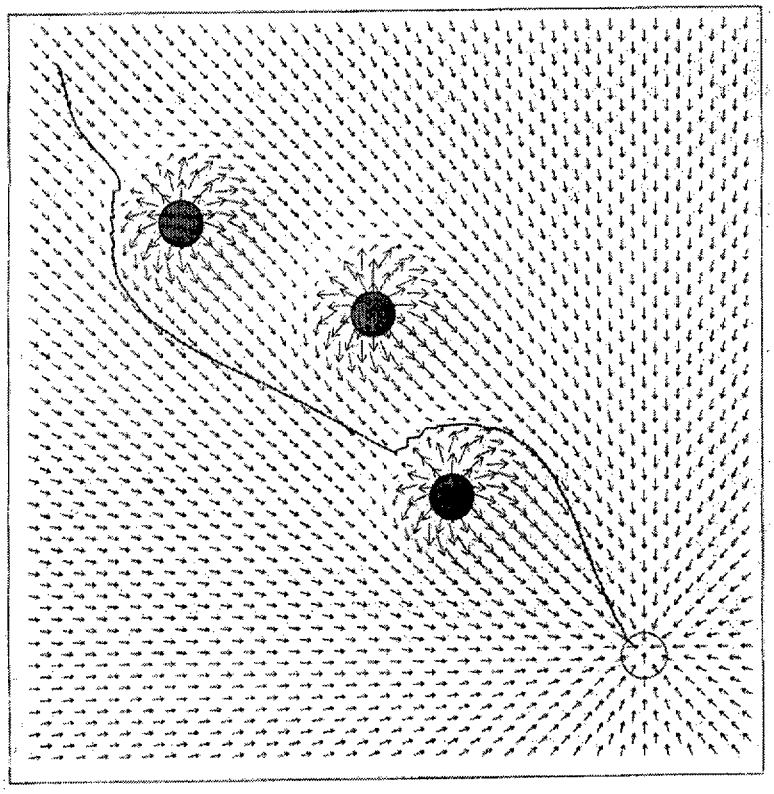


**Figura 5.4** Fusión de conductas. Predominio de la conducta atracción al objetivo

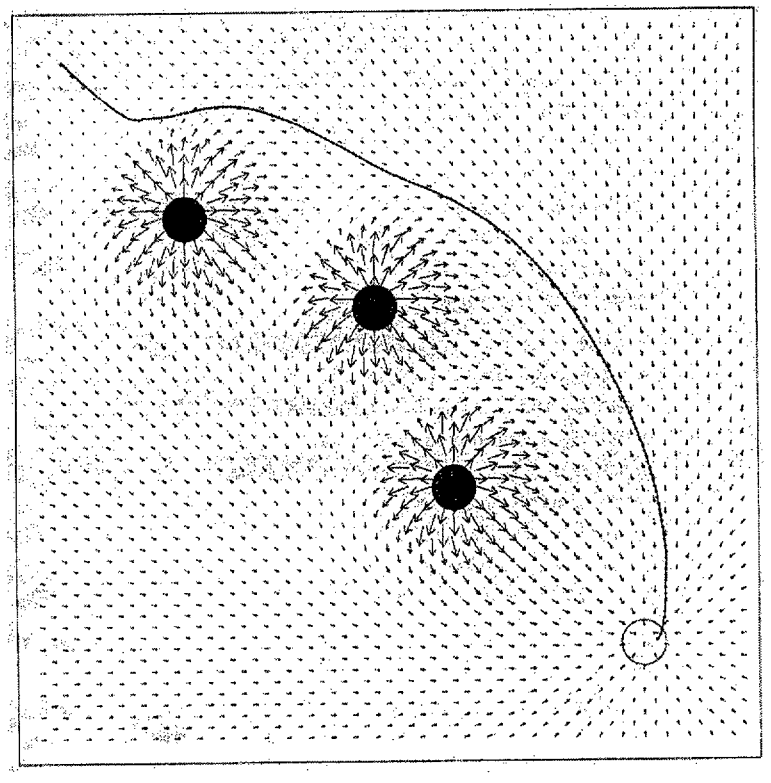


**Figura 5.5** Fusión de conductas. Predominio de la conducta "evitar obstáculos"

La tarea que debe realizar el supervisor de fusión es conseguir comportamientos complejos o emergentes a partir de la fusión de las respuestas dadas por los comportamientos primitivos. La fusión la realiza el supervisor dando como valores de salida la media ponderada, según los pesos que el mismo calcula, de las salidas de cada uno de los comportamientos primitivos. Para el cálculo de los pesos se han empleado distintas técnicas como redes neuronales y lógica difusa.

En los trabajos de Payton [Payton92] se utiliza una técnica de fusión basada en establecer restricciones en las variables de control de tal forma que el sistema de control debe operar de manera que se satisfagan los requerimientos de los comportamientos.

Saffiotti[Saffi95] utiliza métodos formales para mezclar comportamientos. En su trabajo, a cada comportamiento se le asigna una función de "deseo" que pueden ser combinadas usando operadores lógicos especiales llamados "t-norms", que son una forma de lógica multievaluada. Una acción simple es escogida de un conjunto de acciones creadas por mezcla de las ponderaciones de las acciones de los comportamientos primitivos.

- La *secuenciación de comportamientos* es otra de las formas de coordinar comportamientos. En ella los comportamientos son coordinados por un secuenciador que establece las dependencias temporales entre cada uno de ellos. El secuenciador se encarga de escoger cada uno de los comportamientos y de decir cuanto tiempo debe estar activo. En la figura 5.6 se muestra este método de coordinación de comportamientos.

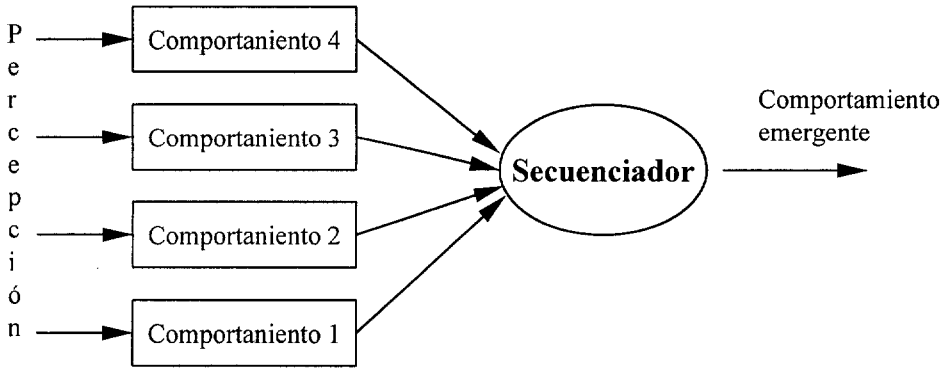



Figura 5.6 **Secuenciación de comportamientos**

### 5.1.3 Habilidades

Peter Bonasso[Bonasso97] utiliza el concepto de habilidad en su arquitectura 3T. Para Bonasso las habilidades representan la conexión con el entorno. El sistema de control en la práctica se crea estableciendo un grupo de habilidades que trabajan juntas en un contexto dado. Es necesario dadas las características de las habilidades crear una interfaz con el secuenciador, que sea independiente de las características físicas del robot y de sus sensores. Esta representación incluye:

1. Unas espedificaciones sobre las entradas y salidas de esas habilidades. Cada habilidad debe tener una descripción de las entradas que espera y una lista de las salidas que genera. Esto permite que las tareas aparezcan en una red, de tal forma que las salidas de una habilidad pueden ser tomadas como entradas de otra, de manera inmediata.

2. Una transformación computacional. Este es el punto en que la habilidad hace su trabajo. Cuando se establece una habilidad, utiliza su transformación para computar de nuevo sus salidas basándose en sus nuevas entradas.

3. Una rutina de inicialización. A cada habilidad se le da la posibilidad de iniciarse por si misma una vez que el sistema se pone en marcha.

4. Una función de habilitación. El secuenciador puede habilitar y deshabilitar habilidades. Dependiendo del contexto, a la habilidad se le da la oportunidad de variar comenzando con un procedimiento especial.

5. Una función de deshabilitación Cuando no se necesita una habilidad, el secuenciador la deshabilitará y la función deshabilitada realizará todos los cambios necesarios.

## 5.2 Nivel Automático

Este nivel es el nivel donde aparece la reactividad del sistema. En él se encuentran las habilidades automáticas que pueden ser ejecutadas o activadas por el nivel deliberativo. Estas habilidades reciben directamente información del sistema sensorial y actúan directamente sobre el sistema locomotor del robot.

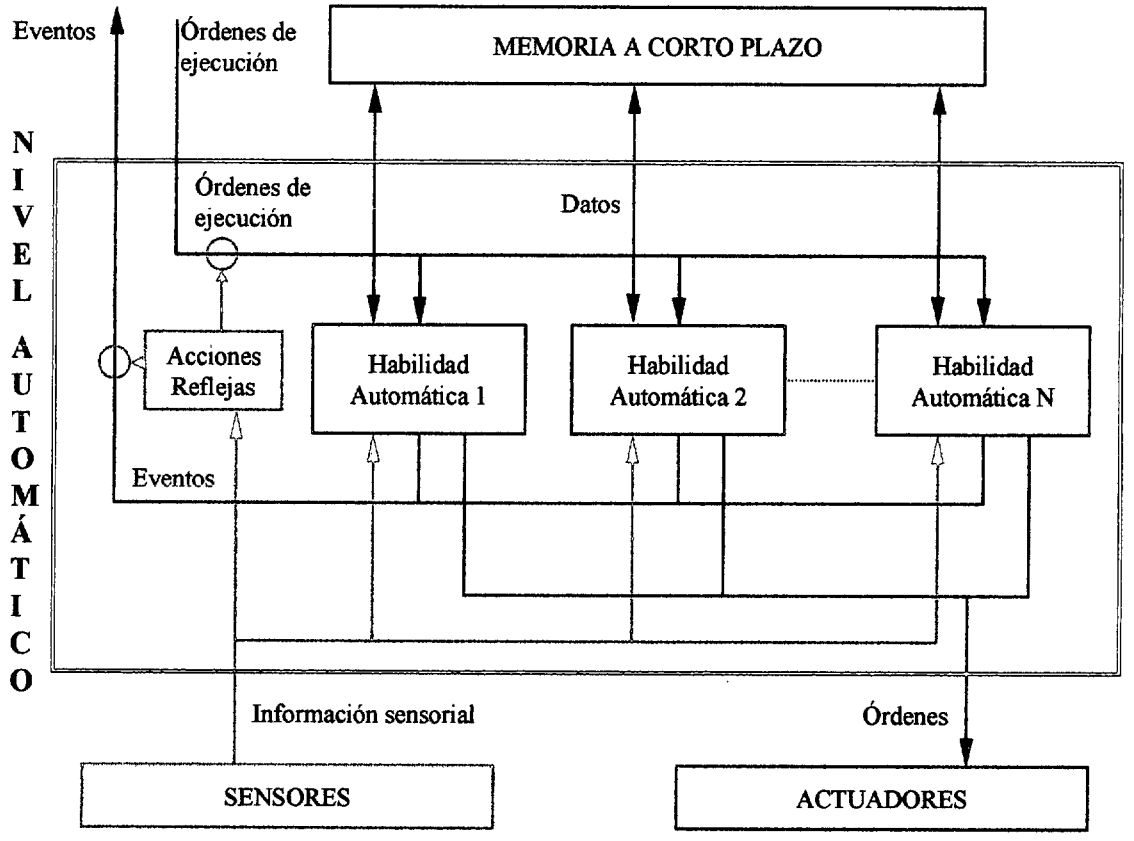



Figura 5.7 **Nivel Automático**

En este nivel se encontrarán los módulos de control a bajo nivel, que actúan directamente sobre los accionadores, así como los módulos que recogen datos de los distintos sensores del sistema. El esquema de este nivel se corresponde con la figura 5.7. En él se aprecian los distintos elementos que forman el nivel automático:

**Habilidades automáticas:** Son las capacidades tanto sensoriales como motoras de que dispone el sistema. Son activadas mediante los secuenciadores de las habilidades del nivel deliberativo al que le devuelve información en forma de datos y eventos. Estas habilidades toman información directamente de los sensores y ejecutan órdenes directamente sobre los actuadores, ya sean al sistema de locomoción del robot o sobre los actuadores que manejan los dispositivos sensoriales y sus elementos auxiliares.

**Acciones reflejas:** Las acciones reflejas proceden directamente de los sensores y consisten en información de carácter prioritario. Dichas acciones son tratadas a modo de interrupciones. Producen una señal que deshabilita las órdenes del nivel deliberativo, teniendo un tratamiento prioritario. Un ejemplo de estas acciones reflejas puede ser el camarero que lleva una bandeja y es picado por un mosquito. Aunque su nivel deliberativo le comunique que lleve la bandeja recta, el picotazo del mosquito genera una interrupción prioritaria que hace que mueva el brazo, contraponiéndose a su nivel deliberativo, con riesgo de que se le caiga la bandeja. La interrupción llega a modo de eventos a las habilidades y al secuenciador del nivel deliberativo y a modo de órdenes de ejecución a las habilidades del nivel automático. Dicha interrupción deshabilita momentáneamente las órdenes del nivel deliberativo y será tratada por un módulo que indique que hacer en el caso de que llegue la interrupción.

El nivel automático se comunica con el nivel deliberativo o reflexivo mediante las dos líneas antes mencionadas y la memoria a corto plazo Por la primera de ellas el nivel automático recibe las órdenes de ejecución de las habilidades del nivel deliberativo. Por la segunda de ellas, envía los eventos generados por las habilidades automáticas. A través de la memoria a corto plazo, se produce el intercambio de datos entre las habilidades del nivel automático y las habilidades deliberativas, como pueden ser parámetros de control.

## 5.3 Habilidades automáticas

Son las capacidades tanto sensoriales como motoras de que dispone el sistema. Estas habilidades toman información directamente de los sensores y ejecutan ordenes directamente sobre los actuadores, ya sean al sistema de locomoción del robot o sobre los actuadores que manejan los dispositivos sensoriales y sus elementos auxiliares. Bajo este concepto se puede englobar los comportamientos elementales y complejos contemplados por otros autores. Una posible clasificación de estas habilidades puede ser en habilidades motoras, sensoriales y sensimotoras.

### 5.3.1 Tipos de habilidades automáticas

Un primer intento de clasificación se puede hacer teniendo en cuenta cómo interactúan con los sensores y los actuadores. Una posible forma de clasificar estas habilidades puede ser en habilidades motoras, sensoriales y sensimotoras

#### *5.3.1.1 Habilidades motoras*

Son habilidades que directamente reciben la orden de ejecutarse y actúan directamente sobre los actuadores. No reciben información sensorial, con lo que se corresponden con actuadores que trabajan en cadena abierta. Es muy difícil encontrar habilidades motoras puras ya que normalmente existe realimentación de los sensores, aunque esta información provenga de los sensores internos, como pueden ser los encoders que proporcionan la odometría. Dicha información propioceptiva hace que el bucle de control se cierre. Su esquema se correspondería con el de la figura 5.8.

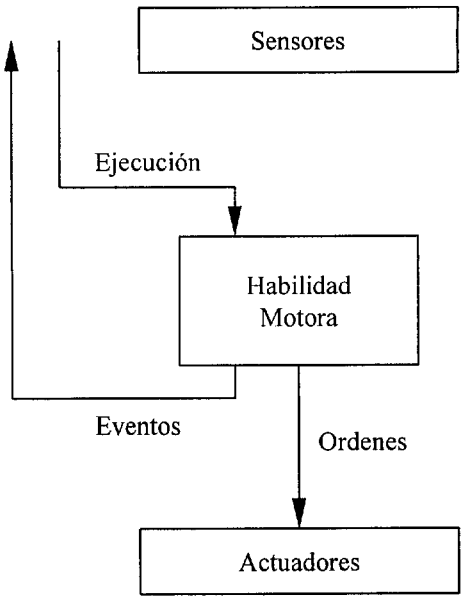



**Figura 5.8** Habilidad motora

También cabe aclarar que en el caso de oclusión sensorial, al perder la información de los sensores, podemos vernos obligados a trabajar en cadena abierta durante un determinado tiempo.

Un ejemplo de esta habilidad puede ser ordenar que avance el robot en una dirección durante un cierto tiempo.

*5.3.1.2 Habilidades sensoriales*

Son habilidades que se limitan a recibir información sensorial y enviarla a las habilidades del nivel deliberativo, donde puede ser procesada o utilizada por las distintas habilidades.

Esta habilidad se corresponde con los sensores pasivos, es decir, aquellos que se limitan a hacer lecturas y no necesitan de acción motora alguna para recibir datos.

Ejemplos de estos sensores pueden ser los sensores de ultrasonidos e infrarrojos o un sistema de telemetría láser. En la figura 5.9 aparece dicha habilidad.

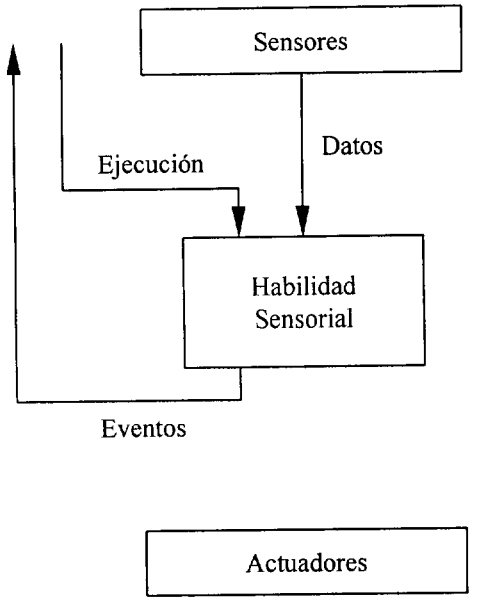



**Figura 5.9** Habilidad sensorial

*5.3.1.3 Habilidades sensimotoras*

Son aquellas habilidades que reciben información de los sensores y producen acciones sobre los actuadores. Pueden ser habilidades relacionadas con el movimiento del robot que cierren su bucle de control con la información sensorial (por ejemplo, una habilidad que siga una pared debe mover al robot y medir la distancia con respecto a la pared) o bien, habilidades relacionadas con la información sensorial que necesiten accionarse mediante algún dispositivo auxiliar. El esquema correspondiente con esta habilidad es el de la figura 5.10.

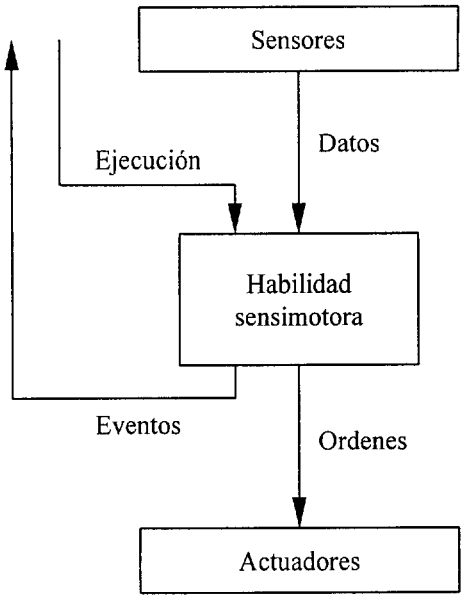



**Figura 5.10** Habilidad sensimotora

Estas habilidades son las más comunes, ya que normalmente las habilidades motoras cierran su bucle de control mediante la información sensorial. Por otro lado, las habilidades sensoriales muchas veces requieren acciones motoras para su activación. Así, por ejemplo una óptica motorizada requiere orientar la torreta y ajustar el zoom para poder captar una imagen adecuadamente.

### 5.3.2 Ejemplos de habilidades automáticas

Dentro de las habilidades automáticas, deberemos distinguir entre habilidades predominantemente sensoriales y habilidades predominantemente motoras. Dentro del primer grupo de habilidades, encontraremos las habilidades sensoriales puras y las habilidades sensimotoras que van encaminadas a proporcinar información sensorial. En el segundo grupo se incluirán las habilidades motoras puras y aquellas sensimotoras cuyo objetivo principal es la de hacer que el robot se mueva.

*5.3.2.1 Habilidades predominantemente sensoriales.*

Las habilidades sensoriales estarán fuertemente vinculadas con los elementos de percepción. A la hora de plantear las condiciones que deben cumplir las habilidades sensoriales, un modelo a seguir es el formado por las habilidades humanas. En muchos casos, imitar el sistema sensorial humano se plantea como un objetivo muy lejano, como por ejemplo en la capacidad humana de extraer información de una imagen. En otros casos, la información de algunos sensores artificiales supera la capacidad humana, como por ejemplo la precisión midiendo distancias de los sistemas de telemetría láser. Las especificaciones de diseño iniciales que se plantean se listan a continuación:

- El sistema debe ofrecer al usuario una biblioteca de habilidades sensoriales lo más completa posible.

- La cantidad y variedad de entidades que el sistema sensorial pueda manejar, no debe estar limitada a priori.

- La información proporcionada por estas habilidades puede ser empleada en diversas tareas: navegación, manipulación, inspección, etc.

- El sistema debe ser lo más independiente posible del hardware.

- Los sistemas sensoriales deben de trabajar cooperando entre sí, siendo capaz de funcionar independientemente.

Dentro de este tipo de habilidades sensoriales, podremos encontrar habilidades más simples y más complejas. Como habilidades sensoriales elementales se pueden distinguir:

- **Reconocimiento de una entidad**: Se entiende por entidad todo aquello susceptible de ser reconocido: un patrón, una marca, un objeto, una escena, un lugar o un determinado tipo de textura entre otros.

Un ejemplo concreto puede ser la detección de elementos naturales del entorno como paredes, esquinas, pasillos o puertas en entornos interiores. Para llevar a cabo dicha habilidad puede emplearse un modelo para cada entidad con el cual contrastar la información procedente de los sensores. La pared puede ser modelada mediante una línea recta, la esquina como dos rectas que se cruzan y el pasillo como dos rectas paralelas. Las rectas pueden proceder bien del tratamiento de los datos procedentes de sensores de ultrasonidos, infrarrojos o telemetría láser entre otros, o bien proceder de la información directa proporcionada por un sistema de visión artificial mediante una cámara.

En entornos exteriores estructurados como pueden ser las carreteras se pueden encontrar elementos naturales del tipo camino o cruce de caminos que se pueden modelar de forma análoga a los utilizados en los entornos interiores mediante líneas paralelas e intersecciones de líneas.

Otro ejemplo puede ser la detección de marcas artificiales como los círculos verdes utilizados en [Armingol97], carteles indicativos en edificios o señales de tráfico e indicaciones de dirección en los entornos exteriores estructurados comentados anteriormente.

- **Tracking de una entidad**: Se trata de mantener seguimiento continuo de una entidad detectada, incluso frente a oclusiones. Para ello es necesario la interacción con un sistema motriz (brazo mecánico, torreta, plataforma móvil).

- **Extracción de datos**: Una vez identificada una entidad, se trata de calcular los datos sobre ella que sean requeridos por los niveles superiores: distancia a la entidad, dimensiones, etc.

Un ejemplo concreto puede ser la habilidad que se encargue de establecer la distancia a una pared, a una esquina, a las paredes de un pasillo o a una puerta en entornos interiores. Para obtener la distancia, se debe haber detectado antes las entidades correspondientes e implementar los algoritmos matemáticos

necesarios para obtener la distancia a las rectas consideradas o al vértice de la esquina.

- **Interpretación de escenas**: La interpretación de escenas consiste en identificar los objetos presentes en la misma y, a partir de las relaciones entre ellos, obtener cierta información de alto nivel sobre la misma, generalmente referente al tipo de entorno.

- **Segmentación de una imagen**: Esta habilidad tiene por objeto dar al usuario una versión de la imagen con los píxeles agrupados por la entidad a que pertenecen. La segmentación puede ser tratada a dos niveles: una primera segmentación no supervisada que se considera como habilidad básica y una segunda segmentación que se va corrigiendo de forma dinámica junto al reconocimiento de entidades, ya que ambas tareas están en realidad muy estrechamente ligadas.

*5.3.2.2 Habilidades predominantemente motoras.*

Las habilidades motoras o sensimotoras marcan la capacidad del robot para moverse. Dichas habilidades podrán estar definidas por una serie de comportamientos básicos o por otros comportamientos más complejos.

Un ejemplo de habilidades motoras puede ser las conductas simples o comportamientos básicos utilizados por otros autores [Brooks86][Gachet93]:

- **Atracción a un punto**: Este comportamiento se limita a dirigir el robot en línea recta hacia el punto prefijado sin preocuparse de si hay o no obstáculos en el camino. Sólo necesita conocer las coordenadas del punto de destino y la velocidad máxima que puede alcanzar y no utiliza en absoluto la información de los sensores de ultrasonidos.

- **Seguimiento de un contorno**: Este comportamiento hace que el robot siga el contorno de un obstáculo inmóvil a una distancia prefijada del mismo. En realidad se implementan dos comportamientos distintos, uno sigue el contorno a la izquierda del robot y el otro a su derecha. La velocidad se controla de forma que el movimiento sea estable, manteniéndose siempre entre dos valores prefijados.

- **Alejarse de objetos cercanos**: Este comportamiento devuelve los parámetros adecuados para que el robot se mueva en la dirección opuesta al objeto más cercano, que se determina por el sensor que de la mínima distancia. Sólo tiene sentido cuando existen objetos cercanos al robot.

- **Búsqueda de zonas libres**: Este comportamiento dirige al robot hacia la zona donde los sensores de ultrasonidos detectan mayores distancias, es decir donde no hay obstáculos o se encuentran más alejados.

Otras habilidades motoras pueden ser acciones más complejas que las anteriores como por ejemplo:

- **Seguimiento de una entidad**: Esta habilidad hace uso de la habilidad sensorial tracking a una entidad. Debe ser capaz de moverse hacia la entidad indicada, detectada previamente.

- **Seguimiento de líneas de fuga**: Esta habilidad debe ser capaz de dirigir al robot a lo largo de dos líneas que confluyen en un punto.

- **Paso de puerta**: Esta habilidad se encargará de resolver el problema de pasar una puerta. Esta habilidad puede estar compuesta por otras habilidades más simples como la de buscar zonas libres, o hacer uso de habilidades sensoriales como la habilidad de detectar puerta descrita anteriormente.

- **Seguimiento de un pasillo**: Dicha habilidad se encargará de guiar al robot por el centro del pasillo. Puede estar compuesta a su vez de otras habilidades como detección de pasillos o seguimiento de líneas de fuga.

A su vez, podrán formar parte de las habilidades motoras, otros comportamientos más complejos formados a partir de los anteriores, como es el caso de seguir un camino o el caso más concreto de realizar un determinado trayecto.

## *5.4 Acciones reflejas*

Como ya ha sido comentado en párrafos anteriores, las acciones reflejas proceden directamente de los sensores y consisten en actividades de carácter prioritario.

Existen determinadas señales provenientes del sistema sensorial que deben ser respondidas inmediatamente, bloqueando todas las acciones que se estuvieran realizando. Dichas acciones son tratadas a modo de interrupciones. Producen una señal que deshabilita las ordenes del nivel deliberativo, teniendo un tratamiento prioritario. En el ejemplo comentado anteriormente del camarero que lleva una bandeja y es picado por un mosquito, la señal que produce la picadura es atendida inmediatamente, anteponiéndose al nivel deliberativo que le comunica que lleve la bandeja recta. El picotazo del mosquito genera una interrupción prioritaria que hace que mueva el brazo, contraponiéndose a su nivel deliberativo, con riesgo de que se le caiga la bandeja.

La interrupción llega a modo de evento al nivel deliberativo, cuyo secuenciador será deshabilitado momentáneamente, y disparará las acciones reflejas del nivel automático. Dichas acciones reflejas activarán con carácter prioritario las habilidades automáticas que deban actuar inmediatamente. Así, en el ejemplo del camarero, las habilidades automáticas encargadas de mover el brazo, serán habilitadas inmediatamente, teniendo como consecuencia la posible caída de la bandeja.

## 5.5 Generación recursiva de habilidades automáticas

Las habilidades automáticas motoras o sensimotoras, pueden combinarse para formar otras habilidades más complejas. Así por ejemplo, a partir de las habilidades recorrer pasillo, pasar puerta y dirigirse a un lugar, se puede obtener una habilidad que sea *ir a una habitación* en concreto de un edificio, formada por todas las anteriores. Dicha habilidad compuesta podría estar formada por un secuenciador que coordinaría las restantes habilidades para ser capaz de generar una habilidad más compleja, como se muestra en la figura 5.11.

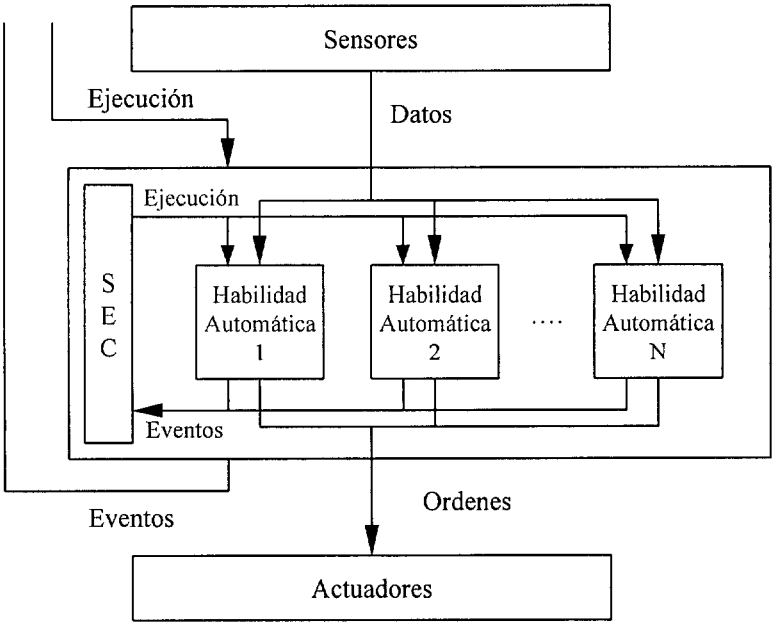



**Figura 5.11** Habilidad formada por secuenciamiento de otras habilidades

En esta figura se puede apreciar como podemos construir habilidades formadas por otras habilidades, que a su vez estén formadas por otras habilidades, apareciendo la idea de recursividad. Por ejemplo, para indicarle a otra persona como salir de un edificio, podemos darle las indicaciones mediante muchos pasos sencillos o pocos pasos y más complejos. Podemos indicarle que llegue hasta un pasillo, que lo recorra, que tome un ascensor, y continuar así hasta que llegue a la puerta, o bien decirle que llegue a conserjería y salga por la puerta mas cercana. De la primera forma, le damos más instrucciones más sencillas. De

la segunda manera le damos solo dos instrucciones, pero más complejas. En este segundo caso, las instrucciones detalladas no se le dan, puesto que las tiene ya automatizadas en una habilidad compuesta por ellas llamada *ir a portería.*

Este ejemplo nos lleva a que pueden ir surgiendo habilidades nuevas a medida que vamos automatizando secuencias que se repiten. Es decir, se pueden aprender habilidades nuevas a partir de las anteriores. Los temas de aprendizaje de habilidades serán tratados más adelante.

Otra forma de combinar habilidades es mediante el paso de información. Por ejemplo, una habilidad como *pasar una puerta*, puede recibir información de otra habilidad que sea *detectar puerta*. El esquema sería el representado en la figura 5.12.

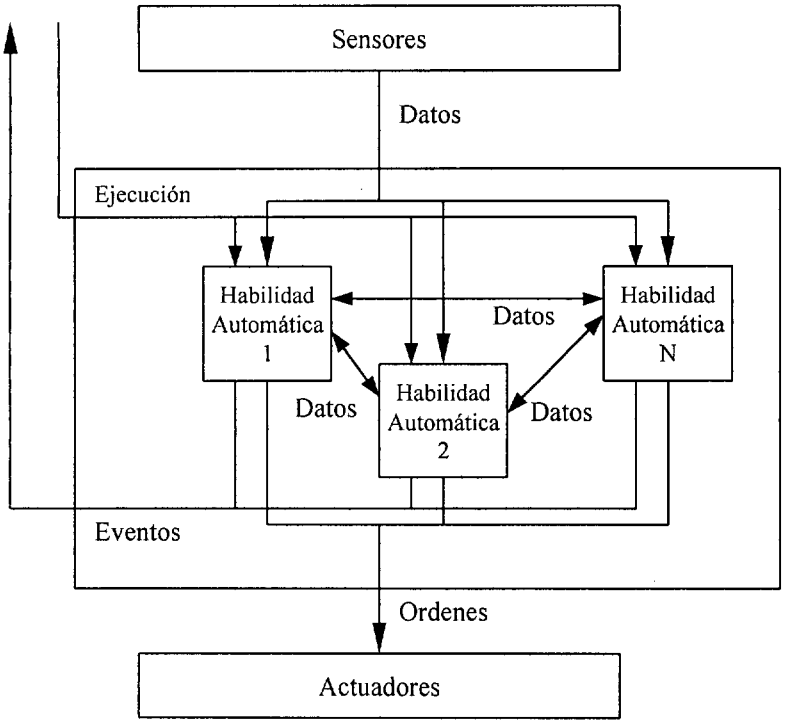



**Figura 5.12** Habilidad formada por paso de información entre habilidades

Otro ejemplo de generación de habilidades automáticas puede ser la habilidad *ir a un punto de forma reactiva*, formada por los comportamientos elementales clásicos como el seguimiento de contorno, atracción a un punto, ir a zonas libres, etc.

## 5.6 Aprendizaje de habilidades Automáticas

Un factor importante a tener en cuenta es que las habilidades pueden ser generadas por aprendizaje a partir de otras más simples. Así, dentro de las habilidades sensoriales, puede aprenderse una nueva entidad mediante adiestramiento del sistema. Este debe emplear los datos suministrados por el usuario para aprender nuevas entidades. De forma independiente, el sistema aprenderá también actualizando su conocimiento con nuevos encuentros. Si el sistema detecta una entidad desconocida de cierta relevancia, deberá solicitar a su tutor humano si debe ser aprendida o descartada.

Este aprendizaje de entidades puede ser considerado a tres niveles:

- Actualización de la base de conocimiento de las entidades aprendidas con información obtenida de cada nuevo encuentro.

- Aprendizaje de una nueva entidad sugerida directamente por el tutor, siendo éste el que provee imágenes de ejemplo y ayuda en mayor o menor medida, a construir la base de conocimiento.

- Almacenaje de información sobre entidades. En una fase avanzada, el sistema debería ser capaz de almacenar información sobre entidades no reconocidas y sugerir al tutor humano si debe aprenderlas. Este aprendizaje podría entonces hacerse a partir de los ejemplos encontrados y con más ejemplos aportados por el tutor. En una última fase, el sistema podría prescindir del tutor humano salvo para dar etiquetas lingüísticas a las entidades que se van aprendiendo.

De la misma forma, podrán generarse habilidades motoras nuevas. A partir del éxito o fracaso de una habilidad motora, el sistema puede aprender a modificar los parámetros que intervienen en dicha habilidad. A su vez, el sistema puede aprender a generar habilidades complejas compuestas de otras habilidades motoras o sensoriales. Así, una vez realizada una determinada tarea, el sistema puede considerar esa tarea como una habilidad

compuesta por todas las habilidades involucradas en ella. De esta forma, el nivel de complejidad de las habilidades depende de lo que haya aprendido el sistema a lo largo de la ejecución de diferentes tareas.

Dicho aprendizaje podría darse a varios niveles:

- Aprendizaje de los parámetros que intervienen en las diferentes habilidades motoras: Los parámetros que intervienen en los ciclos de control de las diferentes habilidades pueden ser modificados para adaptarse a diferentes situaciones o a diferentes entornos, para mejorar el comportamiento dinámico del robot.

- Generación de habilidades: Se pueden tener conductas cuya actuación sea aprendida directamente a través del entorno y de un objetivo a conseguir. Así puede realizarse una conducta que aprenda a seguir un objeto que se le indique.

- Composición de habilidades: A partir de habilidades motoras o sensoriales más simples, se puede enseñar al sistema a generar habilidades más complejas, integrándolas mediante fusión o secuenciamiento.

- Automatización de habilidades: Cuando se realiza de forma reiterada la secuencia de una serie habilidades, dicha secuencia puede pasar a formar una nueva habilidad. Esto ocurre también con las personas. Cuando se realiza una misma tarea repetidas veces, esa tarea, que al principio requería nuestra atención, deja de requerirla y la empezamos a realizar de forma automática. Es decir, procesos que al principio requieren de deliberación, pasan con el tiempo al nivel automático. Un ejemplo de esto ocurre cuando empezamos a conducir. Al principio se requiere toda la atención para poder controlar el vehículo, prestándose poca atención a tareas distintas de la conducción. Al final, se automatiza la capacidad de conducir y se puede prestar más atención al resto de tareas como hablar con otras personas.

## 5.7 Biblioteca de habilidades

Dichas habilidades no formarán un conjunto cerrado. Se podrán incorporar nuevas habilidades a la lista de manera que se pueda hacer uso de ellas en caso de que sea necesario. Para ello, las habilidades estarán incluidas en unas bibliotecas en las que se realizará una gestión dinámica de las mismas, actualizándose en todo momento. El planificador tiene que ser capaz de seleccionar de entre las habilidades sensoriales y motoras cuales son las que deben utilizarse en cada momento.

Dicha biblioteca estará estructurada en forma de base de datos. Cada habilidad motora o sensorial se incluirá como un registro. Para cada registro, tendremos que dar valores a unos campos determinados.

Cada vez que se añada una habilidad, deberá indicarse en que circunstancias podrá ser utilizada. Una forma de hacerlo es adjudicando un coeficiente de idoneidad a cada una de las habilidades. Esto llevaría a decir cuan idóneo es cada habilidad motora para realizar una determinada tarea. A su vez se debe indicar que requerimientos necesita esa determinada habilidad.