

Tema 3: Razonamiento Práctico (I)

Agentes Inteligentes (AIN)

Autores: Vicent Botti, Carlos Carrascosa, Vicente Julián

Tema 3- Índice

3.1 Arquitecturas de agente.

3.2 Agentes de Razonamiento Deductivo.

 3.2.1 Más Problemas...

 3.2.2 Sistemas de Planificación (en general)

 3.2.3 El Mundo de Bloques

 3.2.4 AGENT0 y PLACA

 3.2.5 Concurrent METATEM

3.3 Agentes Reactivos

 3.3.1 Arquitectura de Subsunción

 3.3.2 Red de Comportamientos para Agentes Situados

3.4 Agentes Híbridos

 3.4.1 Arquitectura Capas Horizontales: TouringMachines

 3.4.2 Arquitectura Capas Verticales: InteRRaP

3.5 Agentes de Razonamiento Práctico.

 3.5.1 Arquitecturas BDI

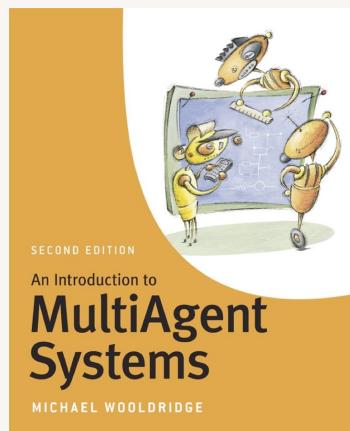
 3.5.2 Razonamiento Dirigido por el Objetivo

 3.5.4 Implementando Agentes de Razonamiento Práctico: JASON

3.6 Conclusión

Bibliografía

Bibliografía básica:

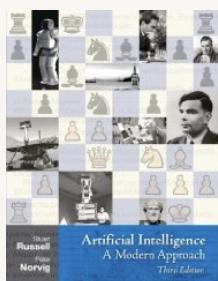


Tema 2

An Introduction to MultiAgent Systems – Second Edition
by Michael Wooldridge

Published May 2009 by John Wiley & Sons
ISBN-10: 0470519460
ISBN-13: 978-0470519462

Bibliografía complementaria:



Artificial Intelligence: A Modern Approach (Third edition)
By Sturat Rusell and Peter Norvig
Published by Prentice Hall Copyright © 2010
Published Date: Dec 1, 2009
ISBN-10: 0-13-604259-7
ISBN-13: 978-0-13-604259-4

Arquitecturas de Agente

- ✿ Un *agente* es un sistema informático capaz de acción autónoma flexible...
- ✿ Temas que hay que tratar para construir sistemas basados en agentes...
- ✿ 3 tipos de *arquitecturas* de agente:
 - ✿ simbólico/lógico
 - ✿ reactivo
 - ✿ híbrido

Arquitecturas de Agente Concretas

- * *Agentes de Razonamiento Deductivo*
 - * 1956 – presente
 - * “Agentes toman decisiones sobre qué hacer vía manipulación de símbolos. Su expresión más pura propone que los agentes usan razonamiento lógico explícito para decidir qué hacer.”
- * *Agentes Reactivos*
 - * 1985 – presente
 - * “Problemas con el razonamiento simbólico llevaron a una reacción contra esto - el llamado movimiento de agentes reactivos.”
- * *Agentes Híbridos*
 - * 1989 – presente
 - * “Arquitecturas híbridas intentan combinar lo mejor de arquitecturas reactivas y de razonamiento.”
- * *Agentes de Razonamiento Práctico*
 - * 1990 – presente
 - * “Agentes usan razonamiento práctico – *beliefs / desires / intentions.*”

Arquitecturas de Agente Concretas

- * *Agentes de Razonamiento Deductivo*
 - * 1956 – presente
 - * “Agentes toman decisiones sobre qué hacer vía manipulación de símbolos. Su expresión más pura propone que los agentes usan razonamiento lógico explícito para decidir qué hacer.”
- * *Agentes Reactivos*
 - * 1985 – presente
 - * “Problemas con el razonamiento simbólico llevaron a una reacción contra esto - el llamado movimiento de agentes reactivos.”
- * *Agentes Híbridos*
 - * 1989 – presente
 - * “Arquitecturas híbridas intentan combinar lo mejor de arquitecturas reactivas y de razonamiento.”
- * *Agentes de Razonamiento Práctico*
 - * 1990 – presente
 - * “Agentes usan razonamiento práctico – *beliefs / desires / intentions.*”

Agentes de Razonamiento Deductivo

- ✿ La aproximación clásica para construir agentes es verlos como un tipo particular de sistema basado en el conocimiento, y usar las metodologías de ese tipo de sistemas asociadas para darles soporte.
- ✿ Este paradigma es conocido como IA simbólica
- ✿ Se define un agente deliberativo o arquitectura de agente deliberativa como aquella que:
 - ✿ contiene un modelo del mundo simbólico explícitamente representado: bd de *creencias* especificadas usando fórmulas de lógica de predicados de primer orden
 - ✿ toma decisiones (por ej., sobre qué acciones realizar) via razonamiento simbólico: conjunto de *reglas de deducción*

Agentes de Razonamiento Deductivo

- ❖ Dos problemas clave a resolver para construir un agente así:

1. El problema de transducción:

traducir el mundo real en una descripción simbólica adecuada ([percibir la información relevante](#)), a tiempo para que dicha descripción sea útil... [visión, entendimiento del habla, aprendizaje](#)

2. El problema de representación / razonamiento:

cómo representar simbólicamente la información sobre entidades y procesos complejos del mundo real, y cómo obtener agentes para razonar con esta información a tiempo para que los resultados sean útiles... [representación del conocimiento, razonamiento automatizado, planificación automática](#)

Agentes de Razonamiento Deductivo

- La mayoría de investigadores aceptan que ningún de los problemas anteriores está completamente resuelto.
- Problema: la complejidad de la manipulación de símbolos: la mayoría de algoritmos basados en búsqueda son altamente intratables.
- Debido a estos problemas, algunos investigadores han buscado técnicas alternativas para construir agentes; las veremos más tarde.

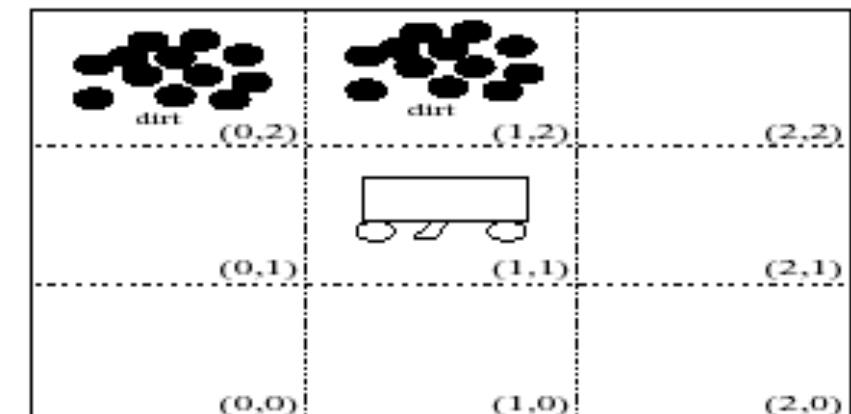
Agentes de Razonamiento Deductivo

- ❖ ¿Cómo puede un agente decidir qué hacer usando demostración de teoremas?
- ❖ La idea básica es usar la lógica para codificar una teoría que declare la mejor acción a realizar en cualquier situación dada
- ❖ Sea:
 - ❖ \mathcal{Q} esta teoría (típicamente un conjunto de reglas)
 - ❖ Δ una bd lógica que describe el estado actual del mundo
 - ❖ \mathcal{A} el conjunto de acciones que puede realizar el agente
 - ❖ $\Delta \models \mathcal{Q}\phi$ significa que ϕ puede ser probada en Δ usando \mathcal{Q}

Agentes de Razonamiento Deductivo

```
/* intentar encontrar una acción prescrita explícitamente */
for each a ∈ Ac do
    if  $\Delta \vdash_Q \text{Do}(a)$  then
        return a
    end-if
end-for
/* intentar encontrar una acción no excluida */
for each a ∈ Ac do
    if  $\neg(\Delta \vdash_Q \neg\text{Do}(a))$  then
        return a
    end-if
end-for
return null /* no se encontró ninguna acción */
```

Agentes de Razonamiento Deductivo



Agentes de Razonamiento Deductivo



- Usar 3 *predicados de dominio* para resolver el problema:
 - $In(x, y)$ el agente está en (x, y)
 - $Dirt(x, y)$ hay suciedad en la posición (x, y)
 - $Facing(d)$ el agente está orientado hacia d
 - $Wall(x, y)$ hay una pared en (x, y)
- Acciones posibles:
 - $Ac = \{turn, forward, suck\}$
- ($turn$ significa “girar a la derecha”)

Agentes de Razonamiento Deductivo



- Reglas Q para determinar qué hacer:

* $\forall x \forall y \text{ Wall}(x,y) \rightarrow \neg \text{Free}(x,y)$

* $\text{In}(x,y) \wedge \text{Dirt}(x,y) \rightarrow \text{Do}(\text{suck})$

$\text{In}(0,0) \wedge \text{Facing(north)} \wedge \neg \text{Dirt}(0,0) \rightarrow \text{Do(forward)}$

$\text{In}(0,1) \wedge \text{Facing(north)} \wedge \neg \text{Dirt}(0,1) \rightarrow \text{Do(forward)}$

$\text{In}(0,2) \wedge \text{Facing(north)} \wedge \neg \text{Dirt}(0,2) \rightarrow \text{Do(turn)}$

$\text{In}(0,2) \wedge \text{Facing(east)} \rightarrow \text{Do(forward)}$

- Usando estas reglas, comenzando en (0, 0) el robot limpiará la suciedad

Agentes de Razonamiento Deductivo



- ❖ Problemas:
 - ❖ ¿Cómo convertir la entrada de los sensores a?
- Dirt(0, 1)*
- ❖ toma de decisiones asume un entorno estático: **racionalidad calculada**
 - ❖ ¡toma de decisiones usando lógica de primer orden es **indecidible!**!
 - ❖ Incluso usando lógica proposicional, la toma de decisiones en el peor caso supone problemas co-NP-completos (**PS: co-NP-completo = ¡malas noticias!**)
 - ❖ Soluciones Típicas:
 - ❖ simplificar la lógica
 - ❖ usar represent. simbólicas, no lógicas
 - ❖ pasar el énfasis del razonam. de la ejecución al diseño

Más Problemas...

- La *aproximación lógica* presentada implica añadir y borrar cosas de una bd (no monotonía).
- Eso no es lógica de primer orden (lógica modal).
- Los primeros intentos de crear un *agente planificador* utilizaron deducción lógica de primer orden para resolver el problema.

AGENTO y PLACA

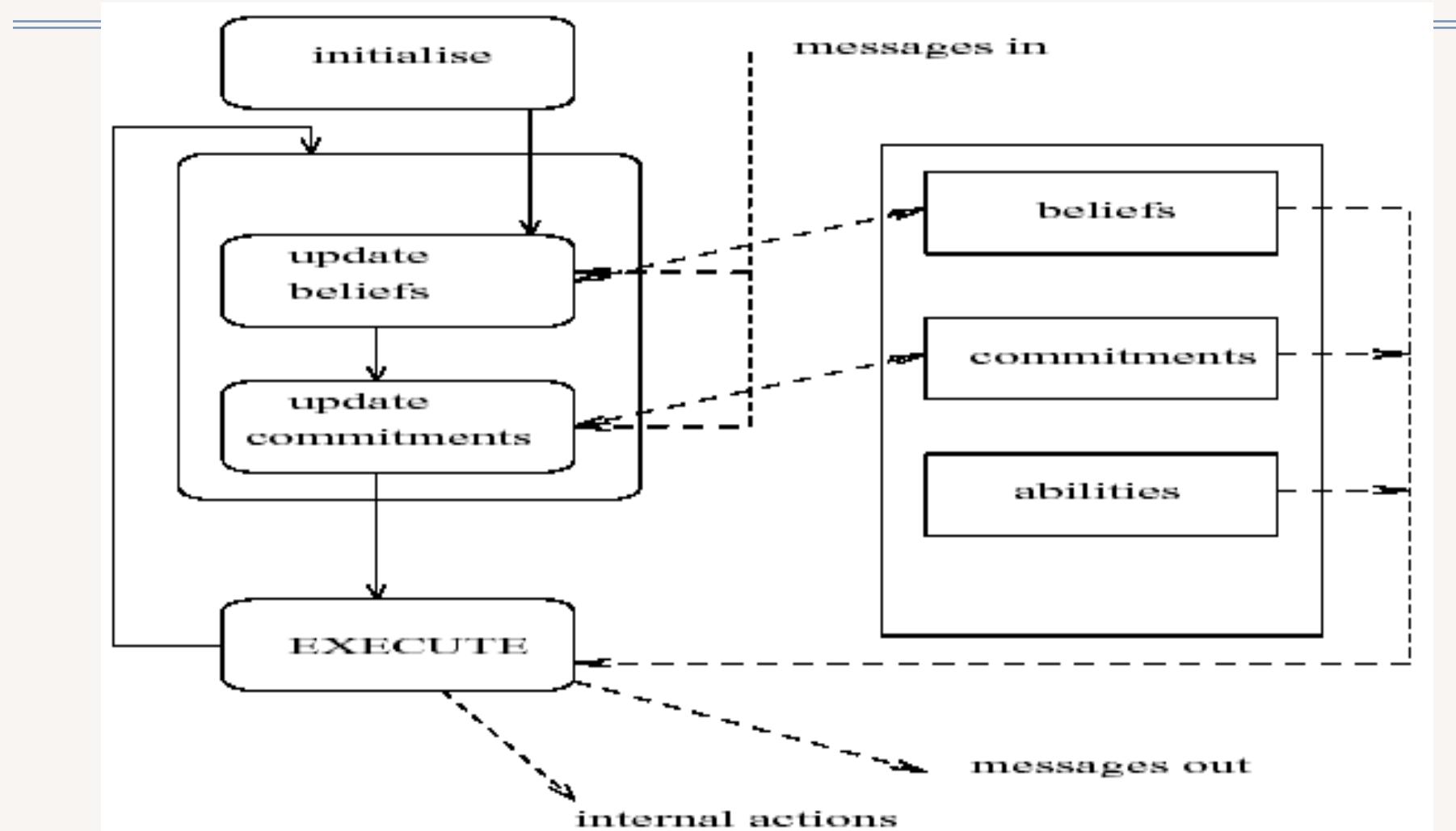
- ❖ Gran parte del interés en agentes de la comunidad de IA ha surgido a partir de la noción de Shoham de *Programación Orientada a Agentes* (AOP):
 - ❖ ‘Nuevo paradigma de la programación, basado en una visión social de la computación’
 - ❖ Programar agentes en términos de nociones intencionales (*creencia, compromiso e intención*)
 - ❖ Motivación: usar actitudes intencionales como *mecanismo de abstracción* para representar propiedades de sistemas complejos.

De la misma manera que usamos actitudes intencionales para describir humanos, puede ser útil usar actitudes intencionales para programar máquinas.

AGENT0

- Según Shoham, un sistema de AOP completo tendrá 3 componentes:
 - una *lógica* para especificar agentes y describir sus estados mentales
 - un *lenguaje de programación* interpretado para programar agentes
 - un proceso de *agentificación*, para convertir aplicaciones (por ej., bd) en agentes
- La relación entre lógica y lenguaje de programación es la semántica
- AGENT0 se considera el 1^{er} lenguaje de AOP (saltándose la lógica)

AGENTO



Arquitecturas de Agente Concretas

- * *Agentes de Razonamiento Deductivo*
 - * 1956 – presente
 - * “Agentes toman decisiones sobre qué hacer vía manipulación de símbolos. Su expresión más pura propone que los agentes usan razonamiento lógico explícito para decidir qué hacer.”
- * *Agentes Reactivos*
 - * 1985 – presente
 - * “Problemas con el razonamiento simbólico llevaron a una reacción contra esto - el llamado movimiento de agentes reactivos.”
- * *Agentes Híbridos*
 - * 1989 – presente
 - * “Arquitecturas híbridas intentan combinar lo mejor de arquitecturas reactivas y de razonamiento.”
- * *Agentes de Razonamiento Práctico*
 - * 1990 – presente
 - * “Agentes usan razonamiento práctico – *beliefs / desires / intentions.*”

Agentes Reactivos

- ❖ “**simbolismo**”



- ❖ “inteligencia requiere una representación simbólica del mundo”

- ❖ “**conexionismo**”

- ❖ Comportamiento inteligente es producto de interacción con el entorno y emerge de comportamientos simples

- ❖ Ejemplo: comportamiento social (ej. Insectos sociales)

Agentes Reactivos

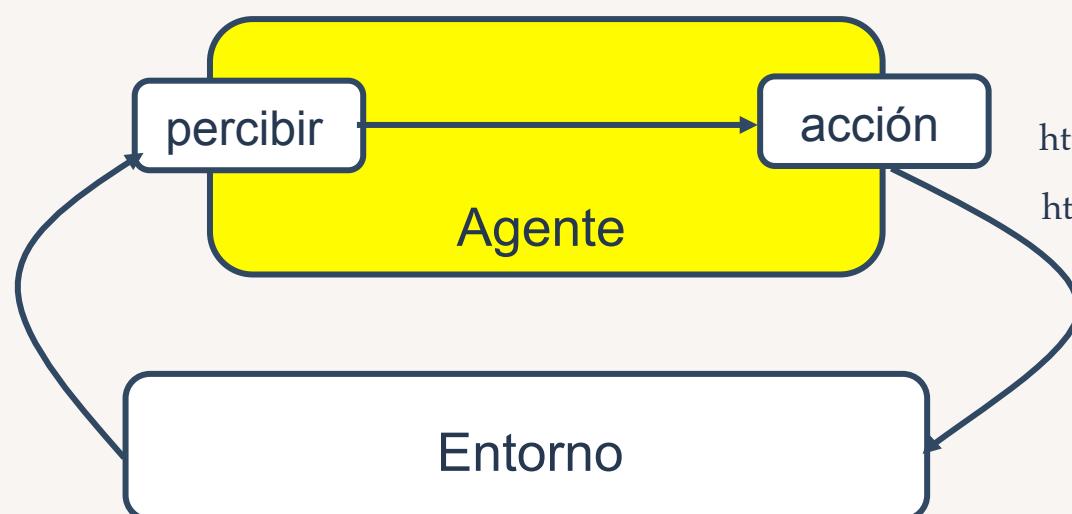
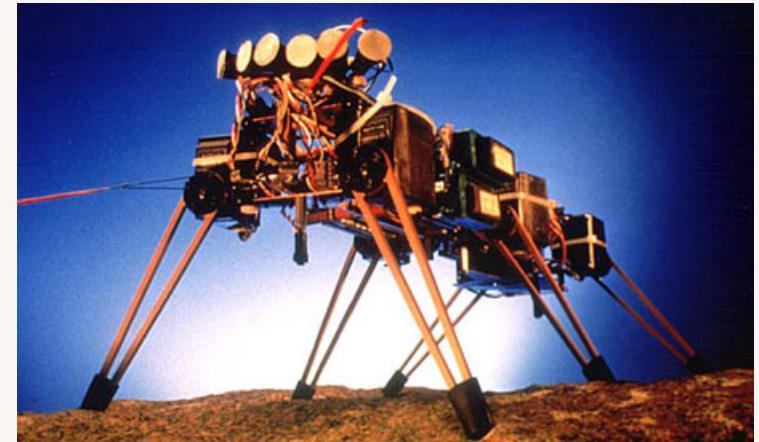
- ✿ Arquitectura de Subsunción de Brooks
- ✿ PENGI
- ✿ Automata Situado RULER / GAPPS
- ✿ “Arquitectura de Red de Comportamientos” de Pattie Maes
- ✿ Arquitectura de Flujo Libre

Rodney Brooks: Arquitectura de Subsunción

- ❖ 2 ideas básicas
 - ❖ Estar situado y materializado: Inteligencia real está situada en el mundo, con un cuerpo, no como sistemas incorpóreos tales como demostradores de teoremas o sistemas expertos.
 - ❖ Inteligencia y emergencia
 - ❖ Comportamiento “inteligente” surge como resultado de la interacción de un agente con su entorno
 - ❖ Inteligencia está ‘en el ojo del observador’, no es una propiedad aislada, innata
- ❖ 2 tesis claves
 - ❖ Inteligencia sin representación: comportamiento inteligente puede ser logrado sin representaciones explícitas del tipo que la IA simbólica propone
 - ❖ Inteligencia sin razonamiento: comportamiento inteligente puede ser logrado sin razonamiento abstracto explícito del tipo que la IA simbólica propone

Rodney Brooks: Arquitectura de Subsunción

- computacionalmente: muy simple
- “*alguno de los robots hacen tareas que serían impresionantes si fuesen conseguidas por sistemas de IA simbólicos*”



<http://www.youtube.com/watch?v=C9p8B7-5MTI>

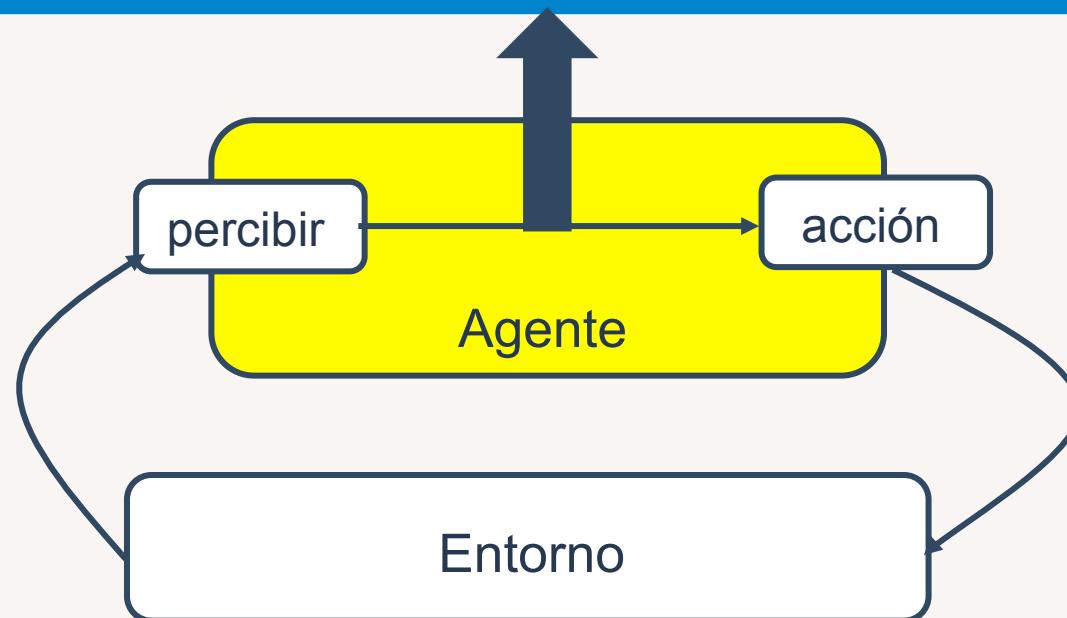
http://www.youtube.com/watch?v=mc4T3_f1c5Q

Rodney Brooks: Arquitectura de Subsunción

¿Cómo se relacionan las percepciones con las acciones?

Mediante el concepto de **comportamiento** / regla condición-acción

SI percepciones **ENTONCES** acción



Rodney Brooks: Arquitectura de Subsunción

CONJUNTO DE COMPORTAMIENTOS

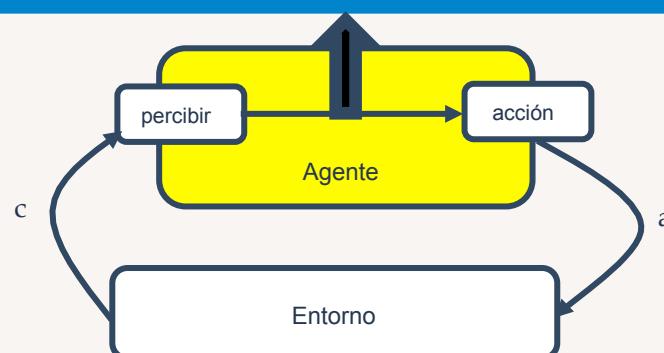
Un comportamiento es un par (c,a)

donde $c \subseteq P$ es un conjunto de percepciones llamadas la *condición* y
 $a \in A$ es una *acción*

(P es el conjunto de todas las percepciones posibles y A el conjunto de acciones que el agente puede ejecutar)

$$\text{Beh} = \{(c,a) \mid c \subseteq P \text{ y } a \in A\}$$

es el conjunto de todos los comportamientos

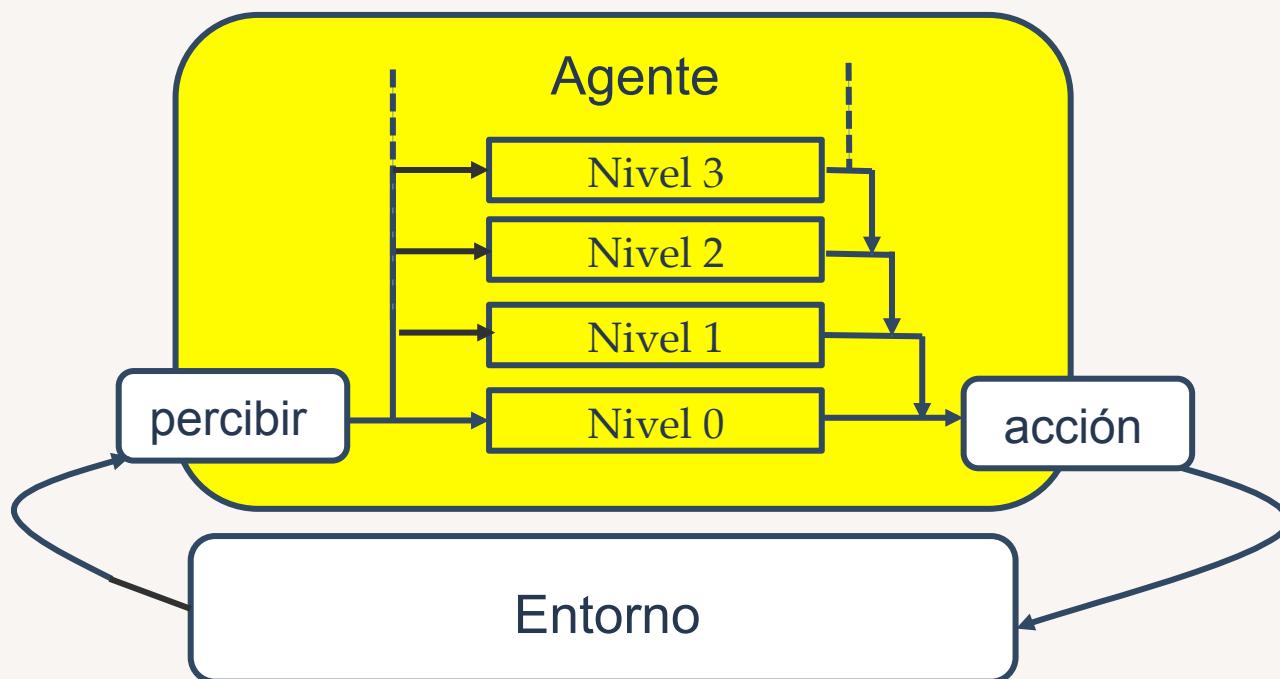


Rodney Brooks: Arquitectura de Subsunción

- Un comportamiento (c,a) puede *dispararse* cuando el entorno esté en el estado $s \in S$ si $c \subseteq \text{percibir}(s)$
- En un estado del entorno ($s \in S$) puede dispararse más de un comportamiento de Beh:
$$\text{disparados} = \{(c,a) \mid (c,a) \in \text{Beh} \text{ y } c \subseteq \text{percibir}(s)\}$$
- Como sólo se puede ejecutar una acción tendremos que elegir un comportamiento de *disparados*.
 - Comportamientos compiten para controlar el agente
 - Selección de acciones: organizando comportamientos en capas (jerarquía)
 - Capas más bajas inhiben (“subsumen”) capas más altas
 - Capas más bajas representan clases de comportamientos más primitivos (tales como evitar obstáculos), y tienen precedencia (sus acciones se ejecutan antes) sobre capas más altas en la jerarquía

Rodney Brooks: Arquitectura de Subsunción

Jerarquía de comportamientos



Rodney Brooks: Arquitectura de Subsuspción

- Jerarquía de comportamientos:

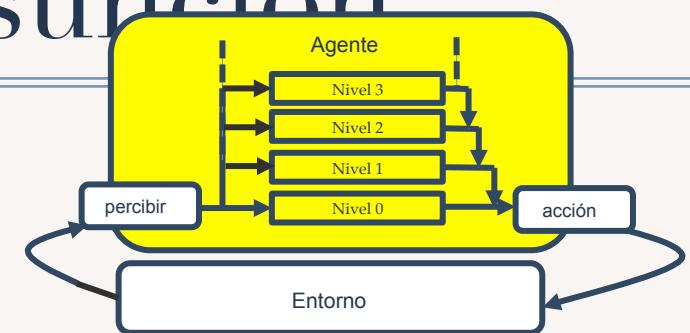
Asociamos a un conjunto de reglas de comportamiento de un agente $R \subseteq \text{Beh}$ una *relación de inhibición binaria* ($<$) en el conjunto de comportamientos, es una *relación de orden total* en R

_entonces dados $b_1, b_2 \in R$ si $b_1 < b_2$ [$(b_1, b_2) \in <$] decimos que

“ b_1 inhibe b_2 ” “ b_1 subsume b_2 ”

(b_1 es más pequeño en la jerarquía que b_2 , y tendrá prioridad sobre b_2)

- La “función de selección de acción” se define como sigue...



Rodney Brooks: Arquitectura de Subsunción

```
function action (p : P, R: Beh) : A
var fired : Beh
var selected : A
Begin
    selected := null
    fired := {(c,a) | (c,a) ∈ R and c ⊆ p}
    for each (c,a) ∈ fired do
        if not ∃(c',a') ∈ fired such that (c',a') < (c,a) then
            selected:= a
        end-if
    end for
    return selected
end function action
```

The flowchart illustrates the execution of the 'action' function:

- An arrow points from the first line of the function definition to a box labeled "Calcular todos los comportamientos que puede disparar".
- An arrow points from the assignment of 'selected' to a box labeled "Determinar si algunos comportamientos subsumen a otros".
- An arrow points from the final 'return selected' statement to a box labeled "Devolver acción apropiada o null".

Arquitectura de Subsunción: Ejemplo: Mars Explorer

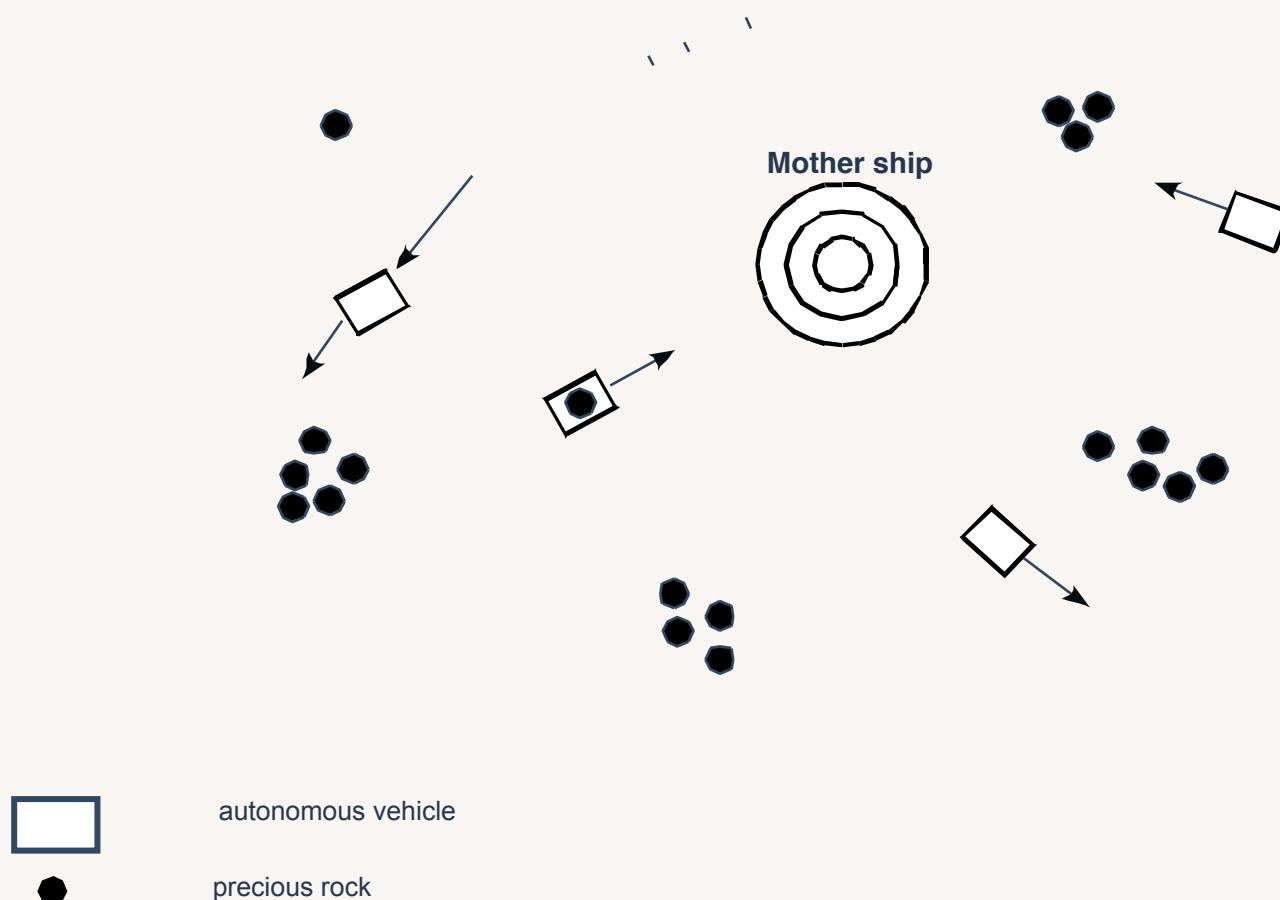


⊕ Mars explorer (L. Steels)

⊕ Objetivo

- ⊕ Explorar un planeta distante, y, en particular, recoger muestras de una roca preciosa.
- ⊕ La localización de las muestras no se conoce de antemano, pero se sabe que están apiñadas.
- ⊕ La nave nodriza transmite la señal de radio (que se debilita con la distancia).
- ⊕ Mapa no disponible.
- ⊕ Colaborativo.

Ejemplo: Mars Explorer



Arquitectura de Subsunción: Ejemplo: Mars Explorer

- ❖ Solución de un único *Explorer*:
 - ❖ comportamientos / reglas
 1. **Si** obstáculo **entonces** cambia dirección
 2. **Si** llevando muestras y en la base
entonces dejarlas caer
 3. **Si** llevando muestras y no en la base
entonces viajar al campo de gradiente de la señal de la base
 4. **Si** detecto muestra **entonces** recogerla
 5. **Si** cierto **entonces** caminar aleatoriamente
 - ❖ Relación de orden total:
$$1 < 2 < 3 < 4 < 5$$



Arquitectura de Subsunción: Ejemplo: Mars Explorer

- ❖ Solución de múltiples *Explorers*:
- ❖ Si un agente encuentra una agrupación de rocas – comunicarlo ?
 - ❖ rango ?
 - ❖ posición ?
 - ❖ Cómo tratar con tales mensajes ? Puede estar lejos...
- ❖ Comunicación indirecta (através del entorno):
 - ❖ Cada agente lleva “migajas radioactivas”, que pueden ser arrojadas, recogidas y detectadas por robots que pasen
 - ❖ Comunicación via entorno es llamada **stigmergy**

Arquitectura de Subsunción: Ejemplo: Mars Explorer

- Solución inspirada en el comportamiento de búsqueda de comida de las hormigas
 - Agente crea un *rastro* de migajas radiactivas de regreso a la nave nodriza siempre que encuentra una muestra de roca
 - Si otro agente se encuentra un rastro, puede seguirlo hasta el grupo de muestras
- Refinamiento:
 - Agentes que siguen el rastro a las muestras recogen algunas migajas para hacer el rastro más débil
 - El rastro que lleva a un grupo vacío será finalmente borrado

Arquitectura de Subsunción: Ejemplo: Mars Explorer

- ❖ Conjunto de reglas modificadas

<http://www.youtube.com/watch?v=H68YF9YKKJ8>

Si detecta un obstáculo entonces cambiar dirección

Si llevando muestras y en la base
 entonces arrojar muestras

Si llevando muestras y no en la base
 entonces arrojar 2 migajas y viajar por el gradiente

Si detecta una muestra entonces recoger muestra

Si percibe migajas entonces recoger una migaja y viajar por el gradiente

Si cierto entonces mover aleatoriamente (nada mejor que hacer)

- ❖ Relación de orden: $1 < 2 < 3 < 4 < 5 < 6$
- ❖ Logra ejecuciones casi óptimas en muchas situaciones
- ❖ Solución barata y robusta (la pérdida de un agente no es crítica).
- ❖ L. Steels argumenta que los agentes deliberativos son *enteramente irreales* para este problema.

Arquitectura de Subsunción: Ejemplo: Mars Explorer

- ❖ Ventajas
 - ❖ simple
 - ❖ económico
 - ❖ computacionalmente tratable
 - ❖ robusto frente a fallos
- ❖ Desventajas
 - ❖ agentes actúan a corto plazo ya que usan sólo información local
 - ❖ sin aprendizaje
 - ❖ ¿cómo desarrollar estos agentes? Difícil si interactúan más de 10 reglas
 - ❖ no hay herramientas formales para analizar y predecir

Arquitecturas de Agente Concretas

- * *Agentes de Razonamiento Deductivo*
 - * 1956 – presente
 - * “Agentes toman decisiones sobre qué hacer vía manipulación de símbolos. Su expresión más pura propone que los agentes usan razonamiento lógico explícito para decidir qué hacer.”
- * *Agentes Reactivos*
 - * 1985 – presente
 - * “Problemas con el razonamiento simbólico llevaron a una reacción contra esto - el llamado movimiento de agentes reactivos.”
- * *Agentes Híbridos*
 - * 1989 – presente
 - * “Arquitecturas híbridas intentan combinar lo mejor de arquitecturas reactivas y de razonamiento.”
- * *Agentes de Razonamiento Práctico*
 - * 1990 – presente
 - * “Agentes usan razonamiento práctico – *beliefs / desires / intentions.*”

Agentes Híbridos

- ❖ ¿lo mejor de ambos mundos?
 - ❖ deliberativo
 - ❖ reactivo
- ❖ Aproximación obvia:
 - ❖ Construir un agente a partir de dos o más subsistemas:
 - ❖ uno deliberativo, conteniendo un modelo del mundo simbólico, que desarrolla planes y toma decisiones de la manera propuesta por la IA simbólica
 - ❖ uno reactivo, que es capaz de reaccionar a eventos sin razonamiento complejo
- ❖ En su mayoría
 - ❖ Se le da precedencia al componente reactivo sobre el deliberativo
- ❖ Esta clase de estructuración lleva naturalmente a la idea de arquitectura *en capas*

Agentes Híbridos

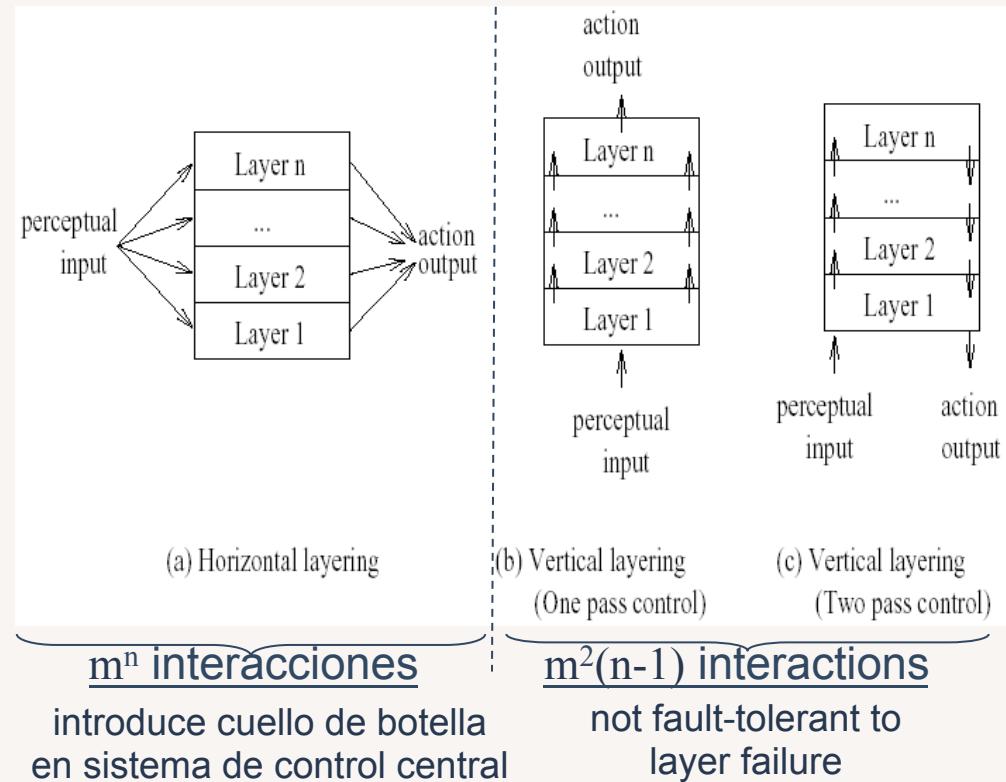
- Capas Horizontales

m acciones posibles sugeridas por cada capa, n capas

- Capas directamente conectadas a la entrada sensorial y la salida de acciones

- Capas Verticales

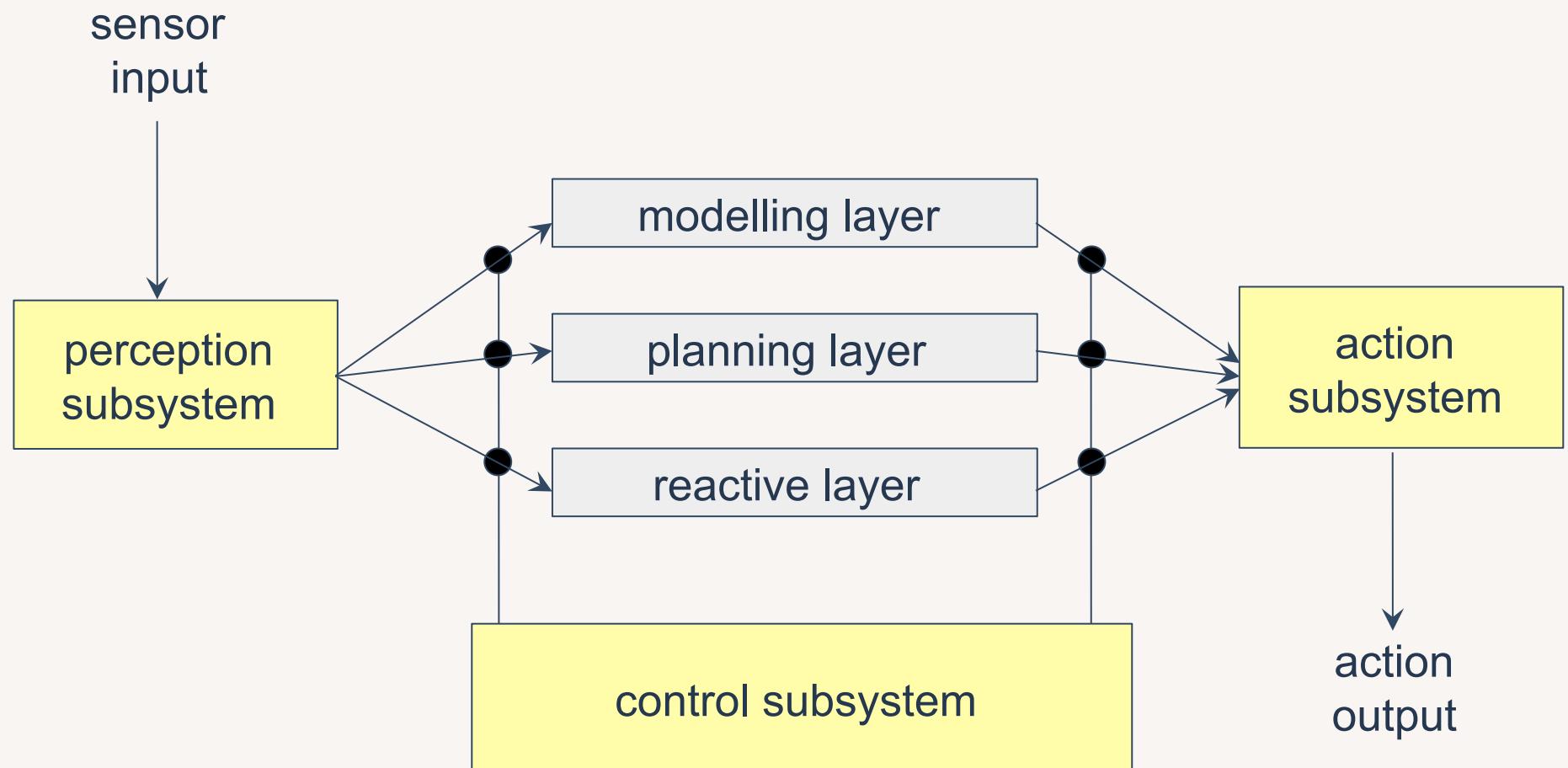
- Entrada sensorial y salida de acciones son tratadas por una capa como mucho



Arquitectura Capas Horizontales: Ferguson – TouringMachines

- ❖ Subsistemas de percepción y acción
interfaz directamente con el entorno del agente
- ❖ 3 capas producen actividad
 - ❖ Capa Reactiva
 - ❖ +- respuesta inmediata a cambios en el entorno
Reglas situación - acción
 - ❖ Capa de Planificación
 - ❖ Comportamiento pro-activo
 - ❖ usa una biblioteca de esqueletos de planes (esquemas)
 - ❖ Capa de Modelado
 - ❖ representa las diversas entidades en el mundo
 - ❖ Predice conflictos y genera nuevos objetivos para resolver estos conflictos
- ❖ un subsistema de control,
un conjunto de reglas de control para decidir qué capa debería tener control sobre el agente

Arquitectura Capas Horizontales: Ferguson – TouringMachines



Arquitectura Capas Horizontales: Ferguson – TouringMachines

- Capa Reactiva

Ejemplo:

```
rule-1: kerb-avoidance
  if
    is-in-front (Kerb, Observer) and
    speed (Observer) > 0 and
    separation (Kerb, Observer) < KerbThreshHold
  then
    change-orientation (KerbAvoidanceAngle)
```

- Capa de Planificación

agente

planifica y selecciona acciones para conseguir los objetivos del

- Capa de Modelado

repr. simbólica del *estado cognitivo* de otros agentes en el entorno del agente

- Capas comunican entre sí y están embebidas en un framework de control, que usa reglas de control

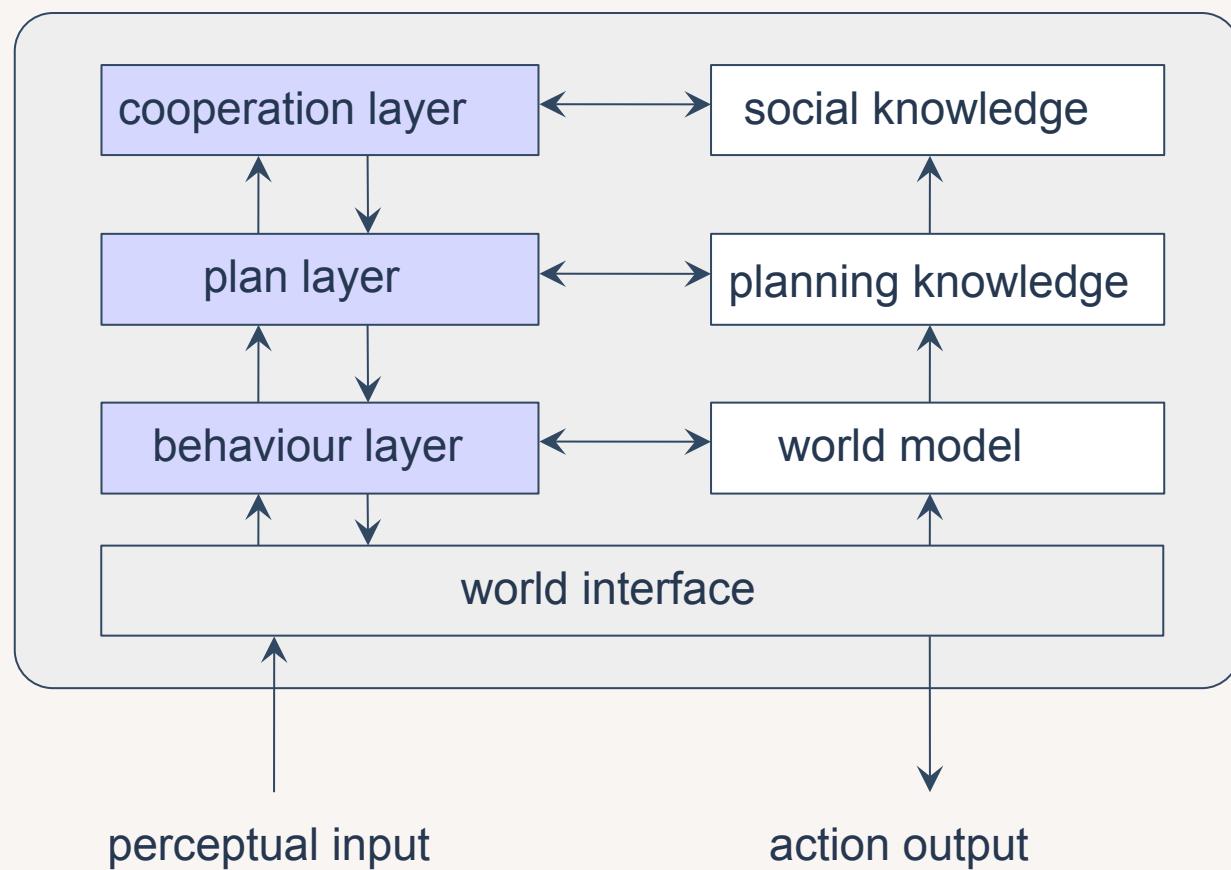
Ejemplo:

```
censor-rule-1:
  if
    entity(obstacle-6) in perception-buffer
  then
    remove-sensory-record(layer-R, entity(obstacle-6))
```

Arquitectura Capas Horizontales

- ❖ Ventajas
 - ❖ Simplicidad Conceptual
 - ❖ Si necesitamos que un agente presente n tipos diferentes de comportamientos, implementamos n capas diferentes
- ❖ Desventajas
 - ❖ Capas Competitivas → ¿no se garantiza comportamiento coherente?
 - ❖ Para evitar esto: función mediador
 - ❖ Toma decisiones sobre qué capa tiene control en un momento dado
 - ❖ La necesidad para tal *control central* puede ser problemática
 - ❖ Diseñador del agente debe considerar potencialmente todas las posibles interacciones entre capas

Arquitectura Capas Verticales: Müller - InteRRaP



Arquitectura Capas Verticales: Müller - InteRRaP

- **3 capas**
 - Capa basada en el comportamiento comportamiento reactivo
 - Capa de Planificación Local para planificación diaria
 - Planificación Co-operativa para interacciones sociales
 - **control entre capas:**
 - Activación de abajo a arriba
 - Si la capa no es competente para tratar con la situación: pasa control a la capa de arriba
 - Ejecución de arriba a abajo
 - Cuando una capa más alta hace uso de las facilidades proveidas por una capa más baja para lograr uno de sus objetivos
 - **Flujo de Control de Agentes**
 - Entrada Perceptual llega a la capa más baja
 - Si capa reactiva puede tratar con esta entrada: lo hará así
si no: activación de abajo a arriba (control pasado a capa de planificación local)
 - Si capa de planificación local puede manejar la situación:
lo hará así (haciendo uso de la ejecución de arriba a abajo)
si no: activación de abajo a arriba pasa control a la capa de arriba

Arquitectura Capas Verticales

- ❖ Ventajas
 - ❖ Complejidad de interacciones entre capas es reducida
 - ❖ $n-1$ interfaces entre n capas
 - ➔ Si cada capa es capaz de sugerir m acciones en como mucho m^2 ($n-1$) interacciones
 - ➔ Mucho más simple que el caso horizontal (m^n interacciones)
- ❖ Desventajas
 - ❖ Esta simplicidad tiene como coste flexibilidad
 - ❖ control debe pasar entre las diferentes capas no tolerante a fallos:
 - ❖ fallos en cualquier capa pueden tener serias consecuencias para la ejecución del agente