Técnicas Avanzadas de Inteligencia Artificial

Curso 2013-2014

German Rigau

german.rigau@ehu.es

http://adimen.si.ehu.es/~rigau

Grado en Ingeniería en Informática /
Ingeniería en Informática

Temario

1. Agentes Inteligentes
2. Sistemas Multiagentes
3. Planificación

ORIGINAL MOTION PICTURE SOUNDTRACK

M-G-M PRESENTS A STANLEY KUBRICK PRODUCTION

2001: a space odyssey

2001: A SPACE ODYSSEY

1 Agentes Inteligentes

1. Introducción
2. Evolución de los Agentes
3. Arquitecturas de Agentes

1.1 Introducción

Debido a fallos inesperados del sistema, una sonda espacial que se está acercando a Saturno se desorienta y pierde contacto con su base en la Tierra.

En lugar de desaparecer en el vacío, la sonda reconoce qué ha ocurrido un fallo crucial, lo diagnostica y aísla, lo corrige, se reorienta y toma de nuevo contacto con la base.

NASA Deep Space 1 –DS1– 1998

1.1 Introducción

El sistema de control de tráfico aéreo del aeropuerto principal de Ruritania falla repentinamente, dejando sin control aéreo a los vuelos que se encuentran en su vecindad.

Distributed Vehicle Monitoring Testbed DVMT, 1991

OASIS 1992 Sydney

Afortunadamente, los sistemas de control de tráfico aéreo de los aeropuertos vecinos reconocen su fallo y cooperan para seguir y manejar los vuelos afectados.

La situación potencialmente desastrosa finaliza sin incidentes.

1.1 Introducción

Después de un curso horroroso necesitas unas vacaciones en algún lugar seco y cálido. Después de especificar tus requisitos a un asistente personal digital (PDA), éste conversa con varios sitios web que venden vuelos, contratan habitaciones de hotel y alquilan coches.

¡¡Después de negociar duramente con ellos de tu parte, tu PDA te muestra un paquete de vacaciones perfecto!!

1.1 Introducción

- Los agentes pueden ayudar a: negociar precios, buscar productos más baratos, organizar viajes, ...
- Tipos de agentes: agentes turísticos, comerciales, agentes de bolsa, judiciales, ...
- *Qué es un agente en general?*

1.1 Introducción

- Características de la **Tecnología de Agentes**:
  - Recoge los trabajos de tres décadas de ingeniería informática e IA.
  - Fusión de tres corrientes especialmente:
    - Ingeniería del Software (IS)
    - Inteligencia Artificial (IA)
    - Sistemas Distribuidos

1.1 Introducción

- De **IS** (Tecnología de Objetos):
  - Encapsulamiento, independencia
  - Mensajes entre objetos (comunicación)
  - Clases, herencia

1.1 Introducción

- De **IA**:
  - Conocimiento (representación del mundo):
    - reglas, frames, lógica ...
  - Razonamiento, ...
  - Aprendizaje, ...
  - Visión, lenguaje, ...

  - Enfoque de agente “inteligente”:
    - Sensores
    - Proceso inteligente
    - Efectores (o actuadores)

1.1 Introducción

- De **Sistemas Distribuidos**:
  - Distribución de datos y procesos
  - Conectividad, Redes, Protocolos
  - Interoperabilidad
  - Internet

  - Especialmente Sistemas Multi-agente!

1.1 Introducción

- Concepto de agente:
  - No existe una definición comúnmente aceptada.
  - (Wooldridge & Jennings, 1995)
    - Cualquier proceso computacional situado en un entorno y capaz de realizar acciones autónomas en ese entorno para alcanzar sus objetivos.

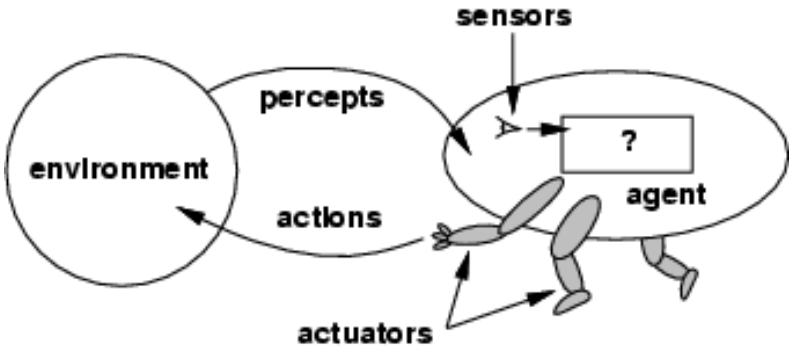
1.1 Introducción

- Características de los Agentes (1):
  - **Autonomía**: capacidad de actuar sin intervención humana directa o de otros agentes.
  - **Reactividad**: un agente está inmerso en un determinado entorno (habitat), del que percibe estímulos y ante los que debe reaccionar en un tiempo preestablecido.
  - **Iniciativa**: un agente no sólo debe reaccionar a los cambios que se produzcan en su entorno, sino que tiene que tener un carácter emprendedor y tomar la iniciativa para actuar guiado por los objetivos que debe de satisfacer.
  - **Racionalidad**: tiene unos objetivos específicos y siempre intenta llevarlos a cabo.

1.1 Introducción

- Características de los Agentes (2):
  - **Sociabilidad**: capacidad de interaccionar con otros agentes, utilizando como medio algún lenguaje de comunicación entre agentes.
  - **Movilidad**: habilidad de trasladarse en una red de comunicación informática.
  - **Veracidad**: no comunica información falsa intencionadamente.
  - **Benevolencia**: no tiene objetivos contradictorios y siempre intenta realizar la tarea que se le solicita.

1.1 Introducción



- Sensores: para percibir el entorno
- Actuadores: para modificar el entorno

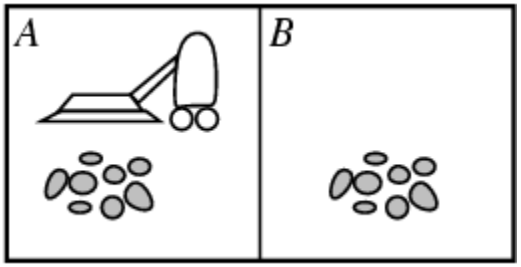
- *Qué puede ser un sensor? Qué un actuador?*

- Las acciones son función de la historia de percepciones

$$[f: P^* \rightarrow A]$$

1.1 Introducción

- El mundo (simple) de una aspiradora



- Percibe: lugar y contenido, p.e. [A, Sucio]
- Acciones: Izquierda, Derecha, Aspirar, NoAcc.

- *Cual debería ser el comportamiento de un agente racional?*

1.1 Introducción

- Un agente debe tratar de "hacer lo correcto", según lo que perciba y las acciones que pueda realizar.
- Medida de rendimiento: un criterio objetivo para medir el éxito de la conducta de un agente
  - Por ejemplo, para un agente aspiradora podría ser la cantidad de suciedad recogida, la cantidad de tiempo empleado, la cantidad de electricidad consumida, la cantidad de ruido generado, etc.

1.1 Introducción

- Para cada secuencia de percepciones posible, un agente racional debe seleccionar una acción que maximize su medida de rendimiento, teniendo en cuenta las evidencias aportadas por la secuencia de percepciones y todo el conocimiento que incorpore el agente.

1.1 Introducción

- La racionalidad es distinta de la omnisciencia (conocimiento absoluto)
- Los agentes pueden llevar a cabo acciones con el fin de obtener información útil (recopilación de información, exploración del entorno)
- Un agente es autónomo si su comportamiento está determinado por su propia experiencia (con la capacidad de aprender y adaptarse)

1.1 Introducción

- Ejemplo (**PEAS**): taxi automático ...

- Medida de rendimiento (**P**erformance Measure):
  - seguridad, comodidad, rapidez, legalidad, maximizar los beneficios, ...
- Entorno (**E**nvironment):
  - calles, semáforos, tráfico, peatones, clientes, ...
- Actuadores (**A**ctuators):
  - volante, acelerador, freno, señal, bocina, ...
- Sensores (**S**ensors):
  - cámaras, sonar, velocímetro, GPS, sensores del motor, micrófonos, ...

Tipos de Entorno

- Propuestos por (Russell and Norvig, 2010):
  - observable vs parcialmente observable;
  - determinista vs no determinista;
  - episódico vs no episódico;
  - estatico vs dinámico;
  - discreto vs continuo.

Observable vs. parcialmente observable

- Un entorno es observable si un agente puede obtener información
  - completa
  - correcta
  - actualizada

  sobre su estado.
- Así, los sensores de un agente le dan acceso al estado completo del entorno en cada instante de tiempo.

- Cuanto más observable sea un entorno, más fácil es construir agentes que pueden operar en el mismo.
- La mayoría de los entornos de la vida real, no son accesibles.

Determinista vs. no determinista

- Un entorno es determinista si cualquier acción tiene un único efecto sobre él, y no hay incertidumbre sobre el estado resultante.
- Así, el siguiente estado del entorno está completamente determinado por el estado actual y la acción ejecutada por el agente.

- Los entornos no deterministas son más problemáticos
- El mundo físico, a todos los efectos, se puede considerar como no determinista.
- En entornos complejos, aunque sean esencialmente deterministas, predecir el efecto de una acción puede ser demasiado complejo para ser factible.

Episódico vs. no episódico

- Un entorno es episódico si el comportamiento del agente puede ser dividido en secuencias de percepción-acción no relacionados entre sí (episodios).
- Así, la experiencia del agente se divide en "episodios" atómicos (cada episodio consiste en el agente percibiendo y luego realizando una sola acción), y la elección de la acción en cada episodio sólo depende del propio episodio.

- Los entornos episódicos son más fáciles para los desarrolladores porque el agente puede decidir qué acción realizar sólo sobre la base del episodio actual;
- no necesita recordar episodios anteriores o razonar sobre los próximos.

Estático vs. dinámico

- Un entorno es estático si podemos suponer que permanece sin cambios (exceptuando las acciones de los propios agentes).
- Así, el entorno no cambia mientras el agente está deliberando.
- El entorno es semidinámico si no cambia con el paso del tiempo, pero de comportamiento del agente sí.

- Los entornos dinámicos son más difíciles para el desarrollador ya que otras entidades pueden interferir con las acciones del agente.
- Muchos entornos de la vida real son muy dinámicos:
  - el mundo real,
  - Internet ...

Discreto vs. contínuo

- Un entorno es discreto si hay un número fijo, finito de acciones y percepciones en el mismo.
- Así, el entorno puede quedar descrito por un número limitado de percepciones y acciones claramente definidas.

- El ajedrez describe un entorno discreto.
- La conducción de un taxi se encuentra en un entorno de continuo.

- Evidentemente, los entornos discretos son más fáciles para los desarrolladores.

1.1 Introducción

- Sistema Basado en Agentes
  - Utiliza los agentes como mecanismo de abstracción, pero aún siendo modelado en términos de agentes, puede ser implementado sin ninguna estructura software correspondiente a éstos.

- Sistemas Multi-agente
  - Es diseñado e implementado como varios agentes interactuando entre sí, para así lograr la funcionalidad deseada.

1.1 Introducción

- Ventajas de la tecnología de agentes:
  - Mejora la funcionalidad y la calidad.
  - Menor coste (reusabilidad).
  - Reduce mantenimiento.
  - Se integra adecuadamente con otras tecnologías (web, BD, componentes, …)
  - Simplifican la labor de los ingenieros (patrones de agentes).

1 Agentes Inteligentes

1. Introducción
2. Evolución de los Agentes
3. Arquitecturas de Agentes

1.2 Evolución de los Agentes

- Inicios: (1975-1980): Los primeros trabajos en el área de la Inteligencia Artificial (IA), ...
- IA Distribuida (80s): Arquitectura de pizarra, red de contratos (negociación), organización y sociedades científicas, ...
- Consolidación (90s): Congresos y publicaciones científicas, prototipos de interés industrial, agentes móviles, programación orientada a agentes, ...

- *Qué congresos (nacionales e internacionales) hay sobre sistemas multiagente?*

1.2 Evolución de los Agentes

- Tipos de agentes:
  - Según sus características individuales:
  - Agentes **reactivos**, tareas sencillas en un ciclo de recepción de eventos/reacción.
  - Agentes **cognitivos**, tareas complejas (razonamiento, planificación o aprendizaje) en un ciclo percención-asimilación-razonamiento-actuación.

1.2 Evolución de los Agentes

- Tipos de agentes:
  - Según el modo de interacción:
  - **Agente-agente**: lenguajes ACL y KQML, y protocolos de comunicación RMI, CORBA, SOAP, HTML, ...
  - **Agente-entorno**: BD, servidores, librerías, ...
  - **Agente-persona**: lenguaje natural (voz o texto), sensores, lenguajes semiformales, gráficas, ... agentes de interfaz

1.2 Evolución de los Agentes

- Tipos de agentes:
  - Según el modo de organización:
  - Agentes **individualistas**
  - Agentes **cooperantes**, roles, responsabilidades, planes comunes, normas, resolución de conflictos (agentes especialistas, negociación).

1.2 Evolución de los Agentes

- Tipos de agentes:
  - Según su utilidad:
  - **Dominio de aplicación:** comercio electrónico, telecomunicaciones, economía (bolsa), administración, ocio y entretenimiento, …
  - **Tarea que realizan:** monitorización, diagnóstico, búsqueda de información, control de sistemas

Agente Inteligente

- Un agente inteligente es un sistema capaz de acciones *autónomas* y *flexibles* en algunos *entornos*.

- Flexible significa (Wooldridge y Jennings, 1995):

  - reactivo
  - proactivo
  - social

Reactividad, Proactividad, Sociabilidad

- La reactividad es la capacidad de un agente para *percibir* su entorno, y para *responder* de manera oportuna a los *cambios* que se producen en el mismo, con el fin de cumplir sus objetivos de diseño.

- La proactividad es la capacidad de un agente para *tomar la iniciativa* con el fin de satisfacer sus objetivos de diseño.

- La sociabilidad es la capacidad de un agente para *interactuar* con otros agentes con el fin de satisfacer sus objetivos de diseño.
- Interactuar significa *cooperar*, *coordinar*, *negociar*.

Reactividad, Proactividad, Sociabilidad

- Muy difícil (de hecho, el problema de la investigación abierto) si se requiere que un agente sea reactivo, proactivo y social simultánemente.
- Lo que se busca es un equilibrio entre:
  - Planificar objetivos alcanzables
  - Perseguir los objetivos
  - Reaccionar a los cambios en el entorno
  - Reconocer las oportunidades del momento
  - Interactuando con otros agentes
  - ...
- ¿Cómo debe el agente distribuir sus recursos y el tiempo entre estos objetivos?
- Difícil, incluso para los seres humanos!

AI vs. DAI

AI	**DAI**
un **solo** agente	**múltiples** agentes
Inteligencia: Propiedad de un **solo** agente	Inteligencia: Propiedad de **múltiples** agentes
Proceso cognitivo de un **solo** agente	Proceso **social** de **varios** agentes

MAS vs DAI clásico

- **DAI** (Distributed AI):
  - Un problema concreto se divide en problemas más pequeños (nodos). Estos nodos tienen un conocimiento común. Se proporciona el método solución.
- **MAS** (Multi-Agent System):
  - Varios agentes coordinan sus conocimientos y acciones. No se proporciona el método solución.

- Hoy día DAI se utiliza como sinónimo de MAS.

Agents vs. Objetos

- **Objetos**:
  - un estado (encapsulado): control sobre un estado interno
  - capacidades para el paso de mensajes a otros objetos
- Java:
  - Métodos privados y públicos.
  - Los objetos conoce su estado, pero no tiene control total sobre su comportamiento.
  - Un objeto no puede impedir que otros utilicen sus métodos públicos.

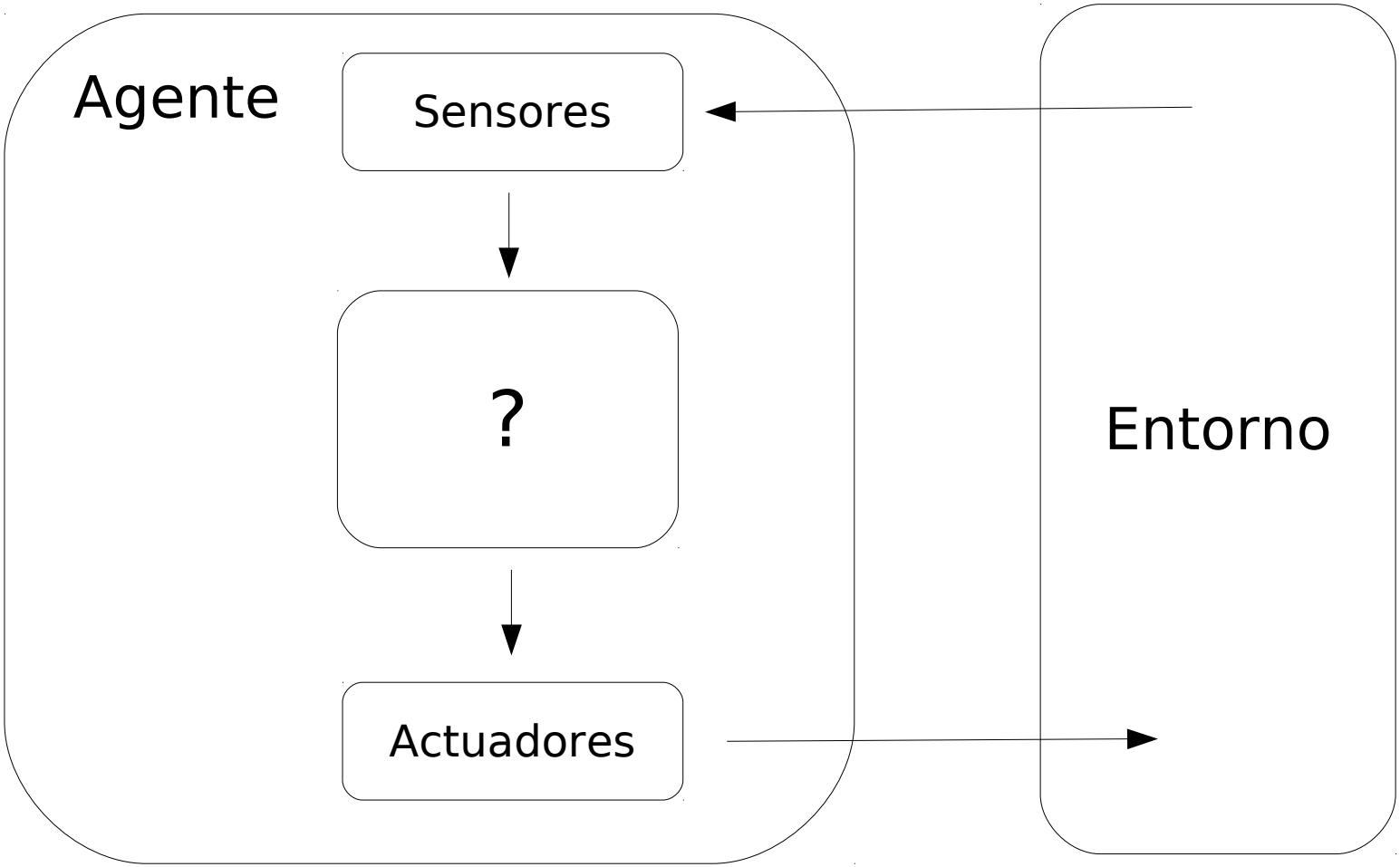
Agents vs. Objetos

- **Agentes**:
  - Agentes se comunican con otros agentes y les solicitan que ejecuten acciones por ellos.
  - Los objetos siempre hacen lo que se les pide, los agentes no.
  - En OO no hay analogía a ser reactivo, proactivo o social.
  - MAS son multi-hilo o multi-proceso: cada agente puede tener su hilo de ejecución
  - En OO sólo el sistema en su conjunto posee uno.

1.3 Arquitecturas de Agentes

- Arquitecturas reactivas
- Arquitecturas deliverativas
- Arquitecturas híbridas

1.3 Arquitecturas de Agentes



1.3 Arquitecturas de Agentes

- La Arquitectura:
  - Determina los mecanismos que utiliza el agente para reaccionar a estímulos, actuar, comunicarse, etc.
  - Especifica cómo es la estructura interna del agente: cómo se descompone en conjuntos de módulos que interactúan entre sí para lograr una funcionalidad
  - agrupa técnicas y algoritmos

1.3 Arquitecturas de Agentes

- (Stone & Veloso 1997) MAS Systems: A survey ...

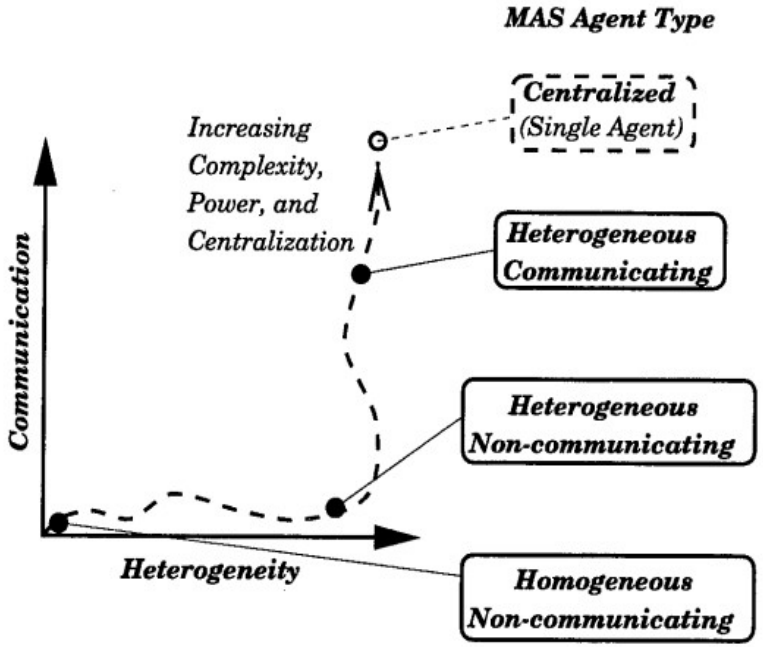


Figure 1: The major categories of the intrafield taxonomy and how they relate to the major dimensions. With full communication of internal state, a centralized system involving a single complex agent may result. Even though there are still multiple entities, they send their sensory perceptions and receive their actions from a central location: a single agent controls them all.

1.3 Arquitecturas de Agentes

- Escenario con un solo agente ...

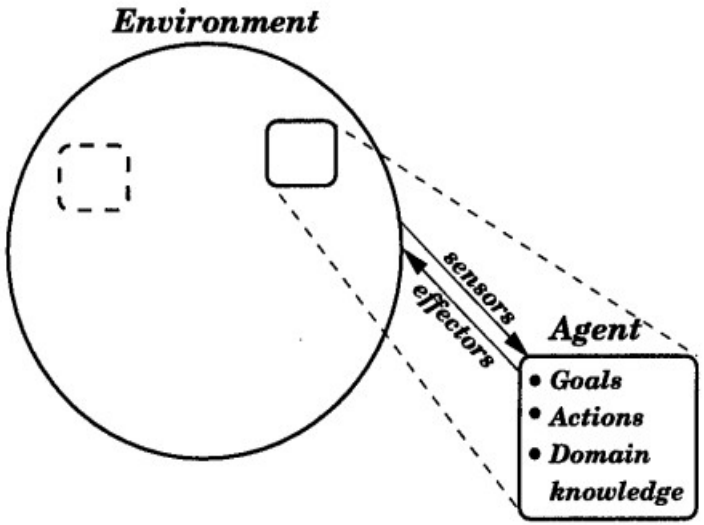


Figure 2: A general single-agent framework. The agent models itself, the environment, and their interactions. If other agents exist, they are considered part of the environment.

1.3 Arquitecturas de Agentes

- Escenario completo con múltiples agentes ...

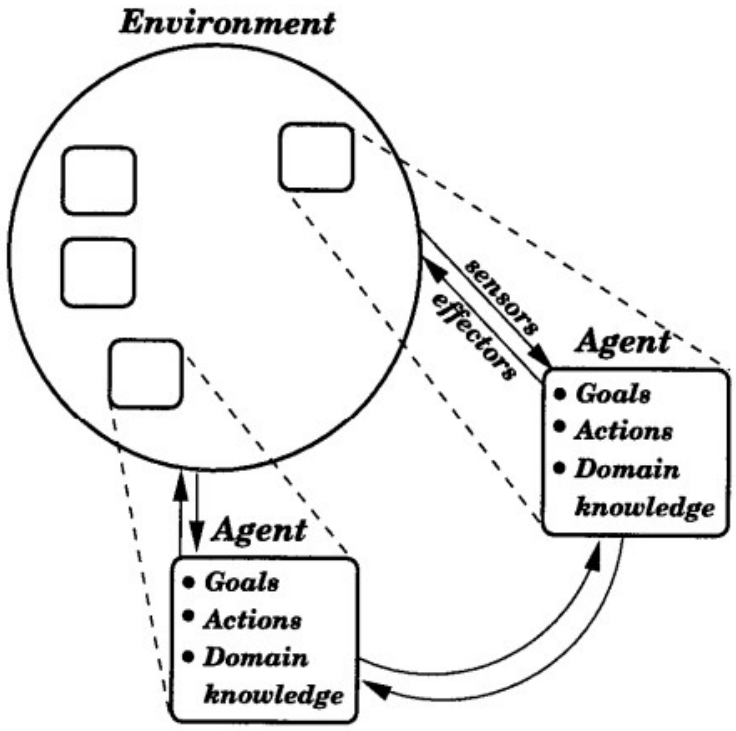
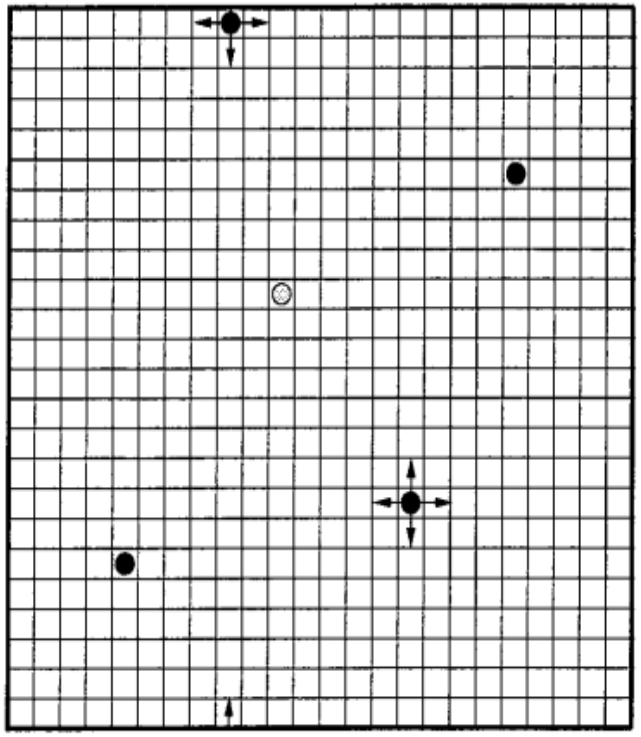


Figure 3: The fully general multiagent scenario. Agents model each other's goals and actions; they may also interact directly (communicate).

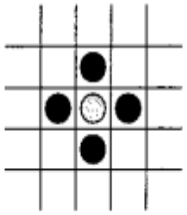
1.3 Arquitecturas de Agentes

- Escenario de caza con una presa y múltiples predadores ...



Orthogonal Game in a Toroidal World

Capture



- ***Predators see each other***
- ***Predators can communicate***
- ***Prey moves randomly***
- ***Prey stays put 10% of time***
- ***Simultaneous movements***

Figure 4: A particular instantiation of the pursuit domain. Predators are black and the prey is grey. The arrows on top of two of the predators indicate possible moves.

1.3 Arquitecturas de Agentes

- Escenario de caza con un solo agente ...

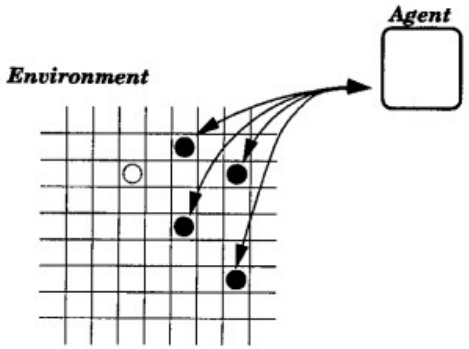


Figure 5: The pursuit domain with just a single agent. One agent controls all predators and the prey is considered part of the environment.

1.3 Arquitecturas de Agentes

- Escenario de caza con un múltiples agentes homogéneos pero sin comunicación ...

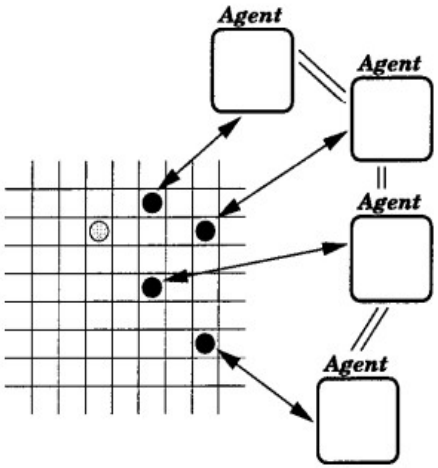


Figure 6: The pursuit domain with homogeneous agents. There is one identical agent per predator. Agents may have (the same amount of) limited information about other agents' internal states.

1.3 Arquitecturas de Agentes

- Escenario de caza con un múltiples agentes heterogéneos pero sin comunicación ...

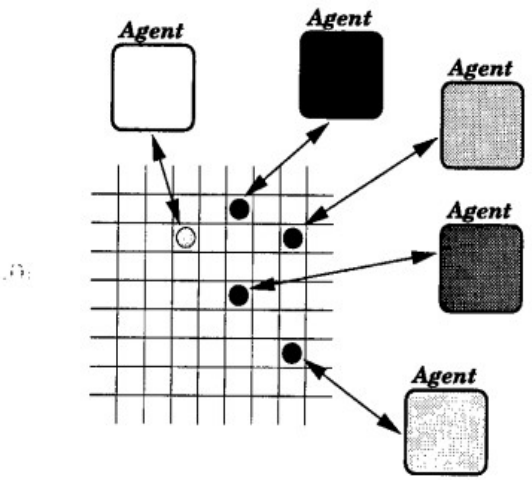


Figure 8: The pursuit domain with heterogeneous agents. Goals and actions may differ among agents. Now the prey may also be modeled as an agent.

1.3 Arquitecturas de Agentes

- Escenario de caza con un múltiples agentes heterogéneos pero con comunicación ...

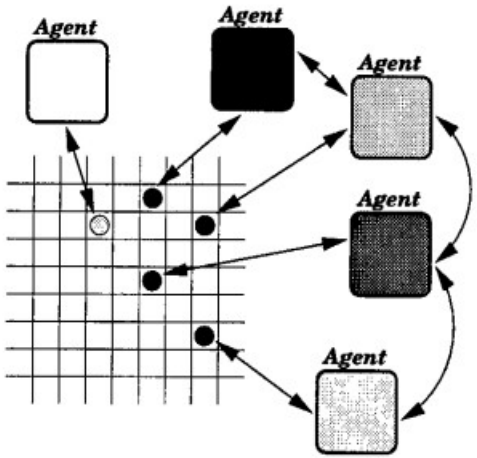


Figure 10: The pursuit domain with communicating agents. Agents can still be fully heterogeneous but now the predators can communicate with one another.

Arquitecturas Reactivas

- Idea:
  - El comportamiento inteligente surge de la interacción de los agentes con su entorno.
  - Emerge al combinar comportamientos e interacciones simples.

Arquitecturas Reactivas

- Arquitectura de subsunción (Brooks 1991)
  - La toma de decisiones se realiza a través de comportamientos (conductas) dirigidas a alcanzar un fin
  - Cada comportamiento es una acción individual.
  - Cada agente obtiene percepciones del entorno y las asocia a una acción a realizar
  - Los comportamientos se implementan mediante máquinas de estados finitos.
  - Las reglas son de la forma:
    - Situación -> Acción

Arquitecturas Reactivas

- Arquitectura de subsunción (Brooks 1991)
  - Pueden dispararse varios comportamientos simultáneamente. ¿Cómo elegir entre ellos?
  - Una jerarquía de subsunción permite priorizar los comportamientos situándolos en capas.
  - Las capas superiores representan comportamientos más generales.

Arquitecturas Reactivas

- Ejemplo: Exploración de un planeta.
  - Un lejano planeta contiene oro. Están disponibles varios vehículos autónomos. Las muestras deben ser llevadas a una nave espacial que aterrizó en el planeta. No se sabe dónde está el oro. Debido a la topografía del planeta no hay conexión entre los vehículos.
- Gradiente de campo
  - La nave espacial envía señales de radio.

Arquitecturas Reactivas

- Reglas de comportamiento

- (1) **IF** detecta un obstáculo **THEN** cambia de dirección
- (2) **IF** (muestras a bordo AND en la base) **THEN** solarlas
- (3) **IF** (muestras a bordo AND no en la base) **THEN** sigue el gradiente
- (4) **IF** detecta muestras **THEN** cógelas
- (5) **IF** true **THEN** toma un camino al azar

- Con el siguiente orden:
  - 1 < 2 < 3 < 4 < 5

Arquitecturas Reactivas

- Pros:
  - Simplicidad, economía (necesidades computacionales), computacionalmente tratable, robusta a fallos y elegante.
  - Respuesta inmediata, ...
- Cons:
  - Decisiones basadas en información local (con efectos globales)
  - Difícil diseñar agentes puramente reactivos que puedan aprender de la experiencia ...
  - La relación entre agentes, entornos y comportamiento no está totalmente clara ...
  - Agentes con ≤ 10 comportamientos son factibles. Pero cuantas más capas, más complicado es entender lo que está pasando.

Arquitecturas Deliverativas

- Idea:
  - Modelo (representación simbólica) del entorno, explícitamente representado.
  - Sistema de planificación, como mecanismos de razonamiento lógico basados en la concordancia de patrones y la manipulación simbólica.
  - Basadas en la teoría clásica de planificación:
    - Dado un estado inicial son capaces de generar planes para alcanzar el Estado objetivo.

Razonamiento

$$\frac{A \rightarrow B \quad A}{B}$$

Razonamiento

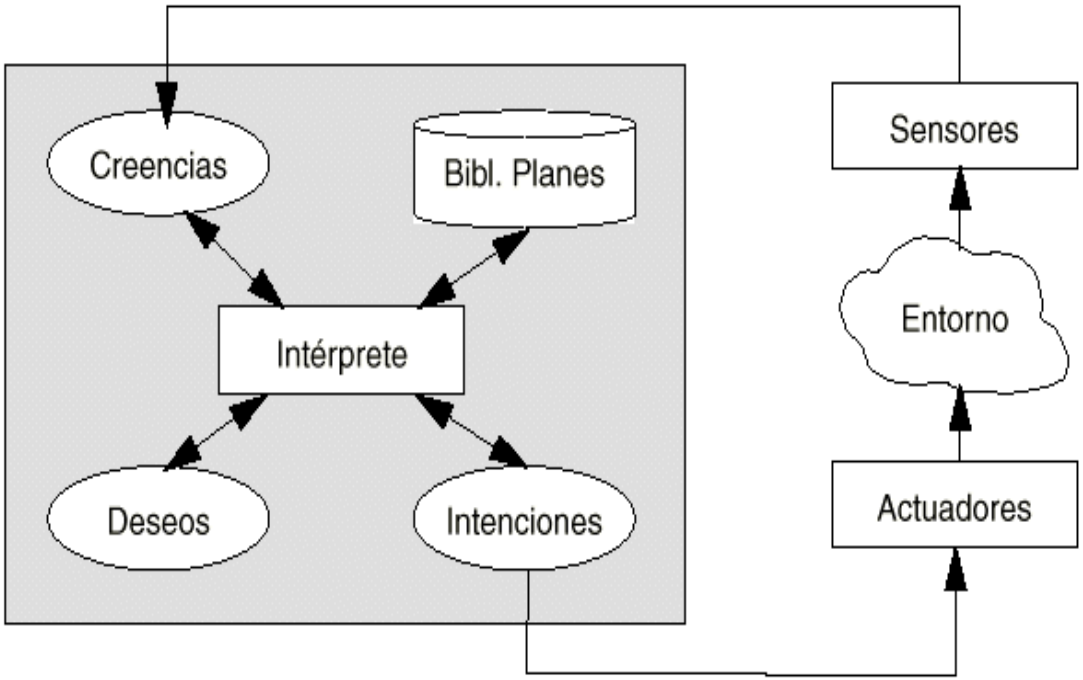
$$\frac{A \rightarrow B \quad A}{B}$$

$\frac{A \rightarrow B \quad A}{?}$	$\frac{? \quad A}{B}$	$\frac{A \rightarrow B \quad ?}{B}$
Deducción	Inducción	Abducción

Arquitecturas Deliverativas

- Arquitecturas **BDI** se basan en la suposición de que la mente (estado mental) de los agentes consiste en:
  - Creencias (**B**eliefs): lo que el agente cree que es verdad sobre el mundo (información).
  - Deseos (**D**esires): estado(s) del mundo que los agentes desean establecer (motivación).
  - Intenciones (**I**ntentions): lo que el agente realmente tiene la intención de hacer y cómo hacerlo (deliberación).
- El mundo para un agente son los otros agentes, el entorno, y el propio agente.

Arquitecturas Deliverativas



Arquitecturas Deliverativas

- BDI permite la interacción entre dos formas de razonamiento:
  - Basado en objetivos (medios para un fin)
  - Valora posibilidades en competencia
  - Y aborda el problema de los recursos limitados

Arquitecturas Deliverativas

- Basado en objetivos (medios para un fin)

- viene de la sub-campo de la IA que se ocupa de la planificación
- Dado: un estado inicial, un conjunto de estados objetivo (fines), así como una descripción de las acciones (medios o capacidades)
- Objetivo: encontrar una secuencia de acciones (plan) que va desde el estado final al estado final

Arquitecturas Deliverativas

- Razonamiento basado en objetivos
- Ejemplo:
  - Estado inicial:
    - estoy en mi casa, tengo una foto, tengo clavos, no tengo marco y estoy sin herramientas.
  - Estado objetivo:
    - la imagen está enmarcada y colgada en la pared
  - Plan:
    - 1. ir a la tienda de bricolaje
    - 2. adquirir un marco y un martillo
    - 3. ir a casa
    - 4. enmarcar la foto
    - 5. usar un martillo y clavos para colgar el cuadro en la pared

Arquitecturas Deliverativas

- Valoración de posibilidades en competencia

- Proviene de la teoría de decisión
- Dadas unas posibilidades en competencia
- Se valoran las posibilidades y se decide por una de ellas
- La selección se basa en la función de utilidad del agente teniendo en cuenta sus creencias (lo sabe el agente) y deseos (lo que quiere el agente)

Arquitecturas Deliverativas

- Valoración de posibilidades en competencia
- Ejemplo:
  - Deseo: disfrutar de una comida
  - Posibilidad 1: ir a Paco's
  - Posibilidad 2: ir a un restaurante de lujo
  - Creencias: Estoy sin fondos
  - Decisión: ir a Paco's

Aplicaciones en tiempo real (deliverativas)

1) El entorno es no determinista, es decir, en cada momento entorno puede evolucionar de varias maneras.
2) El sistema es no determinista, es decir, potencialmente en cada momento hay diferentes acciones a realizar.
3) El sistema puede tener varios objetivos diferentes al mismo tiempo.
4) Las mejores acciones/procedimientos que permiten alcanzar los objetivos dependen de la situación del entorno y son independientes del estado interno del sistema.
5) El entorno sólo puede ser detectado localmente.
6) La velocidad de deliveración y acciones estan limitados por la velocidad a la que evoluciona el entorno.

Aplicaciones en tiempo real (deliverativas)

- Las características:
  - 4) (la mejor acción depende de entorno de estado y es independiente del estado interior del sistema),
  - 1) (entorno no determinista), y
  - 5) (detección local) implican que
- es necesario que haya algún componente del sistema que puede representar la información acerca de la estado del mundo.

~> Creencias (Beliefs)!

Aplicaciones en tiempo real (deliverativas)

- Las características:
  - 3) (varios objetivos simultáneos) y
  - 5) (detección local) implican que
  - es necesario que el sistema también tenga información sobre los objetivos a cumplir.

~> Deseos (Desires)!

Aplicaciones en tiempo real (deliverativas)

- **Idea**: reconsiderar la elección de acciones en cada paso.
- **Dilema**: esto es potencialmente muy caro y la acción elegida, posiblemente, podría ser inválida cuando se selecciona.
- **Asunción**: es posible limitar la frecuencia de la revisión y lograr un equilibrio entre demasiada y o insuficiente reconsideración. Recuérdese la característica 6 (frecuencia razonable de cálculos y acciones).
- **Implicación**: es necesario incluir un componente del sistema que representa el curso de acción elegido actualmente.

~> Intenciones (Intentions)!

Bases de conocimiento (deliverativas)

- Conjunto de creencias:
  - Por lo general, almacenada en una base de creencias.
  - Ejemplo:
    - *Soy un estudiante de informática.*
    - *Estoy en mi cuarto curso, primer cuatrimestre.*
- Conjunto de objetivos:
  - Por lo general, almacenados en una base de objetivos.
  - Ejemplo:
    - *Quiero graduarme en informática.*
- Conjunto de planes:
  - Recetas de cómo llegar a la objetivos. Por lo general, de alguna manera estructurada, por ejemplo, acciones anidadas y almacenados en un base de planes.

Bases de conocimiento (deliverativas)

- Conjunto de creencias:
  - Por lo general, almacenada en una base de creencias.
    - *Soy un estudiante de informática.*
    - *Estoy en mi cuarto curso, primer cuatrimestre.*
- Conjunto de objetivos:
  - Por lo general, almacenados en una base de objetivos.
    - *Quiero graduarme en informática.*
- Conjunto de planes:
  - Recetas de cómo llegar a la objetivos. Por lo general, de alguna manera estructurada, por ejemplo, acciones anidadas y almacenados en un base de planes.
    - *Asistar a un montón de clases.*
    - *Realizar un montón de prácticas.*
    - *Superar un montón de exámenes.*

Bases de conocimiento (deliverativas)

- Lenguage de representación del conocimiento, e.g. Prolog

- Creencias (Beliefs):
  - estudio(yo, informatica).
  - curso(yo, 4), cuatrimestre(yo, 1).
- Deseos (Desires):
  - grado(yo, informatica).
- Plan:
  - [asistir(yo, TAIA), asistir(yo, ...),...]

Iteración de control del agente BDI v1

while true **do**

observar el mundo;

actualizar el modelo de mundo interno;

decidir qué intención lograr a continuación;

razonar para obtener un plan para la intención;

ejecutar el plan;

end

- Decidir: considerando cuidadosamente todas las opciones.
- Planificación: Una vez comprometido a algo, como llegar a la meta?
- Replanificación: ¿Qué pasa si durante la ejecución del plan, las cosas están funcionando fuera de control y el plan original falla?

Iteración de control del agente BDI v2

```
Set<Belief> beliefs = initBeliefBase();
while ( true ) {
    Percept percept = getNextPercept();
    beliefs = beliefRevision(beliefs, percept);
    Set<Intention> intentions = deliberation(beliefs);
    Plan plan = generatePlan(beliefs, intentions);
    execute(plan);
}
```

Iteración de control del agente BDI v2

- Estado interno del agente es una tripleta (B, D, I)
- Las *intenciones* son lo más importante.
- Las *creencias* y las *intenciones* generan *deseos*.
- Los *deseos* pueden ser incompatibles entre sí.
- Las *intenciones* se vuelven a calcular en base a la intenciones actuales, deseos y creencias.
- Las *intenciones* deberían persistir, normalmente.
- Las *creencias* se actualizan constantemente y por lo tanto generan nuevos *deseos*.
- De vez en cuando las *intenciones* deben ser reexaminadas.

Iteración de control del agente BDI v3

```
Set<Belief> beliefs = initBeliefBase();
Set<Intention> intentions = initIntentionBase();
while ( true ) {
    Percept percept = getNextPercept();
    beliefs = beliefRevision(beliefs, percept);
    Set<Desire> desires = findOptions(beliefs,intentions);
    intentions = filter(beliefs,desires,intentions);
    Plan plan = generatePlan(beliefs, intentions);
    execute(plan);
}
```

Iteración de control del agente BDI v3

- Ahora tenemos unas intenciones iniciales.
- La deliberación se ha dividido en dos componentes:

  1) Generar opciones (deseos).

  2) Filtrar las intenciones correctas.
- Las intenciones pueden estar en una pila (por ejemplo de prioridades).

- Pero, no hay manera de volver replanificar si algo sale mal!

Iteración de control del agente BDI v3

- Ahora tenemos unas intenciones iniciales.
- La deliberación se ha dividido en dos componentes:

  1) Generar opciones (deseos).

  2) Filtrar las intenciones correctas.
- Las intenciones pueden estar en una pila (por ejemplo de prioridades).

- Pero, no hay manera de volver replanificar si algo sale mal!

Iteración de control del agente BDI v4

```
Set<Belief> beliefs = initBeliefBase();
Set<Intention> intentions = initIntentionBase();
while ( true ) {
    Percept percept = getNextPercept();
    beliefs = beliefRevision(beliefs,percept);
    Set<Desire> desires = findOptions(beliefs,intentions);
    intentions = filter(beliefs,desires,intentions);
    Plan plan = generatePlan(beliefs,intentions);
    while( !plan.isEmpty() ) {
        Action head = plan.removeFirst();
        execute(head);
        percept = getNextPercept();
        beliefs = beliefRevision(beliefs,percept);
        if ( !sound(plan,intentions,beliefs) ) {
            plan = generatePlan(beliefs,intentions);
        }
    }
}
```

Iteración de control del agente BDI v4

- Pero ... qué es un plan?

- Un plan π es una lista de acciones primitivas. Ellos nos llevan, mediante su aplicación sucesiva, desde el estado inicial al estado objetivo.

Arquitecturas BDI

- Clases de agentes:
  - Audaces:
    - No se paran para reconsiderar las intenciones
    - Coste temporal y computacional bajo
    - Adecuados para entornos que no cambian rápidamente
  - Cautos:
    - Constantemente se paran para reconsiderar las intenciones
    - Explotan nuevas posibilidades
    - Adecuados para entornos que cambian rápidamente
  - Meta-control?
    - Que determine cuándo ser audaz o cauto?

Arquitecturas BDI

- Pros:
  - Modelo Intuitivo, es posible reconocer los procesos para decidir qué hacer y cómo hacerlo.
  - Descomposición funcional, que determina la clase de subsistemas necesarios para crear el agente
- Cons:
  - La mayor dificultad, como siempre, es saber cómo implementar estas funciones eficientemente.
  - Difícil equilibrar una conducta del agente que tenga al mismo tiempo iniciativa y reactividad

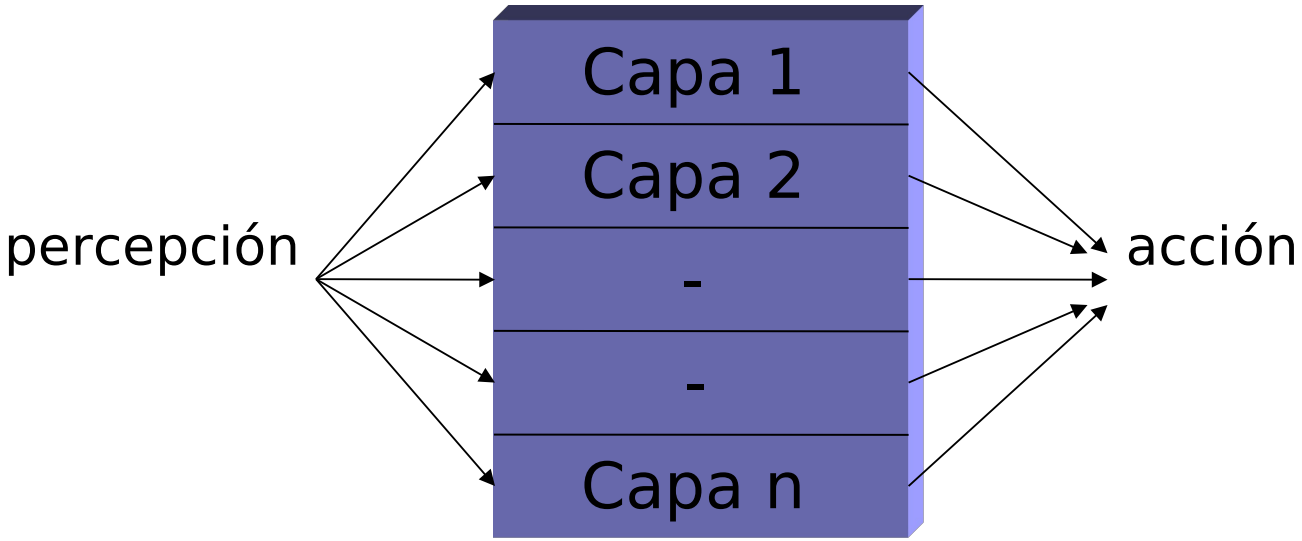
Arquitecturas Híbridas

- Arquitecturas formadas por dos o más subsistemas:
- Reactivo:
  - Para Procesar los estímulos que no necesitan deliberación.
- Deliverativo:
  - Modelo simbólico del mundo
  - Genera planes: determina acciones a realizar para satisfacer los objetivos locales y cooperativos de los agentes

- Estructura por capas: *Horizontal* y *Vertical*

Arquitecturas Híbridas

- Estructura por capas Horizontal
  - Cada capa esta directamente conectada con los sensores y los actuadores
  - Contribuye con sugerencias a la acción de actuar



Arquitecturas Híbridas

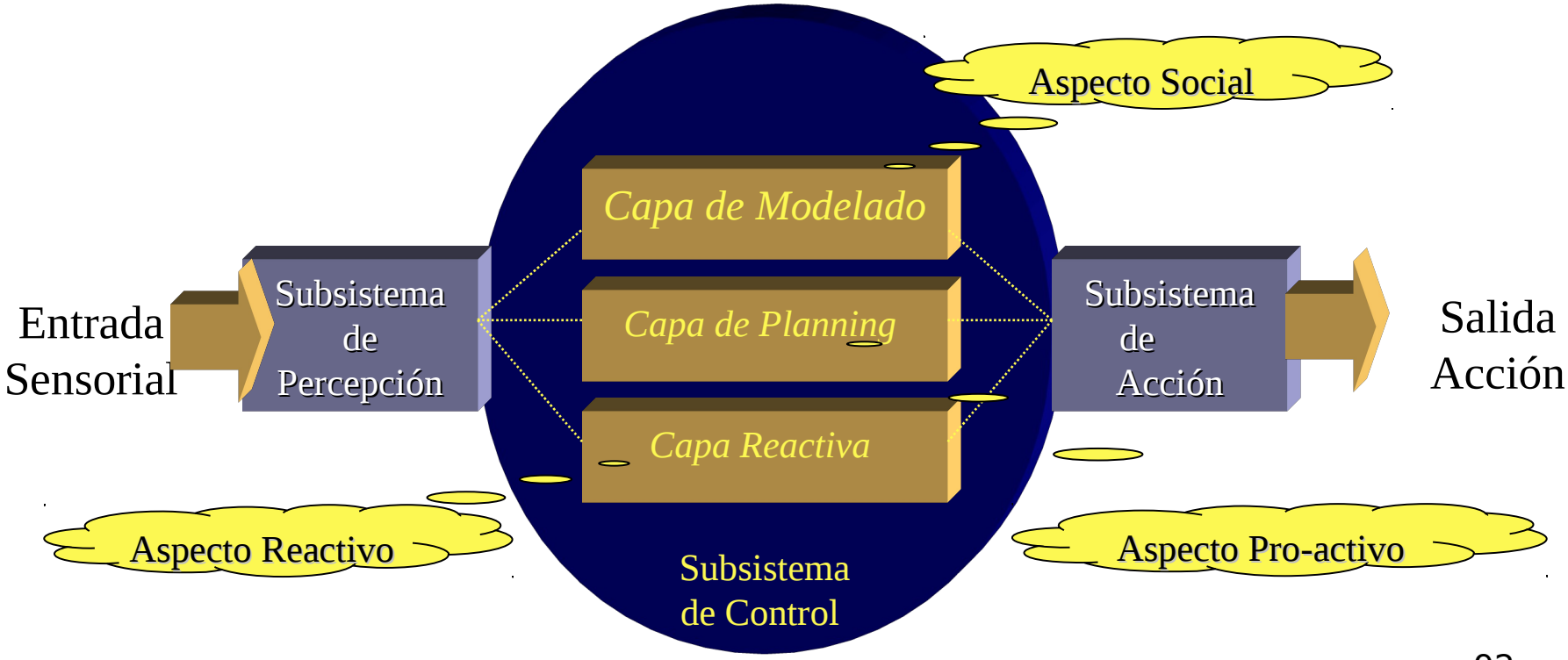
- Estructura por capas Horizontal

- Pros:
  - Simplicidad, n comportamientos diferentes -> n capas.
- Cons:
  - Coherencia? función mediadora que decide qué capa tiene el control del agente,
    - Asegura la consistencia,
    - Cuello de Botella
    - n capas con m posibles acciones ->

      m^n interacciones a considerar!

Arquitecturas Híbridas

- Ejemplo de arquitectura Horizontal:
  - TOURINGMACHINES (Ferguson, 1992)
  - Tres capas horizontales más un módulo de control

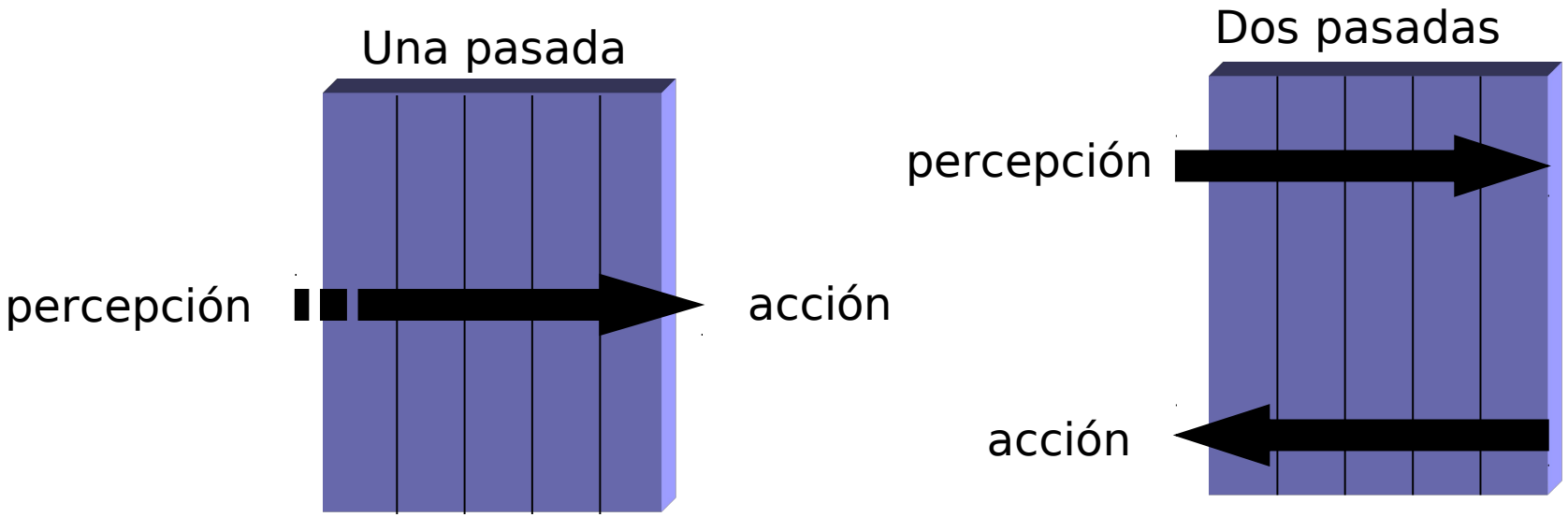


Arquitecturas Híbridas

- Ejemplo de arquitectura Horizontal:
  - TOURINGMACHINES (Ferguson, 1992)
  - Capa reactiva:
    - respuestas más o menos inmediatas a los cambios del entorno, implementada con *reglas situación-acción*
  - Capa de planificación:
    - representa la iniciativa del agente, contiene librería de esqueletos de planes, llamados *esquemas*. Planes estructurados para decidir qué hacer.
  - Capa de modelizado:
    - representa las entidades del entorno
  - Sistema de control:
    - decide qué capa tiene el control sobre el agente para evitar conflictos, implementado con reglas de control que pueden suprimir las entradas y inhibir las salidas

Arquitecturas Híbridas

- Estructura por capas Vertical
  - Los sensores y los actuadores están conectados con una capa



Arquitecturas Híbridas

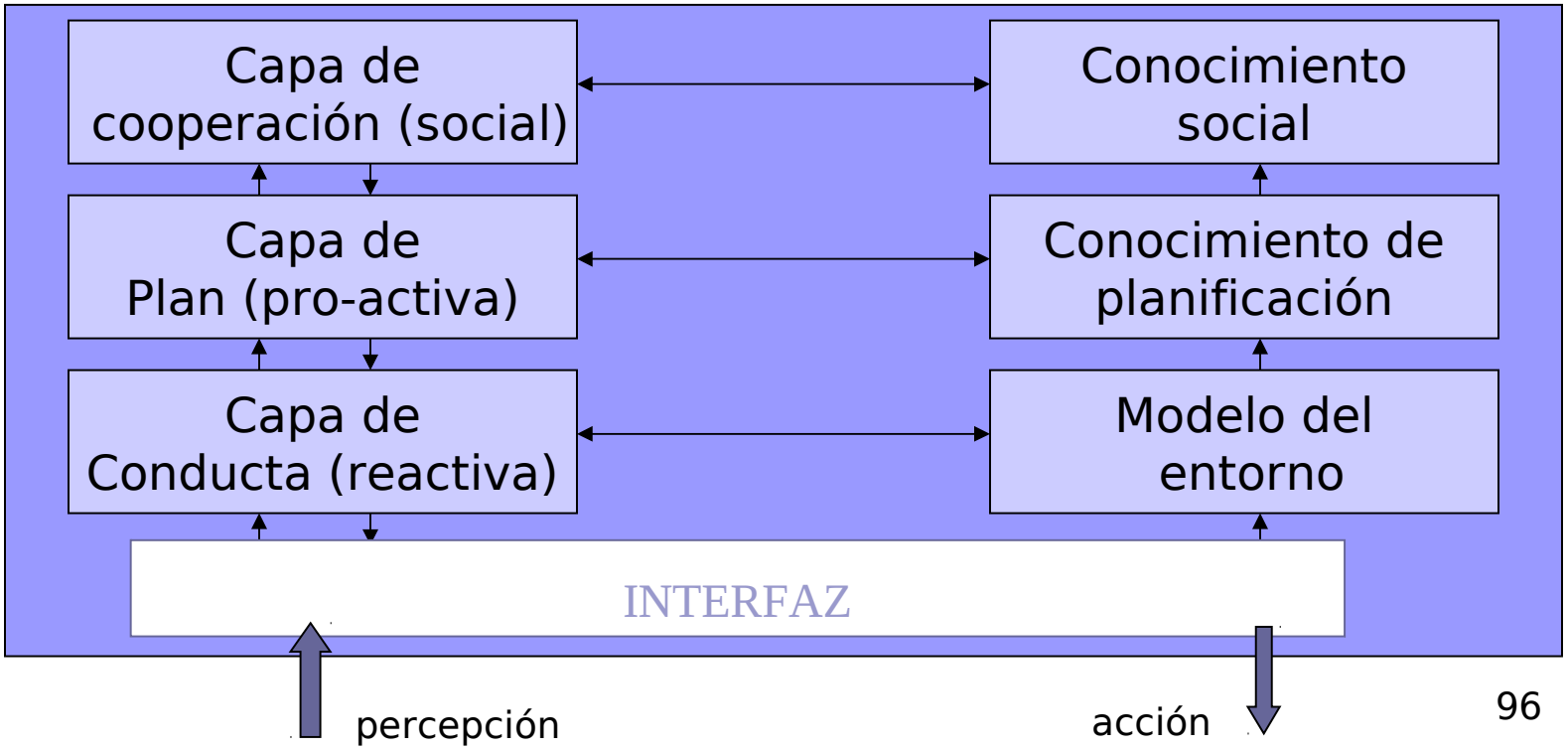
- Estructura por capas Vertical

- Pros:
  - Optima para equilibrar las diferentes conductas del agente (reactividad, iniciativa)
- Cons:
  - Falta de claridad y flexibilidad
  - n-1 interfaces entre capas con m posibles acciones

    $m^2 * (n-1)$ interacciones a considerar

Arquitecturas Híbridas

- Ejemplo de arquitectura Vertical:
  - INTERRAP (Muller, 1997)
  - Tres capas verticales, cada capa tiene su base de conocimiento, dos pasadas



Arquitecturas Híbridas

- Ejemplo de arquitectura Vertical:
  - INTERRAP (Muller, 1997)
  - Conocimiento social:
    - representa los planes y las acciones de otros agentes en el entorno
  - Conocimiento de planificación:
    - representa los planes y las acciones del propio agente
  - Modelo del entorno:
    - informaciones sobre el entorno
  - Interacción entre las capas:
    - Activación desde abajo hacia arriba
    - Ejecución desde arriba hacia abajo

Referencias

- TAIA 2012-2013. Maite Urretavizcaya. Grupo Galan. LSI. 2013.

- Brooks, R. A. (1991). Intelligence without representation. Artificial intelligence, 47(1), 139-159.
- Ferguson, I. A. (1992). TouringMachines: An architecture for dynamic, rational, mobile agents. Cambridge CB2 3QG, England: University of Cambridge, Computer Laboratory.
- Müller, J. P. (1997). A cooperation model for autonomous agents. In Intelligent Agents III Agent Theories, Architectures, and Languages (pp. 245-260). Springer Berlin Heidelberg.
- Russell S. and Norvig P. (2010). Artificial Intelligence: A Modern Approach, 3rd Edition. Pearson.
- Stone, P., & Veloso, M. (2000). Multiagent systems: A survey from a machine learning perspective. Autonomous Robots, 8(3), 345-383.
- Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: Theory and practice. Knowledge engineering review, 10(2), 115-152.

Técnicas Avanzadas de Inteligencia Artificial

Curso 2013-2014

German Rigau

german.rigau@ehu.es

http://adimen.si.ehu.es/~rigau

Grado en Ingeniería en Informática /
Ingeniería en Informática