Lecture 2: Agent Architectures

Multi-Agent Systems

Universitat Rovira i Virgili

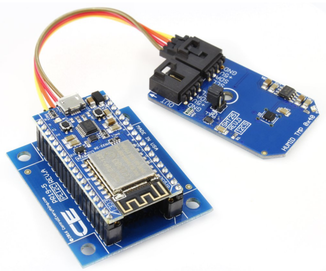
MESIIA – Master’s Degree in Computer Security Engineering and Artificial Intelligence
MAI – Master's Degree in Artificial Intelligence

Outline

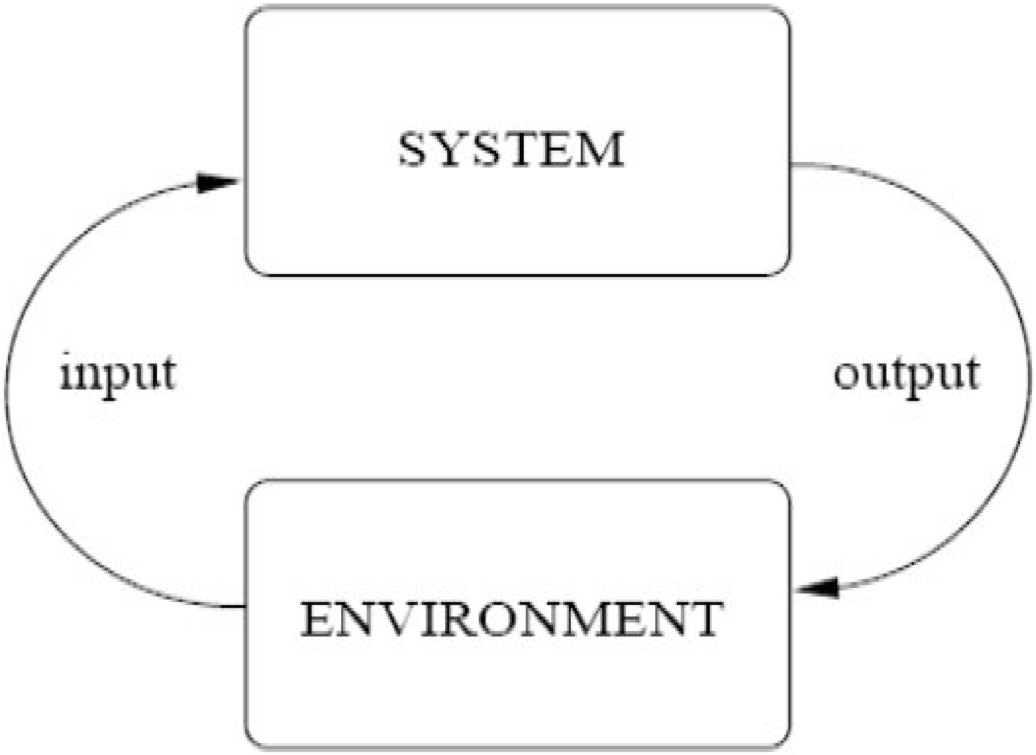
1. Intelligent agent - definition
2. Kinds of environments
3. Agent architectures:
    1. Reactive
    2. Deliberative
    3. Hybrid

1. Intelligent agent - definition

- "An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors”
- "Autonomous agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed"
- “An autonomous agent is a system situated within and part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to affect what it senses in the future”



Basic abstract view of an agent



Properties

- An agent has to be able to react (adapt its behaviour) in an appropriate way to the dynamic changes in its "environment"
  - Other computational agents
  - Human users
  - External information sources (e.g., sensors)
  - Physical objects (e.g., robots)
  - World Wide Web
  - …
- This is one of several properties that an intelligent agent should have … [more on that next week]

2. Kinds of environments

- Accessible vs inaccessible
  - An accessible environment is one in which the agent can obtain complete, accurate, up-to-date information about the environment's state
  - Most moderately complex environments (including, for example, the everyday physical world and the Web) are inaccessible
  - The more accessible an environment is, the simpler it is to build agents to operate in it

2. Kinds of environments

- Deterministic vs non-deterministic
  - A deterministic environment is one in which any action has a single guaranteed effect — there is no uncertainty about the state that will result from performing an action
  - The physical world can to all intents and purposes be regarded as non-deterministic
  - Non-deterministic environments present greater problems for the agent designer

2. Kinds of environments

- Episodic vs non-episodic
  - In an episodic environment, the performance of an agent is dependent on a number of discrete, independent episodes, with no link between the performance of an agent in different scenarios
  - Episodic environments are simpler from the agent developer's perspective because the agent can decide which action to perform based only on the current episode — it does not need to reason about the interactions between different episodes

2. Kinds of environments

- Static vs dynamic
  - A static environment is one that can be assumed to remain unchanged except by the performance of actions by the agent
  - A dynamic environment is one that has other processes operating on it, and which hence changes in ways beyond the agent's control
  - The physical world and the Web are highly dynamic environments
  - It is hard to design and implement agents in dynamic environments

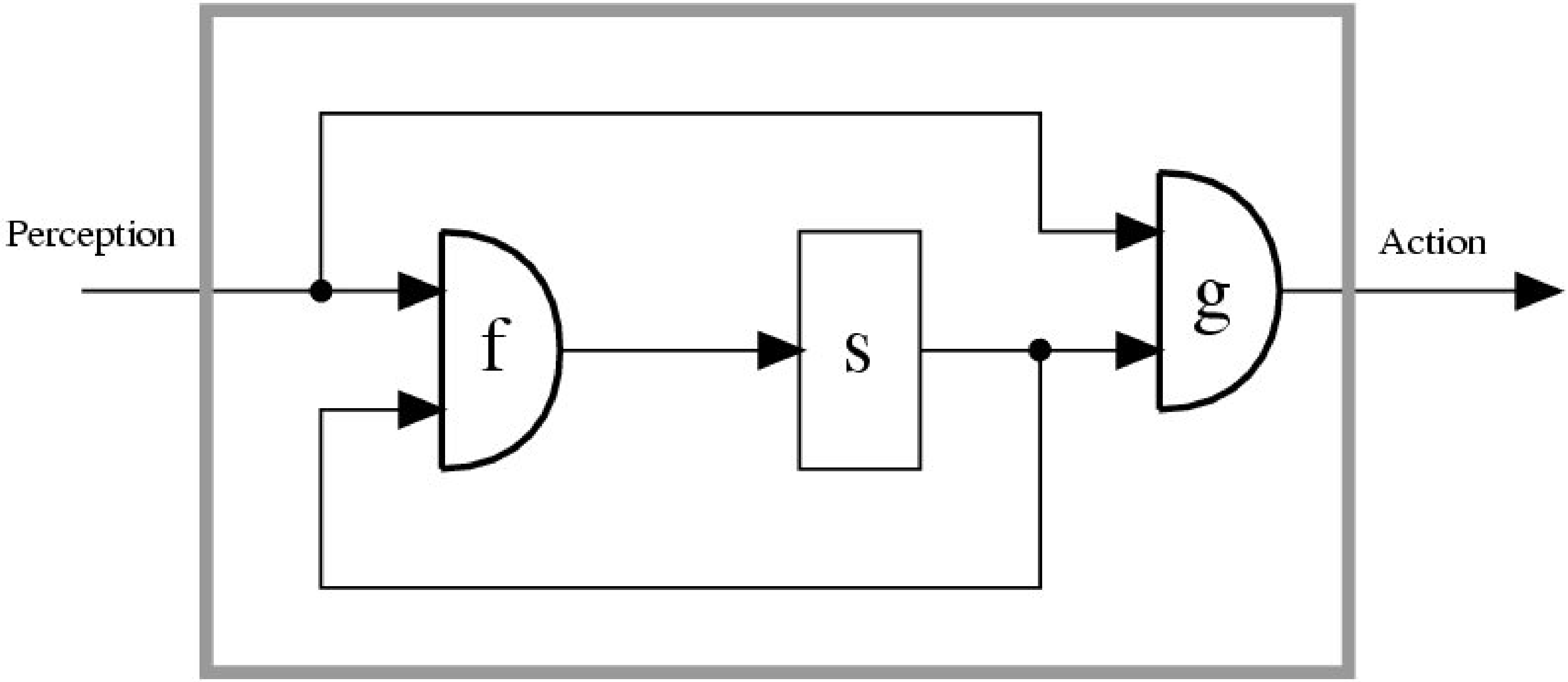
2. Kinds of environments

- Discrete vs continuous
  - An environment is discrete if there is a fixed, finite number of actions and percepts in it
  - The real world is a continuous environment
  - Discrete environments are much simpler than continuous ones

3. Agent architectures

- An architecture proposes a particular methodology for building an autonomous agent
  - How the construction of the agent can be decomposed into the construction of a set of component modules
  - How these modules should be made to interact
- These two aspects define how the sensor data and the current internal state of the agent determine the actions (effector outputs) and the future internal state of the agent

From perception to action



f = state update function
s = internal state
g = output function

Main kinds of agent architectures

- Reactive architectures
  - Focused on fast reactions/responses to changes detected in the environment
- Deliberative architectures (symbolic)
  - Focused on long-term planning of actions, centred on a set of goals
- Hybrid architectures
  - Combining a reactive side and a deliberative side

Example: Reactive vs Deliberative

- Robot that has to reach a certain point
  - Reactive
    - Sensor in the front of the robot
    - Without any representation of the environment
      - Example: robot changes movement right/left when sensor detects obstacle or it reaches a crossroad. Minimal computation based on current location and destination point
  - Deliberative
    - Explicit representation of the environment (map)
    - Planning procedure that finds the minimal route between the current position and the destination
      - High computational cost
      - Optimal route (in static environments)
      - Possible dynamic re-plannings needed

3.1. Agent architectures: Reactive – basic ideas

- Reactive agents have
  - At most a very simple internal representation of the world
  - A tight coupling of perception and action
  - Behaviour-based paradigm
  - Intelligence is a product of the interaction between an agent and its environment

Classical example: ant colony

- A single ant has very little intelligence, computing power or reasoning abilities
- The union of a set of ants and the interaction between them allows the formation of a highly complex, structured and efficient system

Swarm intelligence

- Agents and virtual swarms
- Example: morphogenesis

Reactive agents – Main characteristics

- Emergent functionality
  - Simple agents
  - Simple interaction
  - Complex behaviour patterns appear as a result of the dynamic interactions
  - The global behaviour of the system is not specified a priori
    - Dynamic movement of robots, depending on obstacles

Reactive agents – Main characteristics

- Task decomposition
  - Agents composed of autonomous modules
  - Each module manages a given task
    - Sensor, control, computations
  - Minimal, low-level communication between modules
- There isn't any world global model
- There isn't any "planning/controller/coordinator agent"

Reactive agents – Main characteristics

- Raw data
  - Basic data from sensors
  - There isn't any complex symbolic management of data as in classical AI
    - Refusal of the *Hypothesis of the physical symbols system* [basic pillar of symbolic AI]
      - "Intelligent behaviour can only be obtained in symbol-processing systems"

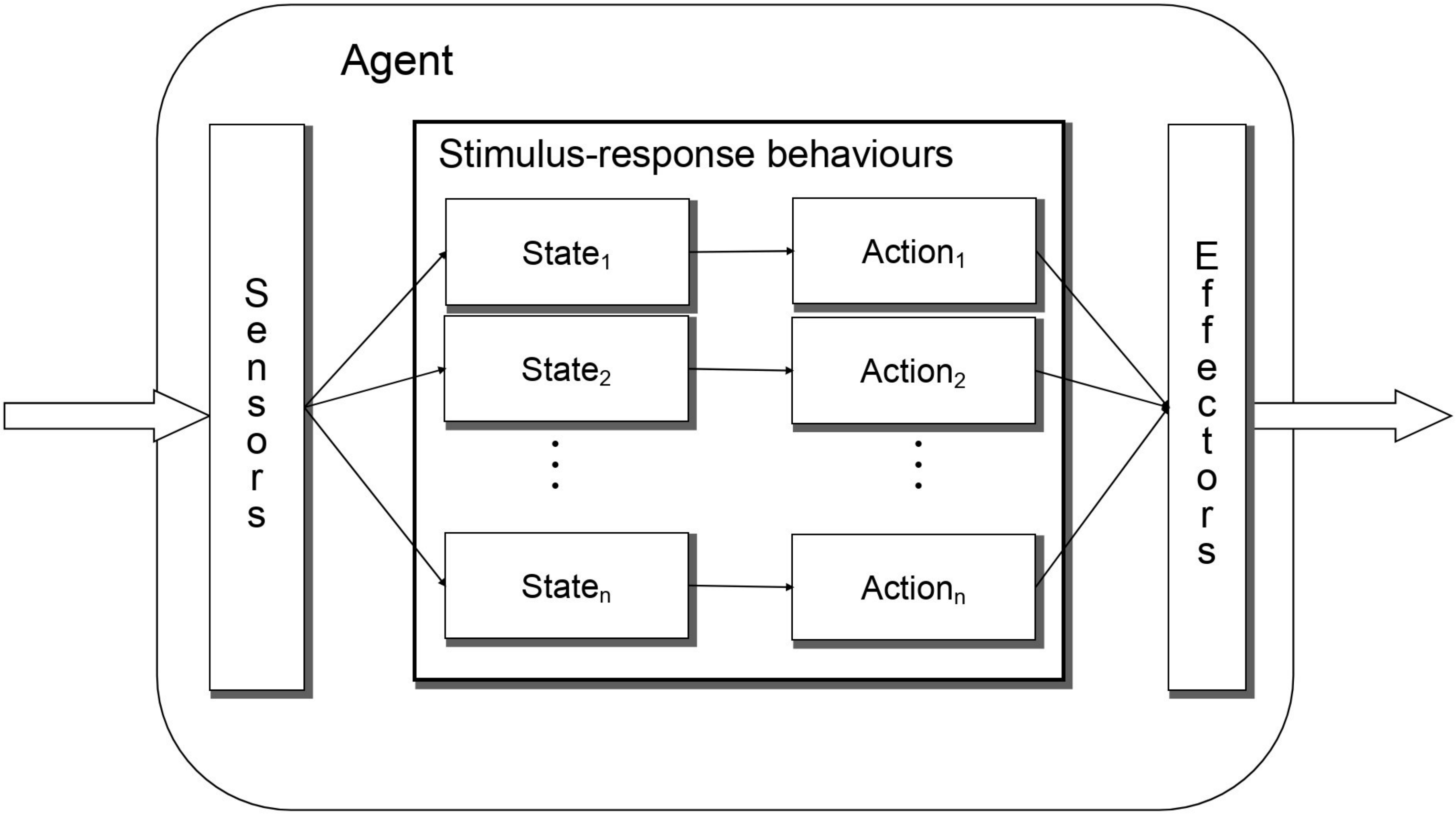
Reactive agents – Basic concept

- Each behaviour continually maps perceptual input to action output
- Reactive behaviour: action rules: $S \rightarrow A$

  where S denotes the states of the environment, and A the primitive actions the agent is capable of performing
- Example:

$$action\,(s) = \begin{cases} Heater\ off, & if\ temperature\ is\ over\ 15 \\ Heater\ on, & otherwise \end{cases}$$

Basic schema of reactive architecture



Rodney Brooks

Director of the Computer Science and Artificial Intelligence Lab (MIT) 1997-2007

- Roomba’s i-robot father

Brooks refutal of symbolic AI

- Brooks put forward three theses:
  1. Intelligent behaviour can be generated *without explicit representations* of the kind that symbolic AI proposes
  2. Intelligent behaviour can be generated *without explicit abstract reasoning* of the kind that symbolic AI proposes
     - Reduced computation on sensor-like data
  3. Intelligence is an *emergent* property of certain complex systems. Intelligence is 'in the eye of the beholder'; it is not an innate, isolated property

Brooks – key ideas

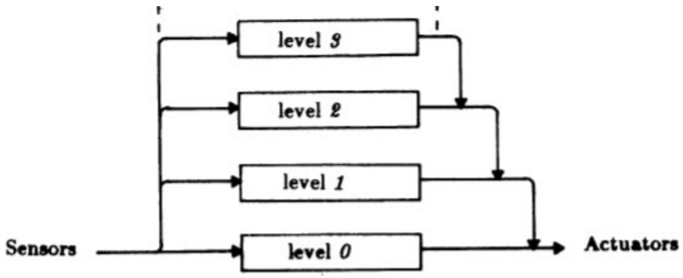
- Situatedness: ‘Real’ intelligence is situated in the world
  - The world is its best model
  - The world is always up-to-date
  - A model is an abstraction, a simplification of the world, considering a particular set of characteristics and disregarding others
- Embodiment: ‘Real’ intelligence requires a physical body, and cannot be found in disembodied systems such as theorem provers or expert systems
  - These are called Physical Agents

Brooks – subsumption hierarchy

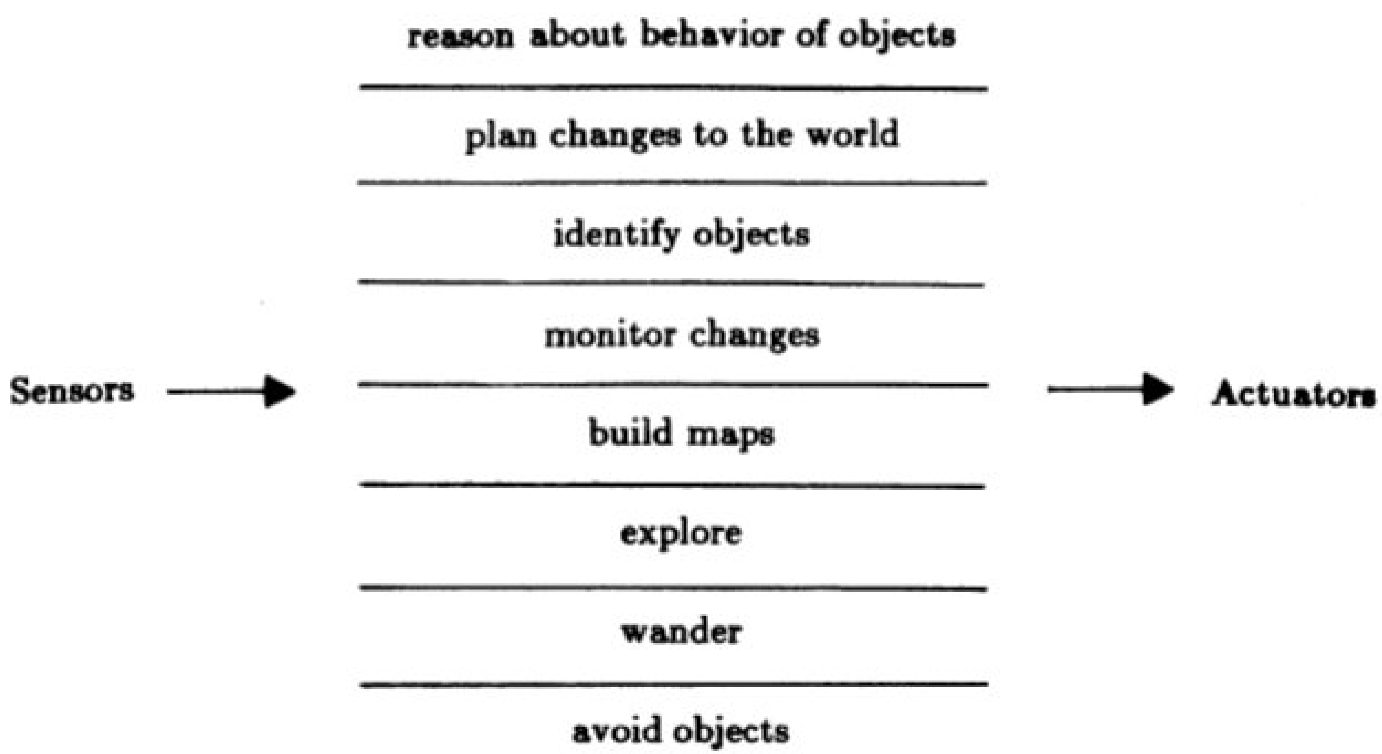
- To illustrate his ideas, Brooks built some systems based on his *subsumption architecture*
- A subsumption architecture is a hierarchy of task-accomplishing *behaviours*
- Each behaviour is a rather simple rule-like structure
- Each behaviour 'competes' with others to exercise control over the agent, as different behaviours may be applicable at the same time

Behaviour layers

- Lower layers represent primitive kinds of behaviour (such as avoiding obstacles)
- Higher layers represent more complex behaviours (e.g., identifying an object)
- Lower layers have precedence over layers further up the hierarchy
- The resulting systems are, in terms of the amount of computation they do, *extremely* simple
- Some of the robots do tasks that would be impressive if they were accomplished by symbolic AI systems

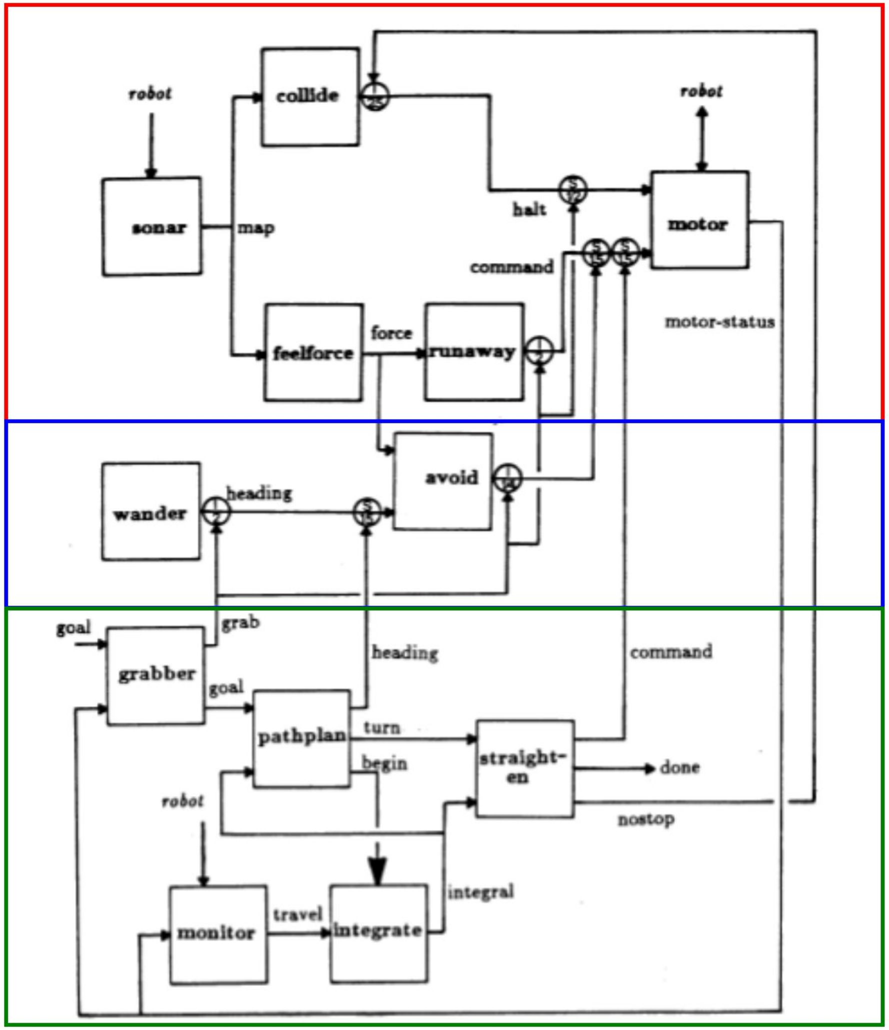


Decomposition Based on Task-Achieving Behaviours



From Brooks, “A Robust Layered Control System for a Mobile Robot”, 1985

Levels 0, 1 and 2 Control Systems

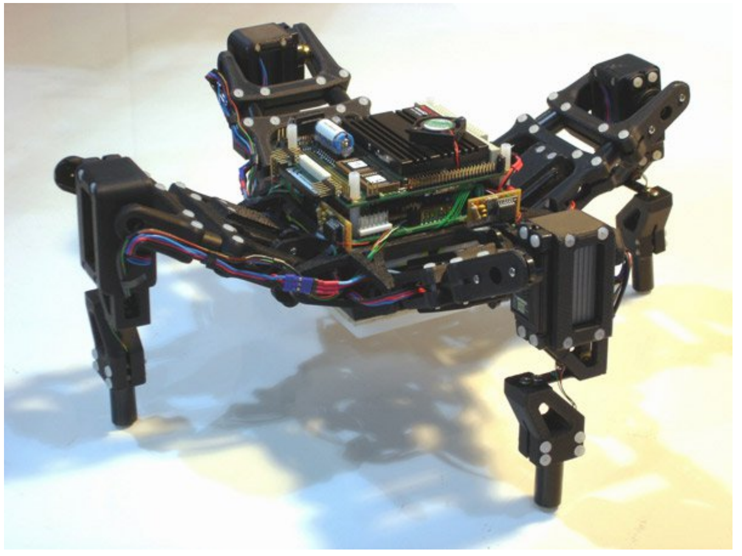


Avoid collisions

Random movement

Pick up objects, monitor, explore

From Brooks, “A Robust Layered Control System for a Mobile Robot”, 1985



[Some robots from MIT Lab]

Example: Luc Steels' Mars Explorer

- Steels' Mars explorer system, using the subsumption architecture, achieves near-optimal cooperative performance in simulated 'rock gathering on Mars' domain:
  The objective is to explore a distant planet, and in particular, to collect sample of a precious rock. The location of the samples is not known in advance, but it is known that they tend to be clustered

Steels' Mars Explorer Rules

- For individual (non-cooperative) agents, the lowest-level behaviour, (and hence the behaviour with the highest "priority") is obstacle avoidance:

  if detect an obstacle *then* change direction (1)

- Any samples carried by agents are dropped back at the mother-ship:

  if carrying samples *and* at the base
  then drop samples (2)

- Agents will collect samples they find:

  if detect a sample *then* pick sample up (3)

Steels' Mars Explorer Rules

- Agents carrying samples return to the mother-ship:

  if carrying samples and *not* at the base
  then move towards base and drop 2 crumbs (4)

- Agents can detect trails of crumbs:

  if not(carrying samples) and detect crumb

  then pick up 1 crumb and move away from base (5)

- An agent with "nothing better to do" will explore randomly:

  if true *then* move randomly (6)

Simulation

Advantages of Reactive Agents

- Simplicity of individual agents
- Flexibility, adaptability
  - Ideal in very dynamic and unpredictable environments
- Low computational cost
  - Avoiding complex planning/reasoning procedures
  - Avoiding continuous model update
- Robustness against failure
  - No central planning component (e.g., ant colony)
- Elegance

Limitations of Reactive Agents

- Agents without environment models must have sufficient information available from the local environment
- If decisions are based on *local* environment, how can we take into account *non-local* information?
  - “Short-term” view
- No long-term planning capabilities
- Limited applicability
  - Games, simulations, basic robots (insects)

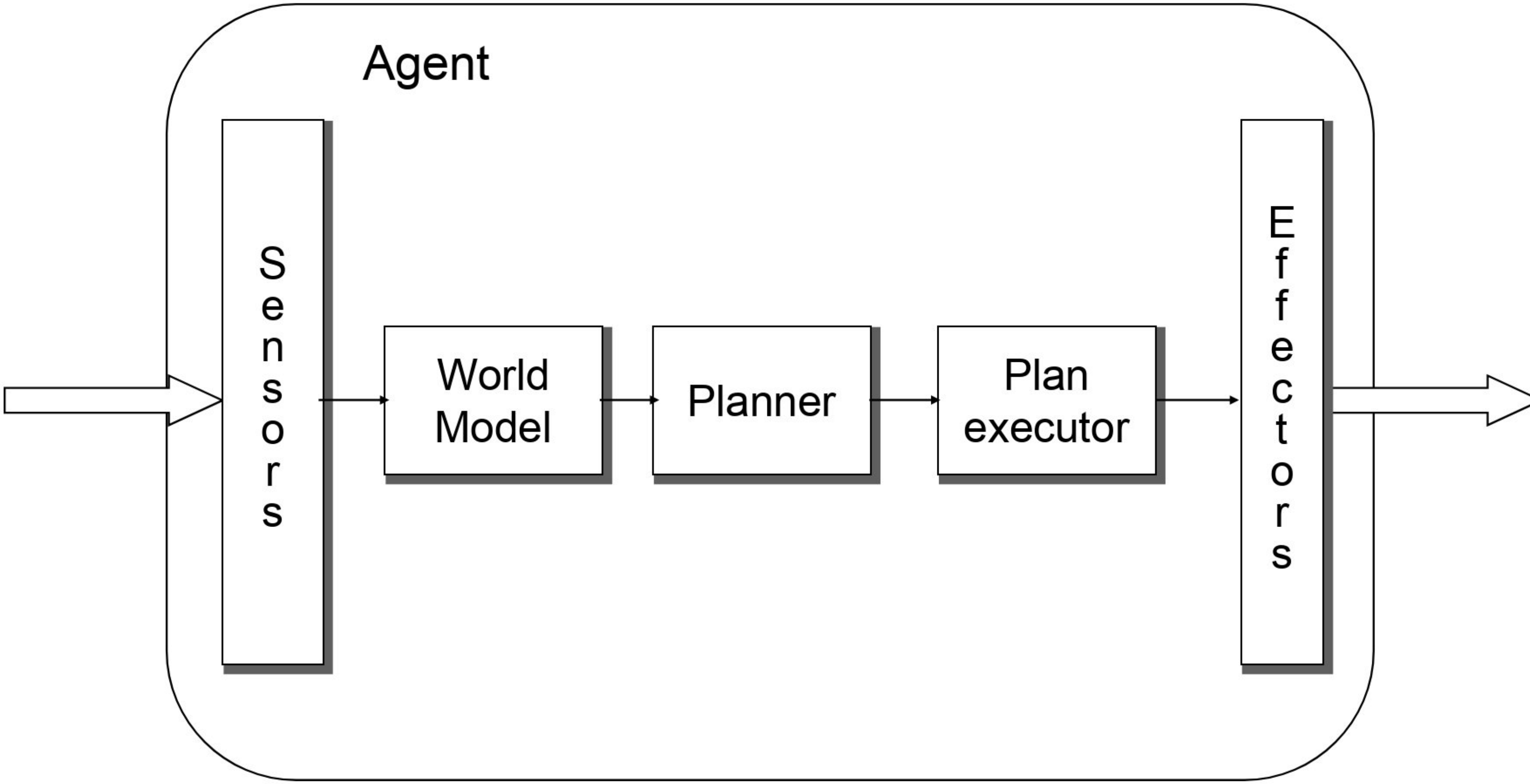
Limitations of Reactive Agents

- Difficult to make reactive agents that learn
  - Dynamic evolution of rules?
- Since behaviour emerges from component interactions plus environment, it is hard to see how to *engineer* specific agents (no principled methodology exists)
- It is hard to engineer agents with large numbers of behaviours as the dynamics of interactions become too complex to understand

3.2. Agent architectures: Deliberative

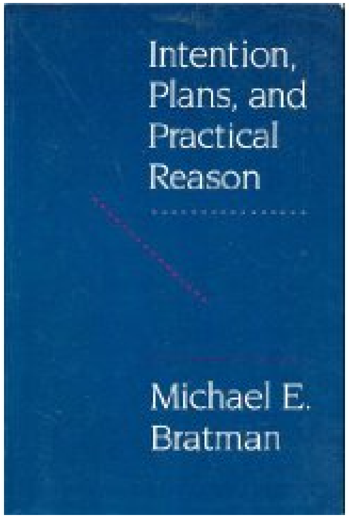
- Explicit symbolic model of the world
- Decisions are made via logical reasoning, based on pattern matching and symbolic manipulation
- Sense-plan-act problem-solving paradigm of classical AI planning systems

Basic schema of deliberative architecture



Belief-Desire-Intention (BDI) model

- A theory of practical reasoning
- Originally developed by Michael E. Bratman in his book ”*Intentions, Plans, and Practical Reason*”, (1987)
- Concentrates in the roles of the intentions in practical reasoning



Human practical reasoning

- Human practical reasoning consists of two activities:
  - *Deliberation*, deciding *which* state of affairs we want to achieve
    - the outputs of deliberation are *intentions*
  - *Means-ends reasoning*, deciding *how* to achieve these states of affairs
    - the outputs of means-ends reasoning are *plans*

Belief-Desire-Intention paradigm

- Beliefs:
  - Agent's view of the environment/world

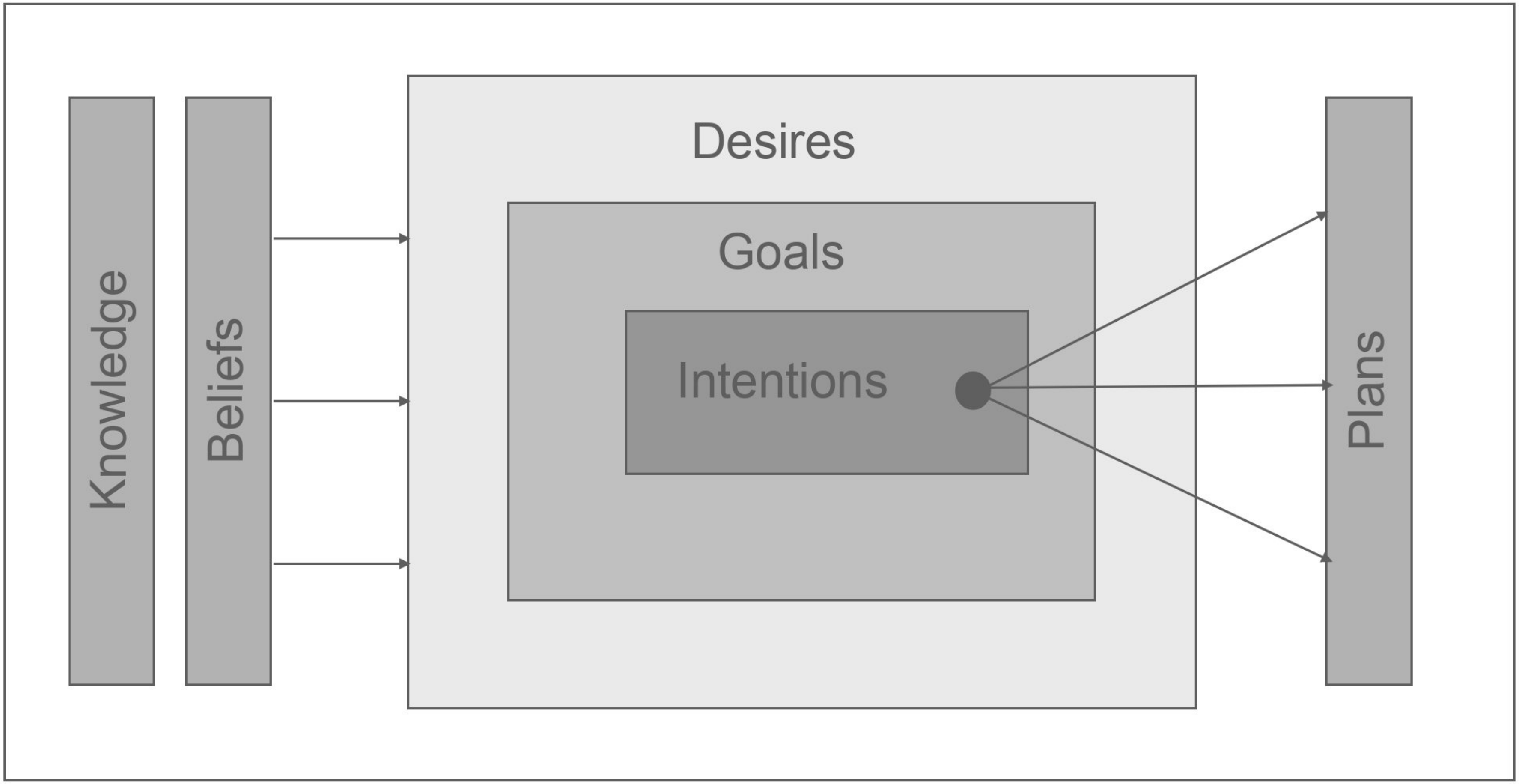
- Desires:
  - Follow from the beliefs. Desires can be unrealistic and inconsistent
- Goals:
  - A subset of the desires. Realistic and consistent. Determine potential processing.
- Intentions:
  - A subset of the goals. A goal becomes an intention when an agent decides to commit to it (e.g., by assigning priorities to goals)

- Plans:
  - Sequences of actions that are needed to achieve the intentions, given the agent's beliefs

Human practical plans

- In BDI implementations plans usually have:
  - *Name*
  - *Goal:* invocation condition that is the triggering event for the plan
  - *Pre-condition list:* list of facts which must be true for the plan to be executed
  - *Delete list:* list of facts that are no longer true after the plan is performed
  - *Add list:* list of facts made true by executing the actions of the plan
  - *Body:* list of actions

Belief-Desire-Intention architecture



Intention is choice with commitment (Cohen & Levesque)

- Intentions (& plans) enable the agent to be goal-driven rather than event-driven
- By committing to intentions the agent can pursue long-term goals
- An autonomous agent should act on its intentions, not in spite of them
  - Adopt intentions that are feasible
  - Drop the ones that are not feasible
  - Keep (or commit to) intentions, but not forever
  - Discharge those intentions believed to have been satisfied
  - Alter intentions when relevant beliefs change

Using plants to constrain reasoning

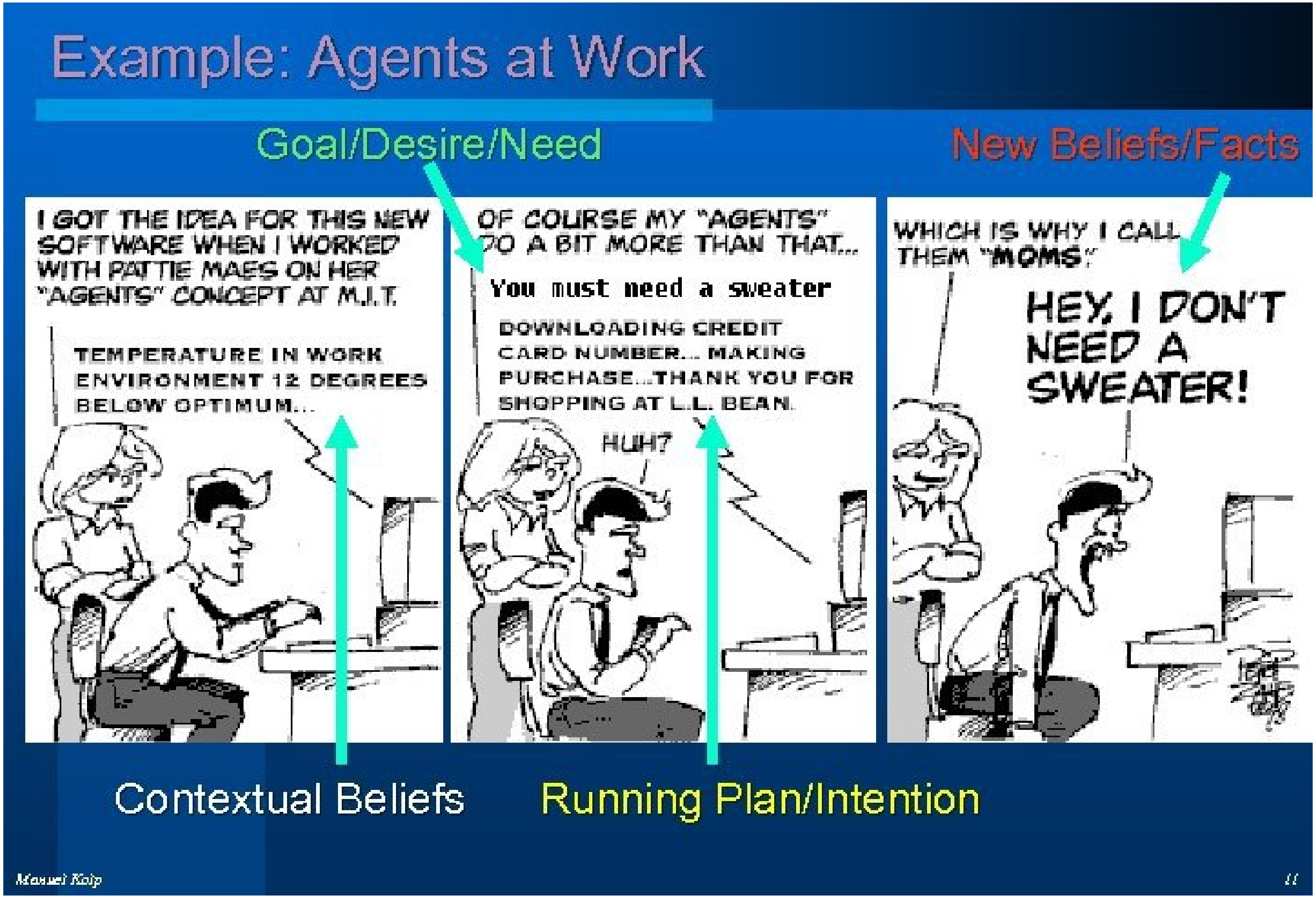
- An agent's plans serve to frame its subsequent reasoning problems so as to constrain the amount of resources needed to solve them
  - Agents *commit* to their plans
  - Their plans tell them *what to* reason about, and *what not* to reason about
- Plans can help reasoning in differents levels of abstraction

Problems in the deliberative approach

More about Planning at PAR course

- Dynamic world
  - Update symbolic world model
  - World changes while planning is being done
- Representation language
  - Expressive enough to be useful in any domain
  - Limited enough to be computationally tractable
- Classical planning → complete, optimal solutions
  - High computational cost
  - Sometimes a sub-optimal low-cost fast reaction can be effective

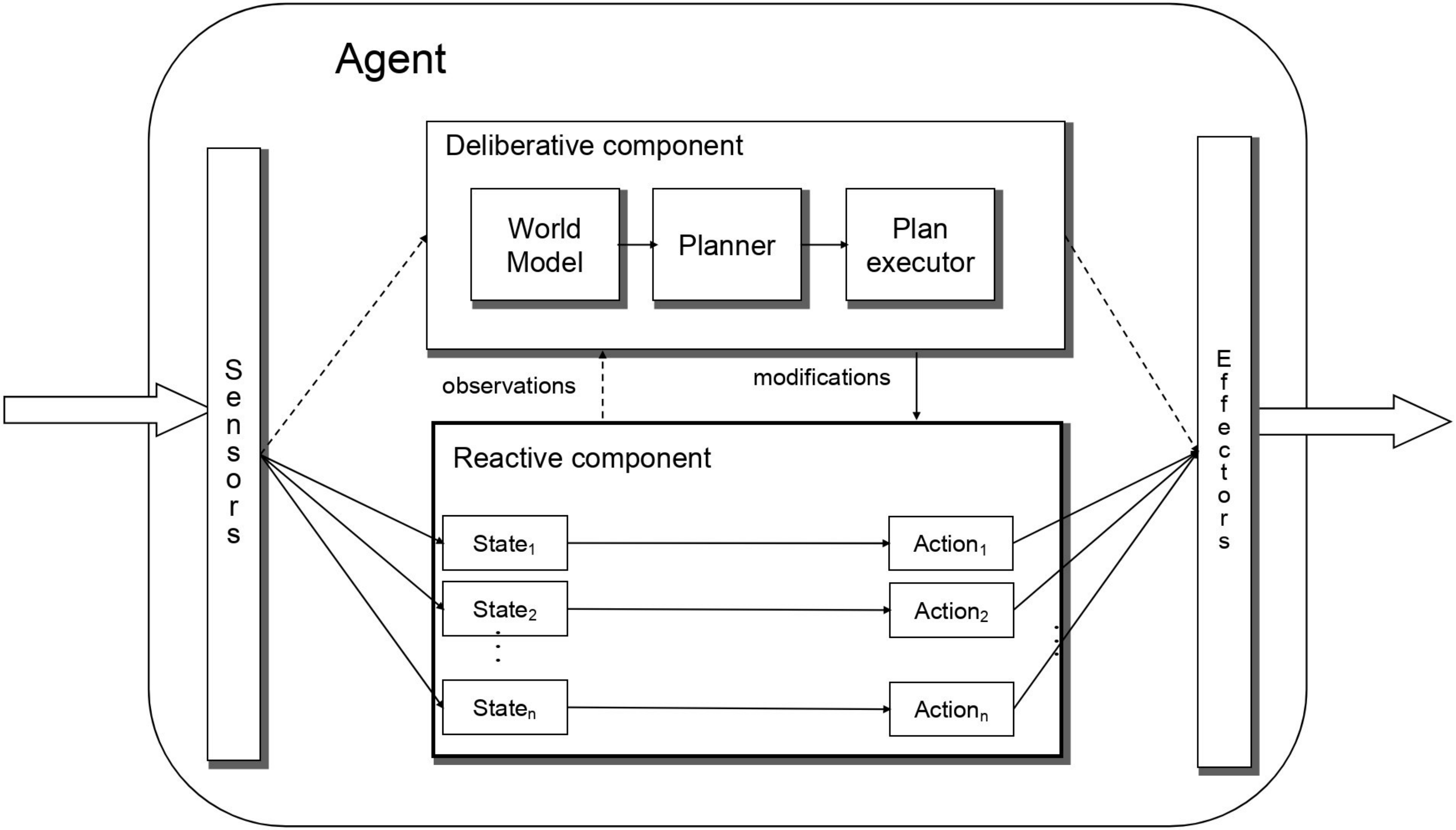
Problems in the deliberative approach



3.3. Agent architectures: Hybrid

- Many researchers have argued that neither a completely deliberative nor a completely reactive approach are suitable for building agents
- They have suggested using *hybrid* systems, which attempt to marry classical and alternative approaches
- An obvious approach is to build an agent out of (at least) two subsystems:
  - A *deliberative* one, containing a symbolic world model, which develops plans and makes decisions in the way proposed by symbolic AI
  - A *reactive* one, which is capable of reacting quickly to events without complex reasoning

Basic schema of hybrid architecture



Layered architectures

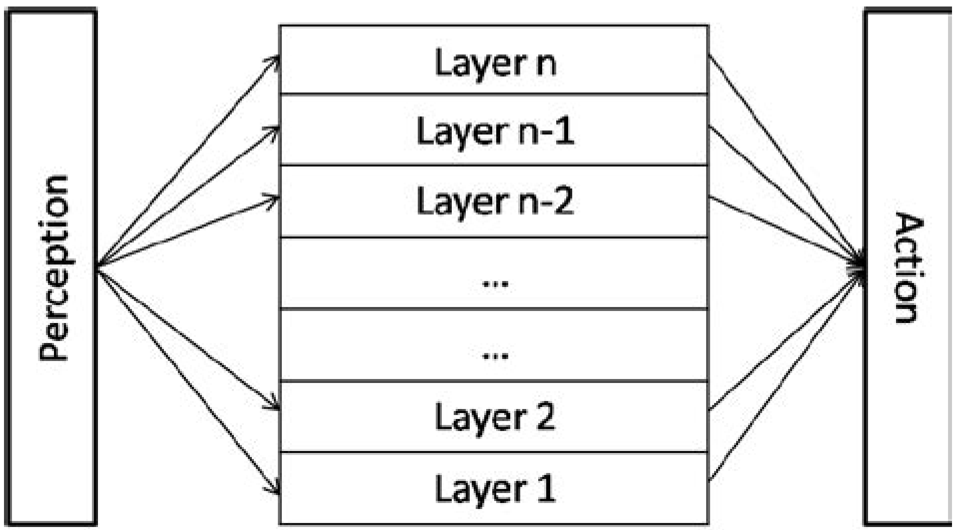
- Often, the reactive component is given some kind of precedence over the deliberative one
- This kind of structuring leads naturally to the idea of a *layered* architecture, of which TOURINGMACHINES and INTERRAP are examples
- In such an architecture, an agent's control subsystems are arranged into a hierarchy, with higher layers dealing with information at increasing levels of abstraction

Layering techniques

- A key problem in such architectures is what kind of control framework to embed the agent's subsystems in, to manage the interactions between the various layers
  - *Horizontal layering*
  - *Vertical layering*

Horizontal layering

m possible actions suggested by each layer, *n* layers

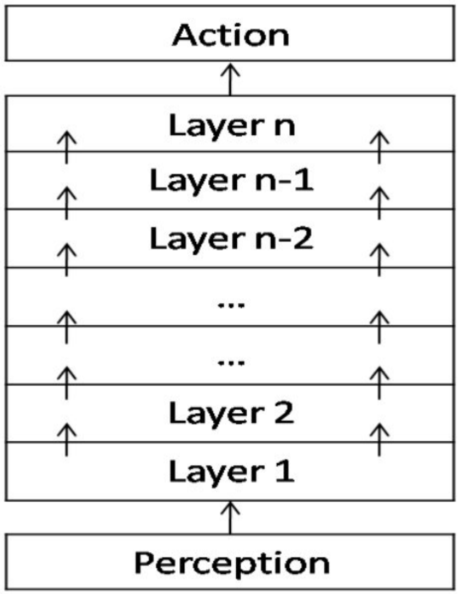


Each layer is directly connected to the sensory input and action output, and suggests actions to perform

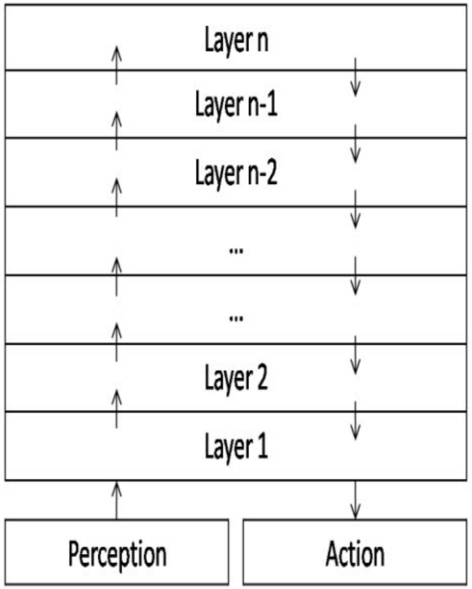
Central control system can be complex, because there are $O(m^n)$ possible options to be considered

Vertical layering

m possible actions suggested by each layer, n layers



One pass



Two pass

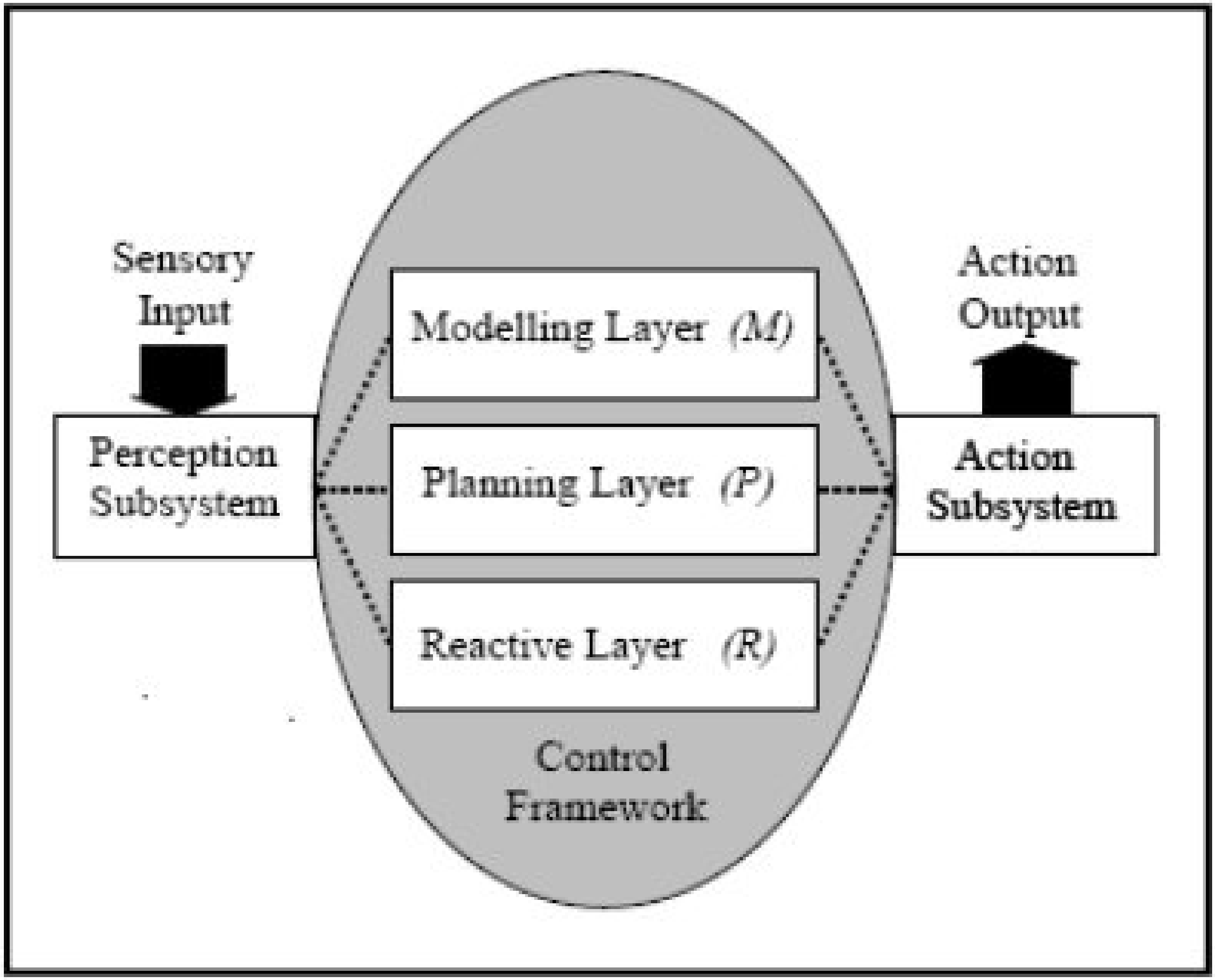
$O(mn)$ interactions between layers

Sensory input and action output dealt with by one layer
Not fault tolerant to layer failure

Ferguson - TOURINGMACHINES

- The TOURINGMACHINES architecture consists of *perception* and *action* subsystems, which interface directly with the agent's environment, and three *control layers*, embedded in a *control framework*, which mediates between the layers

Ferguson - TOURINGMACHINES



Ferguson - TOURINGMACHINES

- The *reactive layer* is implemented as a set of situation-action rules (subsumption architecture)

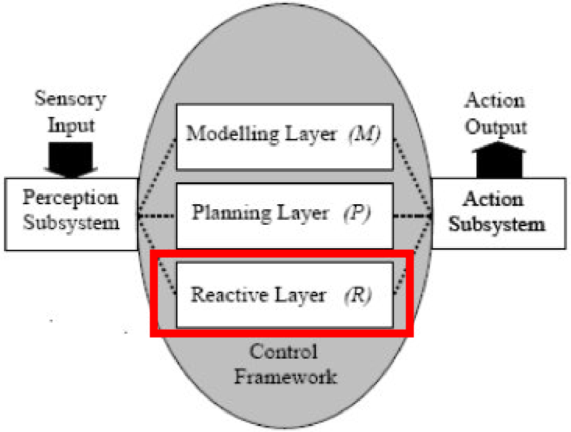
  rule-1: obstacle-avoidance

  if

  is-in-front(Obstacle, Observer) and
  speed(Observer) > 0 and
  separation(Obstacle, Observer) < ThreshHold

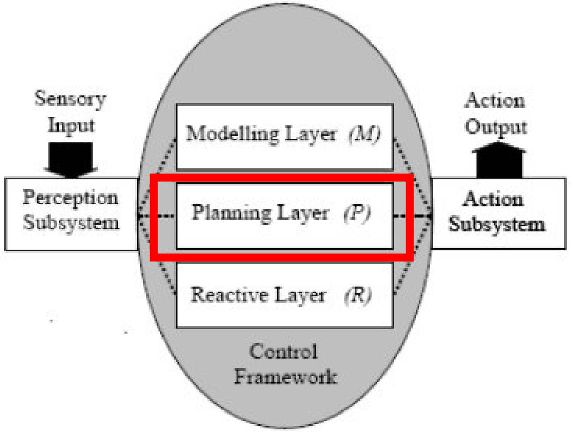
  then

  change-orientation(ObstacleAvoidanceAngle)



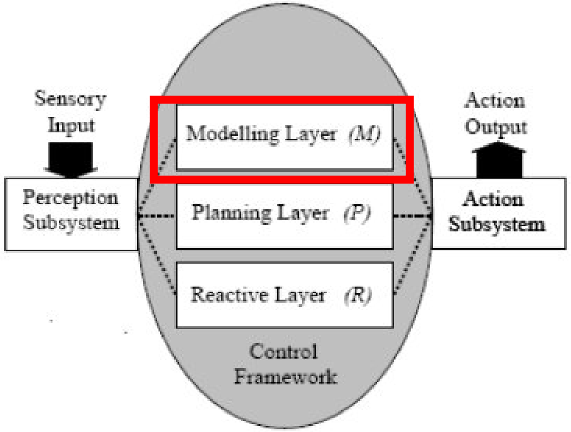
Ferguson - TOURINGMACHINES

- The *planning layer* constructs plans and selects actions to execute in order to achieve the agent's goals



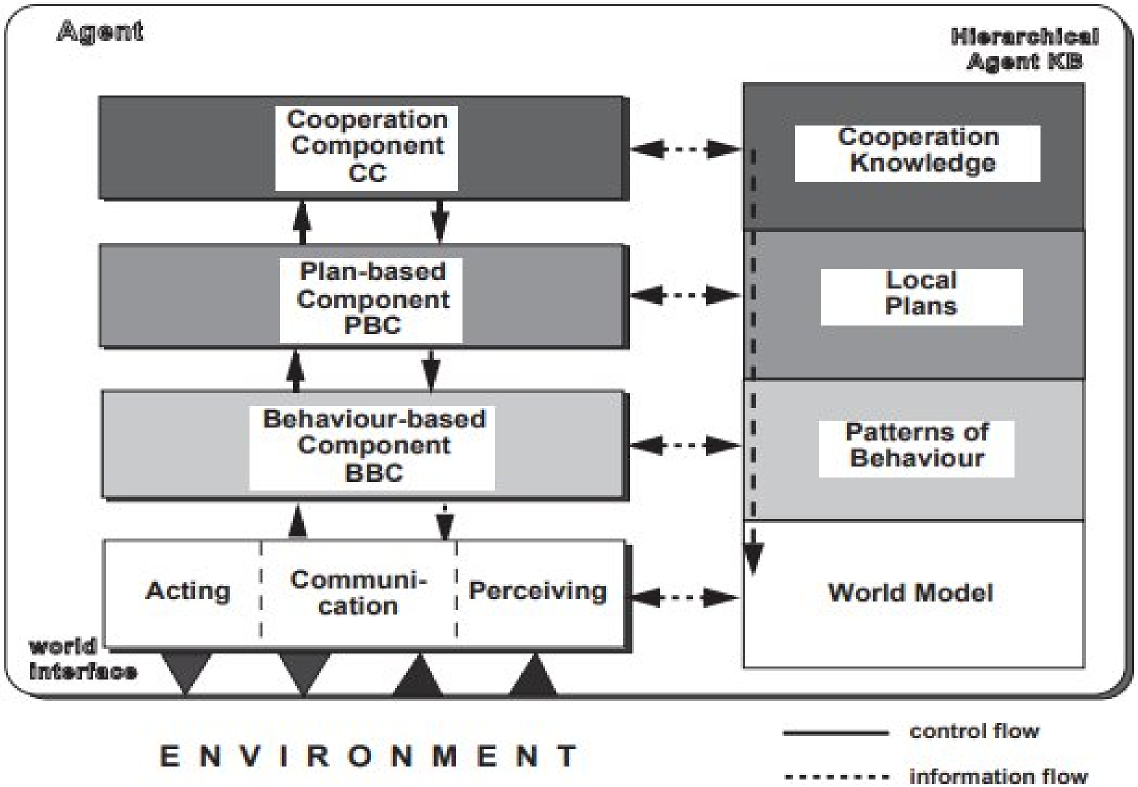
Ferguson - TOURINGMACHINES

- The *modeling layer* contains symbolic representations of the 'cognitive state' of other entities in the agent's environment
  - E.g., identifying entity behaviours or any other world events which had not been expected, detecting and resolving goal conflicts, making short- or long-term spatio-temporal predictions about entities



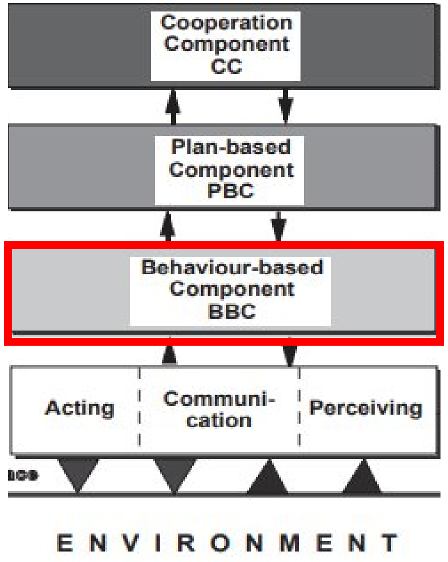
Müller - InteRRaP

- Integration of Reactive Behaviour and Rational Planning
- Vertically layered



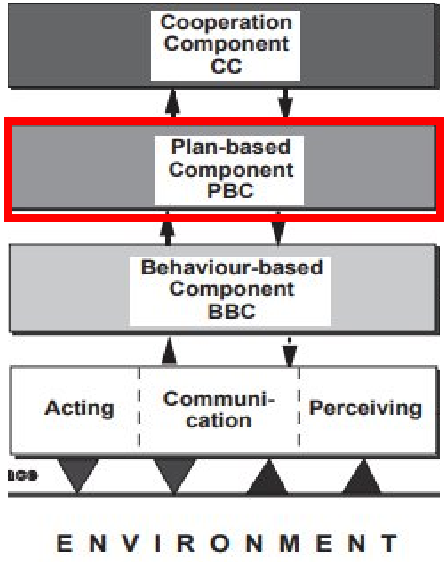
Müller - InteRRaP: Behaviour layer (BBC)

- Reactive part of the architecture
- Works with the world model (beliefs on the world state)
- Only one level interacts with the real world
- Has a set of "situation action" rules
  - Fast recognition of situations that deserve a quick reaction
- Makes routine tasks efficiently, without complex symbolic planning



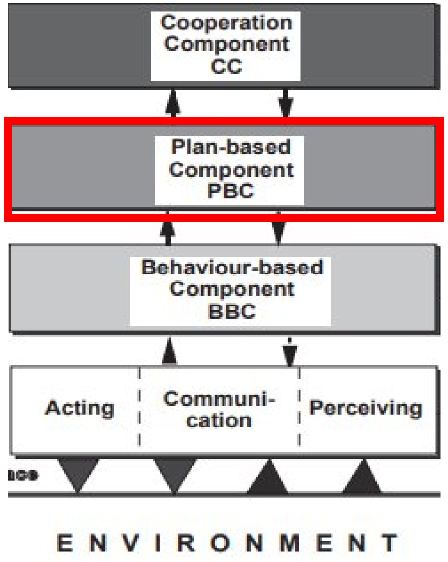
Müller - InteRRaP: Planning layer (PBC)

- Works with the mental model (beliefs on the own agent)
- Standard deliberative level
- Implements local behaviour guided towards certain goals



Müller - InteRRaP: Cooperative planning layer (CC)

- Works with the social model (beliefs on other agents of the system)
- Allows planning and cooperation with other agents
  - Global plans of action
  - Conflict resolution



Critiques to hybrid architectures

- Lack of general methodologies to guide the design process
- Very specific, application dependent
- Unsupported by formal theories

Proposed readings

- M. Wooldridge: *An introduction to Multi-Agent Systems* – *chapter 5 (reactive, hybrid)*
- A.Mas: *Agentes software y sistemas multi-agente: conceptos, arquitecturas y aplicaciones* – *chapter 2*