Agent Architectures

Hybrid Agents

Sources

- www.wirtschaft.tu-ilmenau.de/wi/wi2/SPP-Agenten/
- http://www.csc.liv.ac.uk/~mjw/pubs/imas

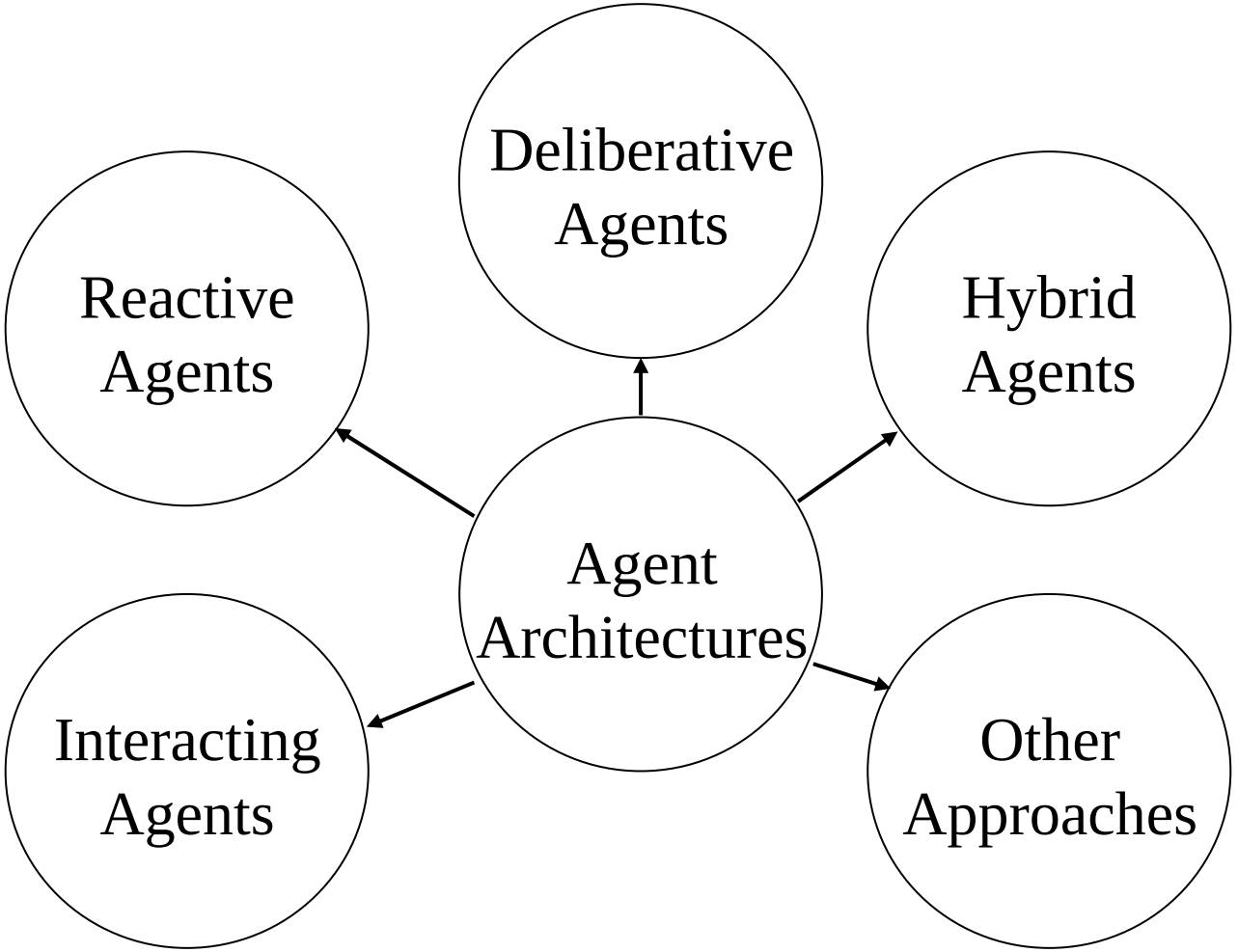
ASSIGNMENT

1. After reading these notes answer the following questions regarding ***hybrid architectures*** (one page max for all!)

- How does the architecture distinguish between reaction and deliberation?
- How does it organize responsibilities in the deliberative portion?
- How does overall behavior emerge?

- 2. Draft an agent architecture for your project and bring it to class on a .ppt so you can describe it in 5 mins – making the point of what AI technologies you can use

Agent architectures



Hybrid architectures

- Combine **reactive** and **deliberative** components and form a hierarchy of interacting layers
- Each layer reasons at a different level of abstraction
- Two types of layering:
  - Horizontal layering
  - Vertical layering

Agent Architectures

Reactive Agent

- Each behaviour continually maps perceptual input to action output
- Reactive behaviour:

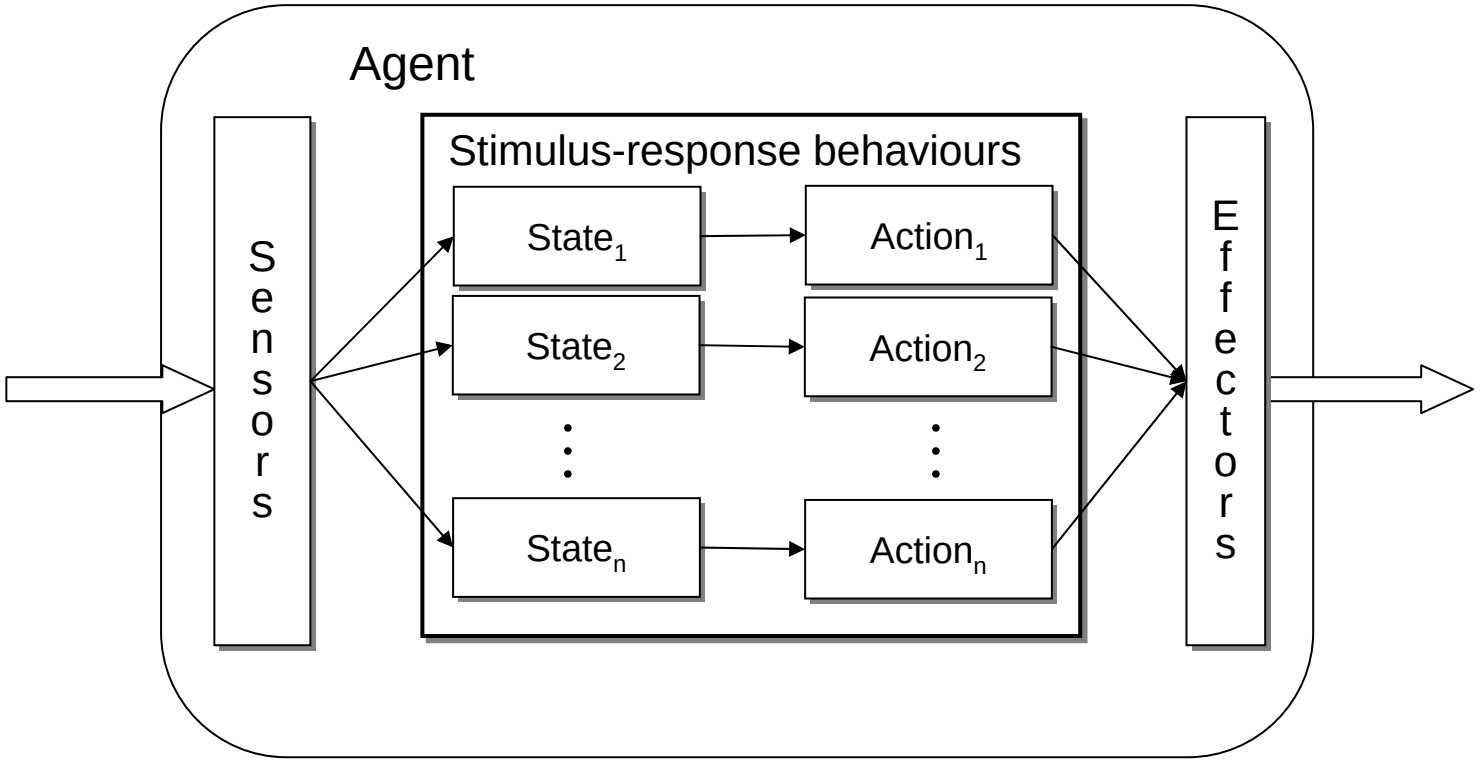
  action: S -> A

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

$$\text{action}(s) = \begin{cases} \textit{Heater on}, \text{ if temperature too low} \\ \textit{Heater off}, \text{ otherwise} \end{cases}$$

Agent Architectures

Reactive Agent



Agent Architectures

Reactive Agent

- Problems
  - a great deal of local information needed
  - learning?
  - Typically “handcrafted”
    - Development takes a lot of time
    - Impossible to build large systems?
    - Can be used only for its original purpose

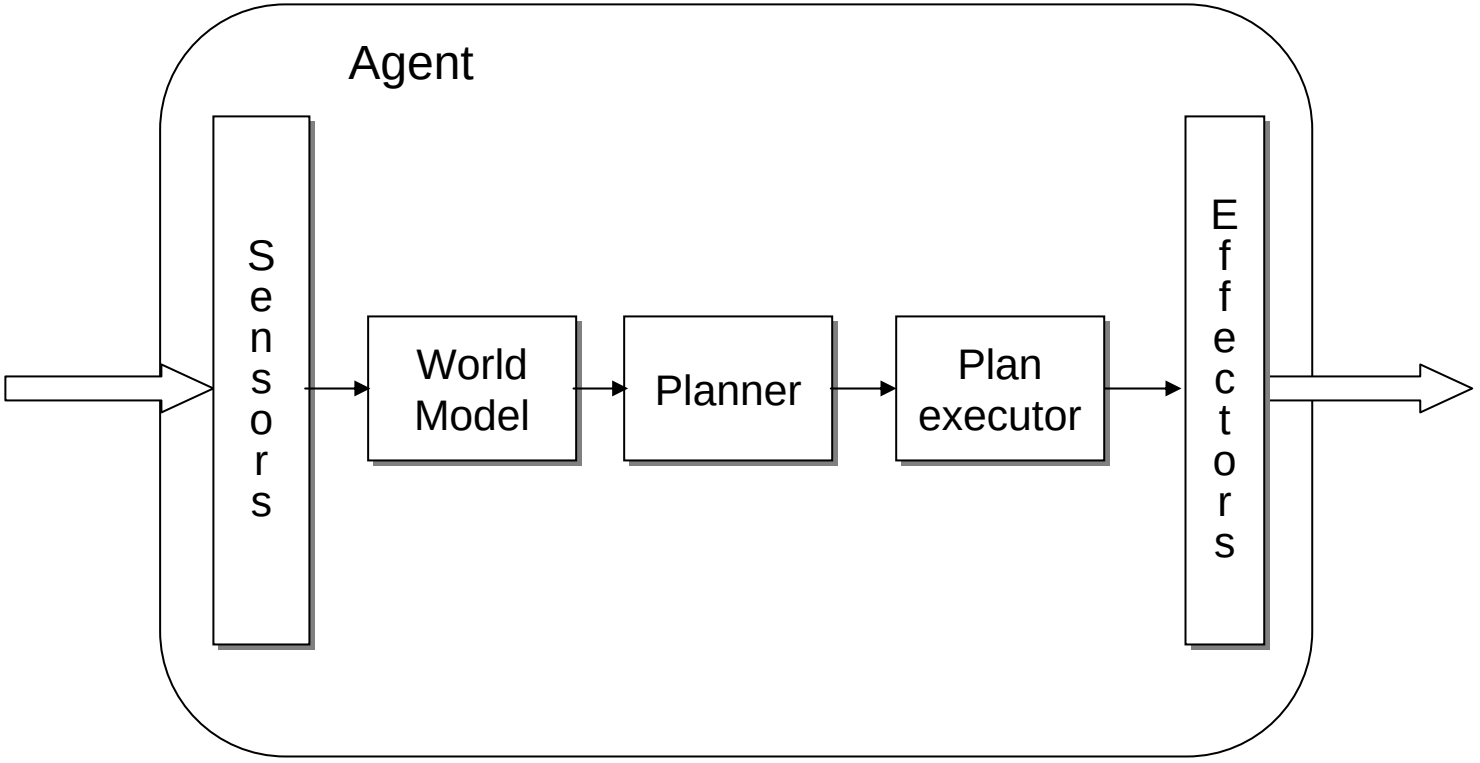
- Examples
  - Brooks: subsumption architecture
    - ref: Http://ai.eecs.umich.edu/cogarch3/Brooks/Brooks.html

Agent Architectures

- Deliberative Agent
  - Explicit symbolic model of the world in which decisions are made via logical reasoning, based on pattern matching and symbolic manipulation
  - sense-plan-act problem-solving paradigm of classical AI planning systems

Agent Architectures

Deliberative Agent



Agent Architectures

Deliberative Agent

- Examples of deliberative architectures
  - BDI
  - Shoham: Agent-Oriented Programming

Agent Architectures

Deliberative Agent

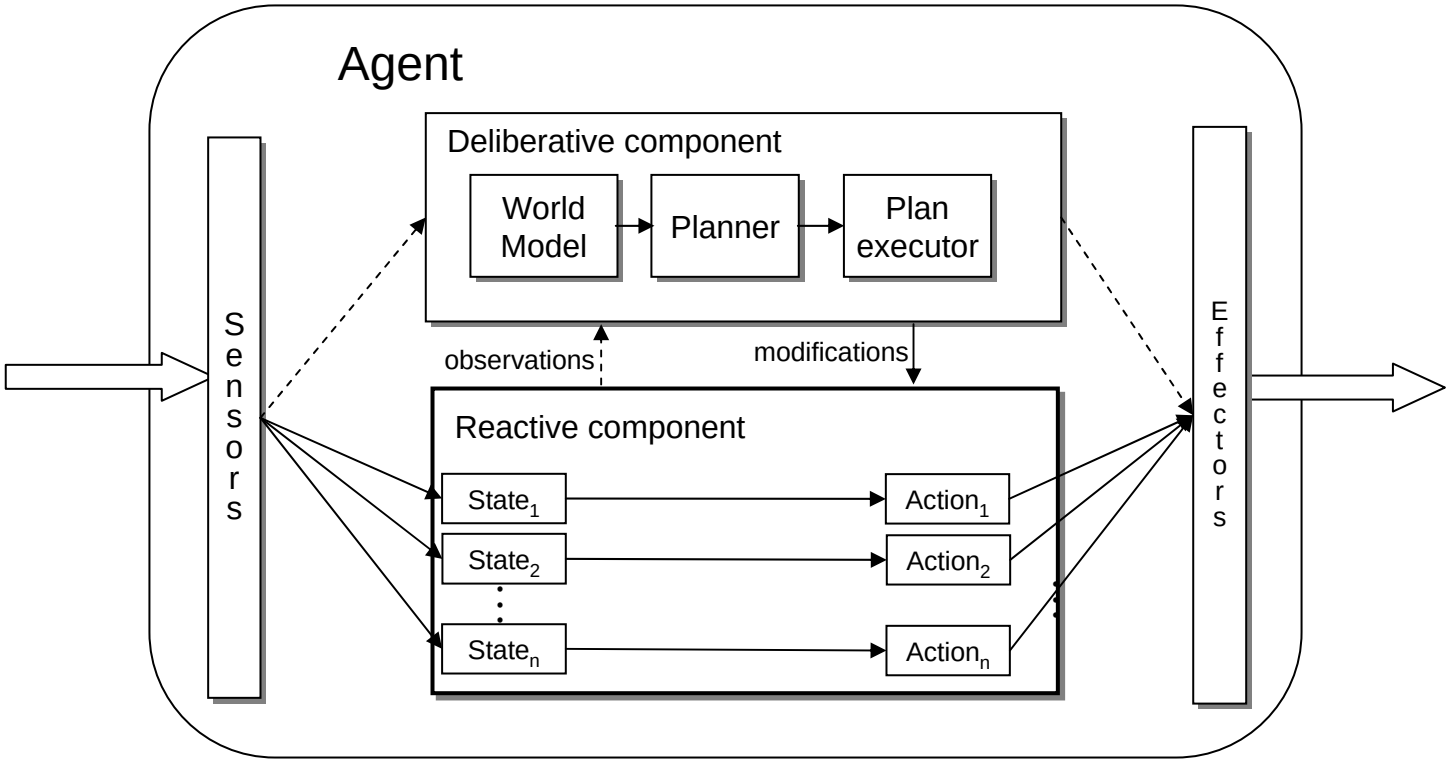
- Performance problems
  - *transduction* problem
    - time consuming to translate all of the needed information into the symbolic representation, especially if the environment is changing rapidly.
  - *representation* problem
    - how the world-model is represented in symbolically and how to get agents to reason with the information in time for the results to be useful.
- Late results may be useless
- Does not scale to real-world scenarios

Agent Architectures

- Reactive agents have
  - at most a very simple internal representation of the world,
  - but provide tight coupling of perception and action
- Behaviour-based paradigm
- Intelligence is a product of interaction between an agent and its environment
- Do we really need abstract reasoning?

Agent Architectures

Hybrid Agent



Deliberation v. Reaction as a function of TIME

- Past, Present, Future
- Reactive
  - exists in the PRESENT (will a bit of duration)
- Deliberative
  - can reason about the PAST
  - can project into the FUTURE

Agent Architectures

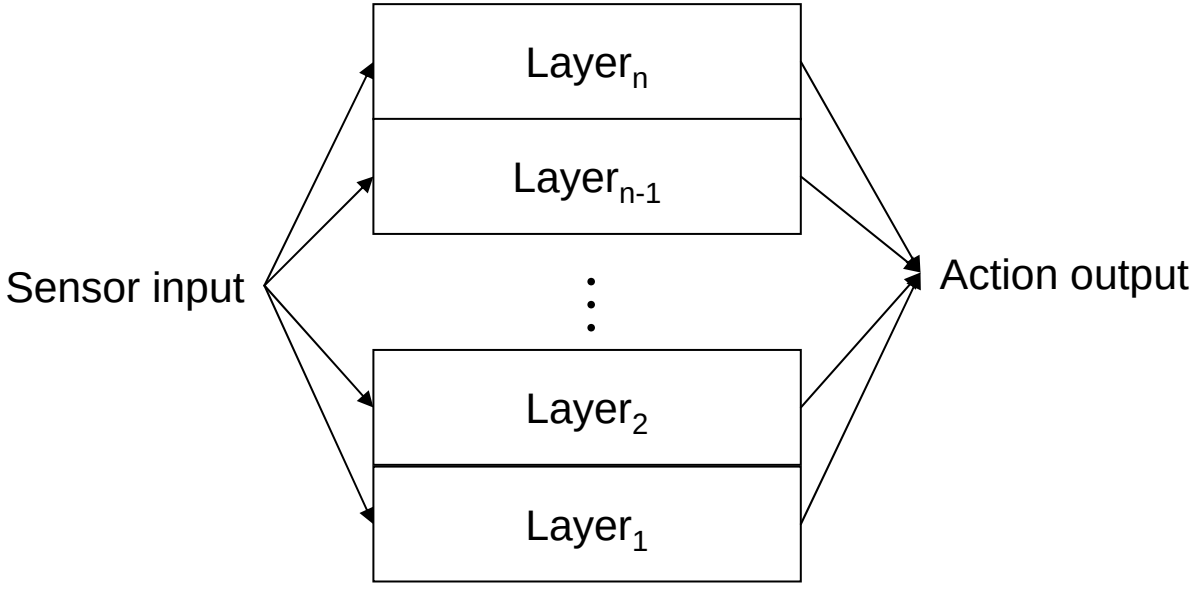
Hybrid Agent

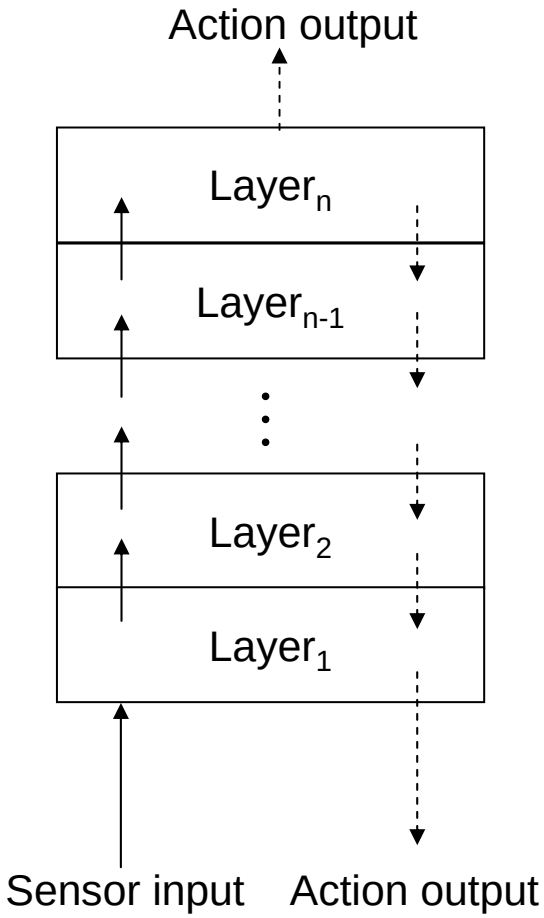
- Combination of deliberative and reactive behaviour
  - An agent consists of several subsystems
    - Subsystems that develop plans and make decisions using symbolic reasoning (deliberative component)
    - Reactive subsystems that are able to react quickly to events without complex reasoning (reactive component)

- Layered architectures

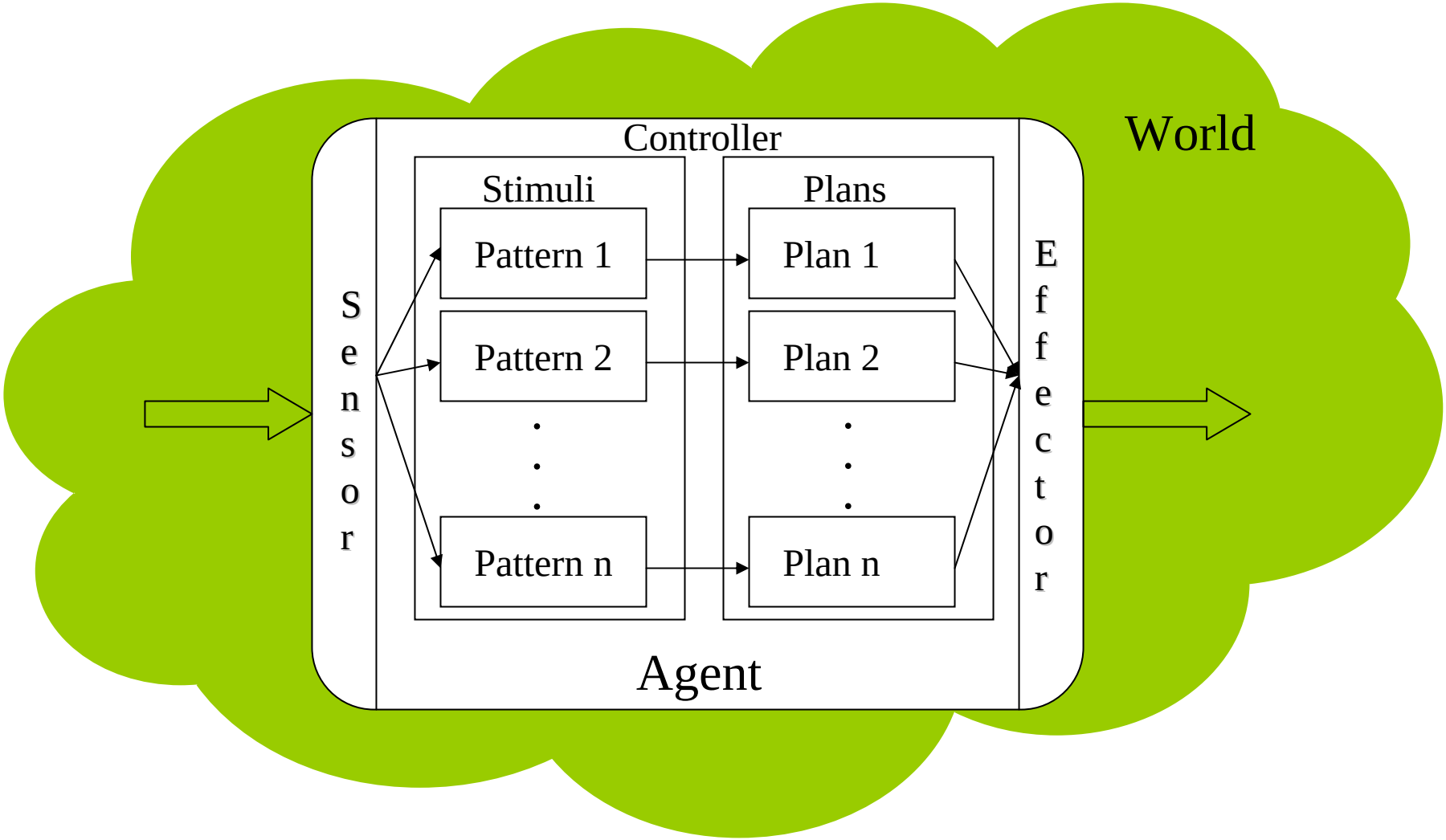
Agent Architectures

Hybrid Agent

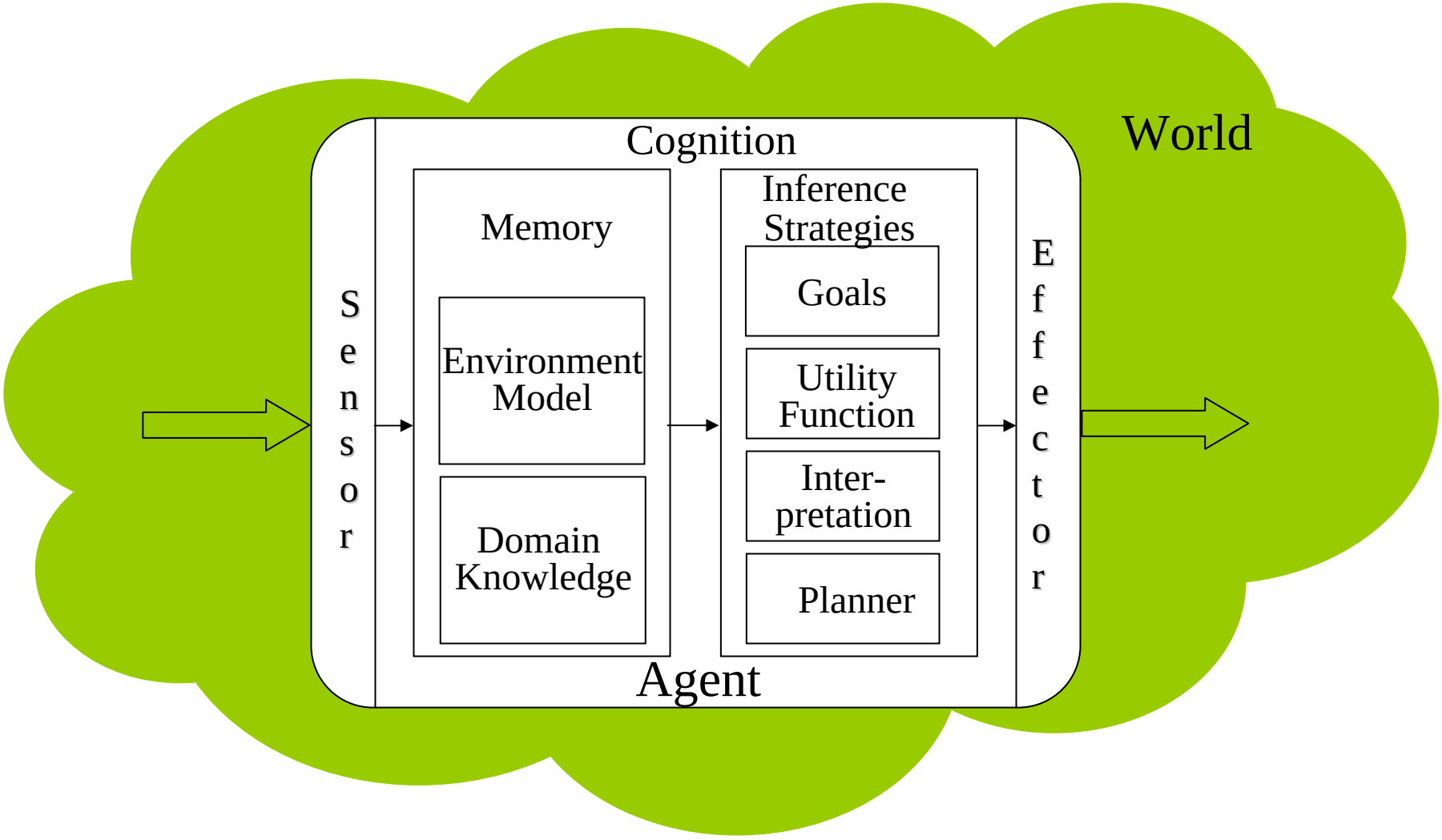




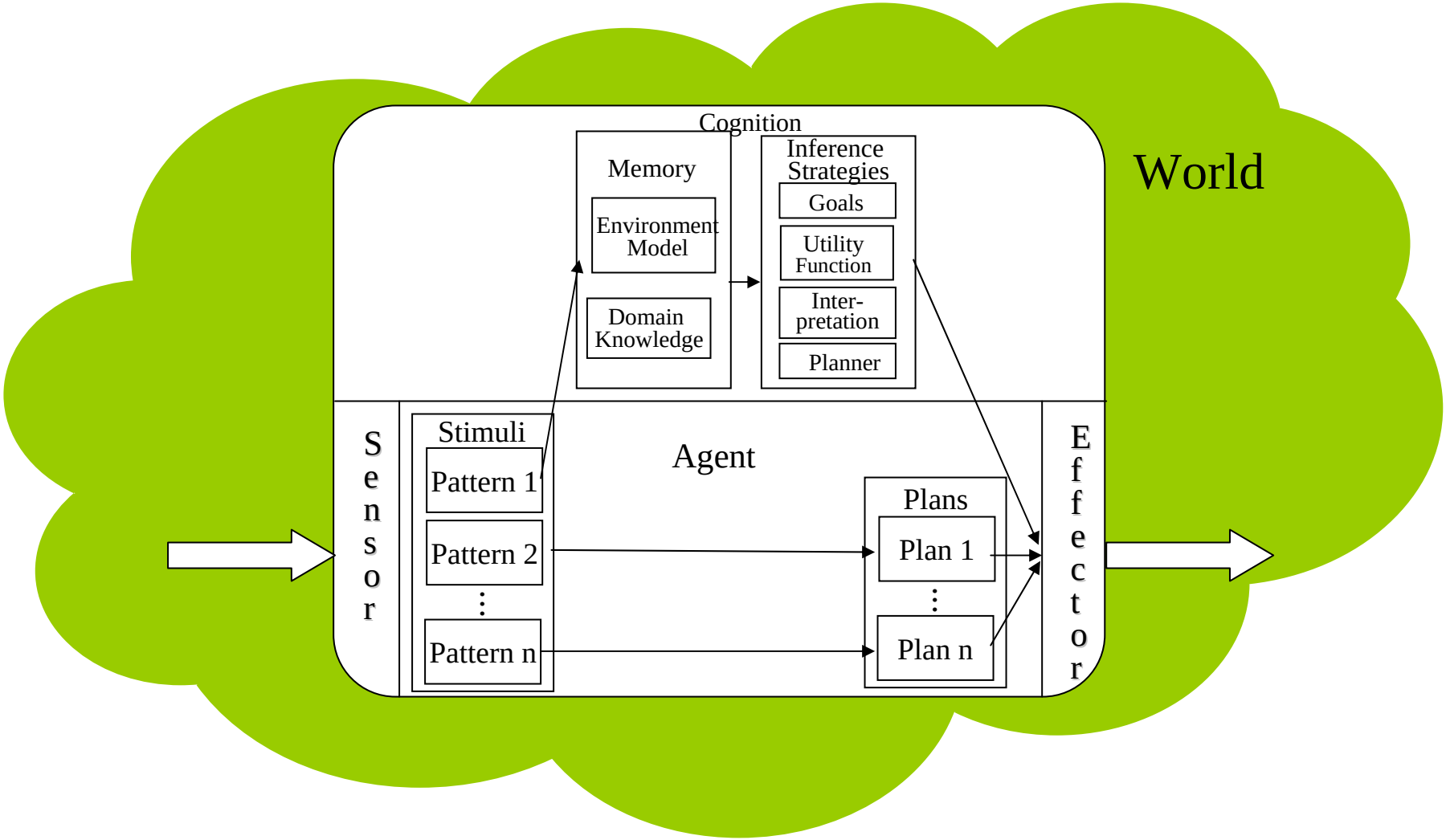
Reactive Agents



Deliberative Agents

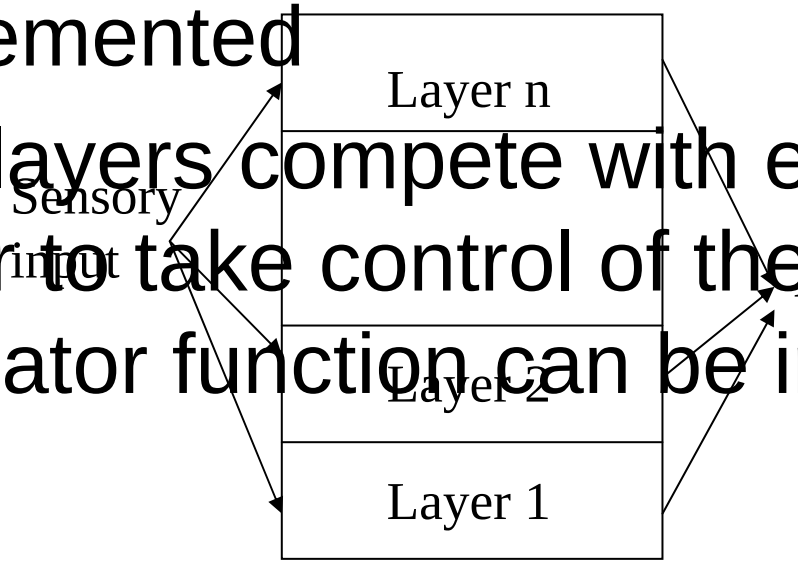


Hybrid Agents



Horizontal layering

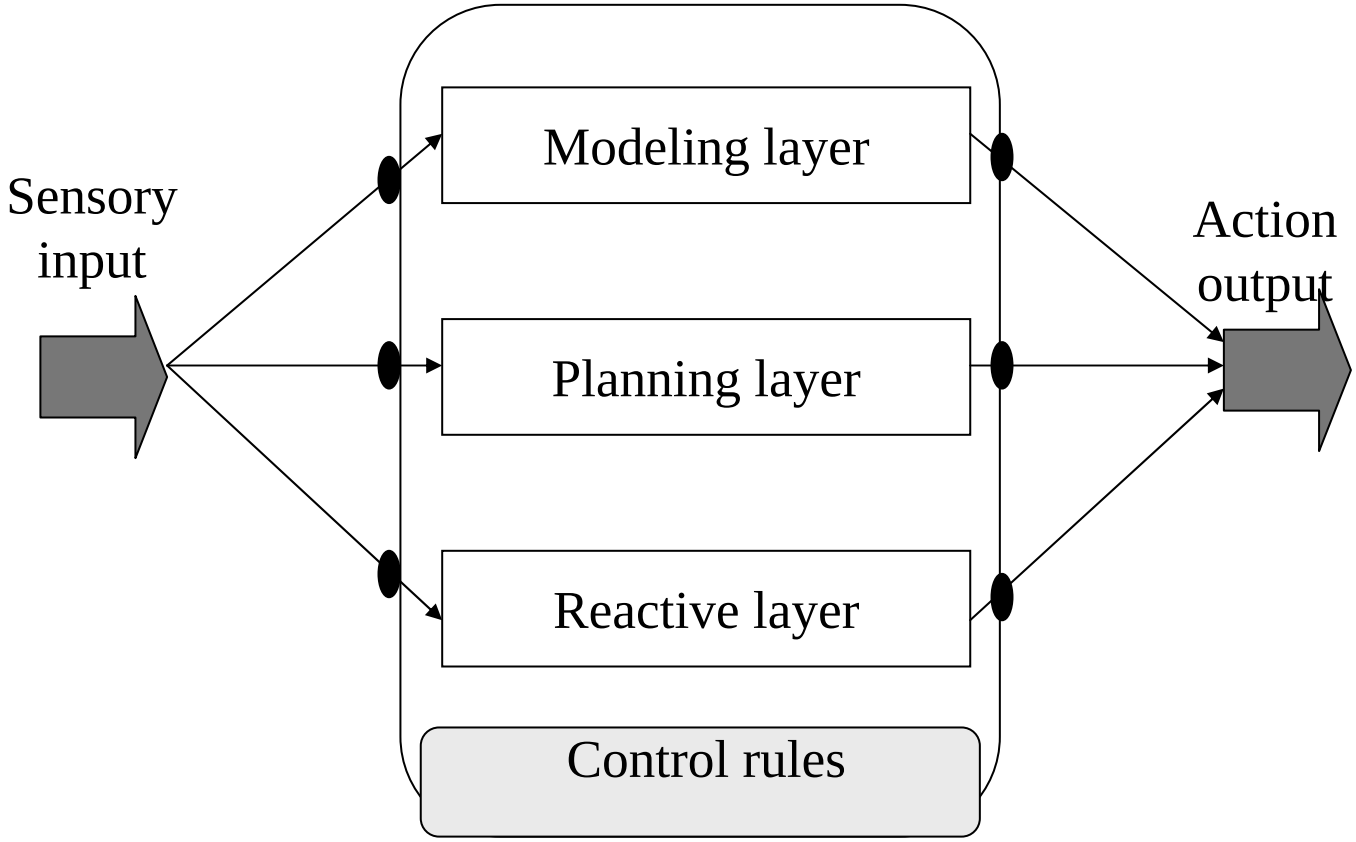
- Each layer can act as an independent agent
- For *n* different behaviours *n* layers are implemented
- The layers compete with each other in order to take control of the agent; a mediator function can be introduced



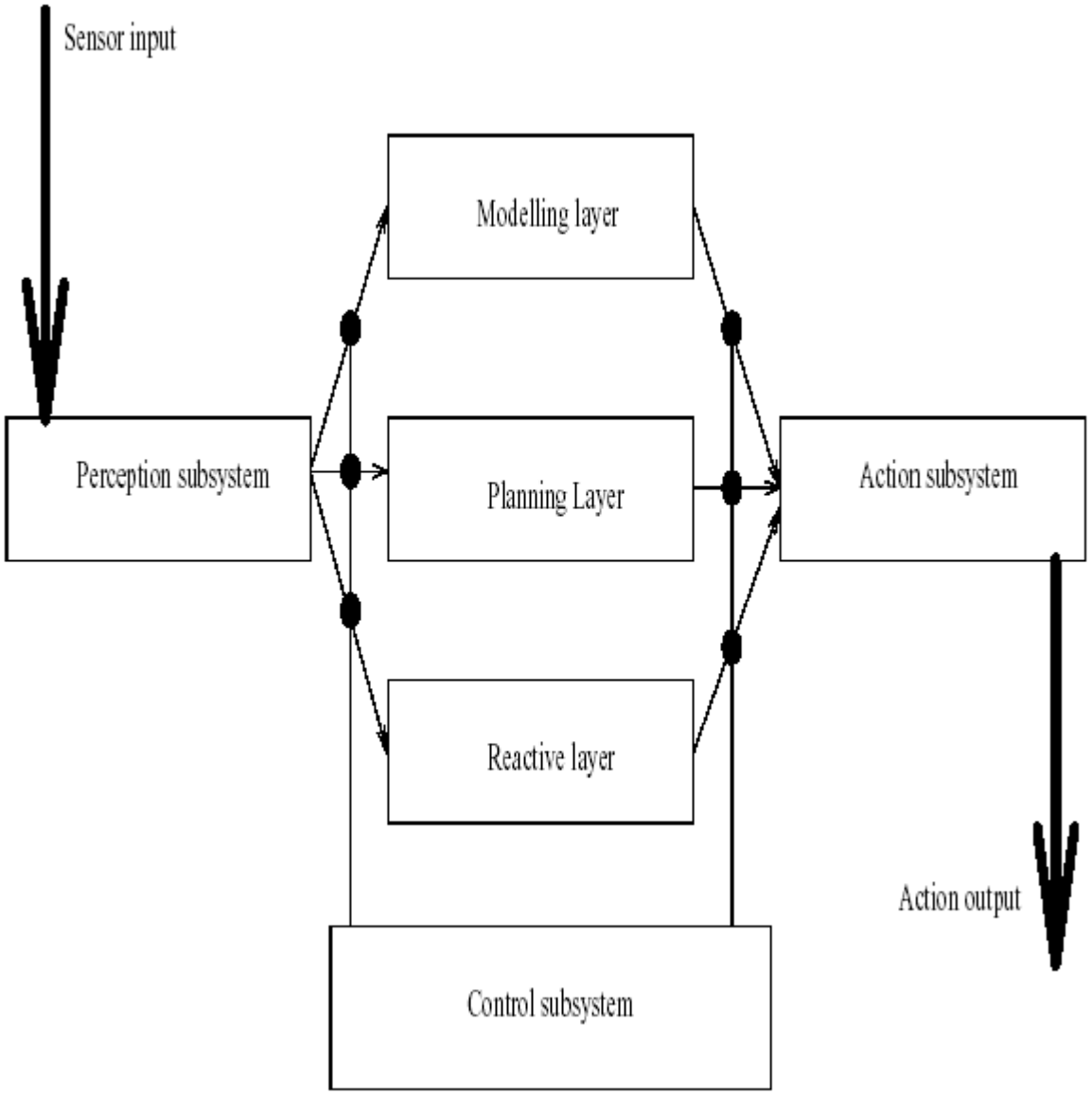
Problems

- The layers' competition for the agent's control can cause incoherence
- Consistency can be achieved by introducing a function which achieves mediation between the layers
- Mediator function is exponentially complete: if there are *n* layers capable of suggesting *m* possible actions there are m^n interactions
- The mediator function or a central control system can introduce a bottleneck into the agent's decision making

Example: TouringMachines



Ferguson –


Sensor input
Modelling layer
Perception subsystem
Planning Layer
Action subsystem
Reactive layer
Action output
Control subsystem

Ferguson – TOURINGMACHINES

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

  Example:

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

- 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
- The three layers communicate with each other and are embedded in a control framework, which use *control rules*

Example:

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

Reactive layer

- Acts as a reactive agent and responds to changes as they occur
- Implemented through situation-action rules
- There is no model of the environment in this layer

Planning layer

- Achieves the agent’s pro-active behaviour via plans based on a library of plan skeletons or schemas

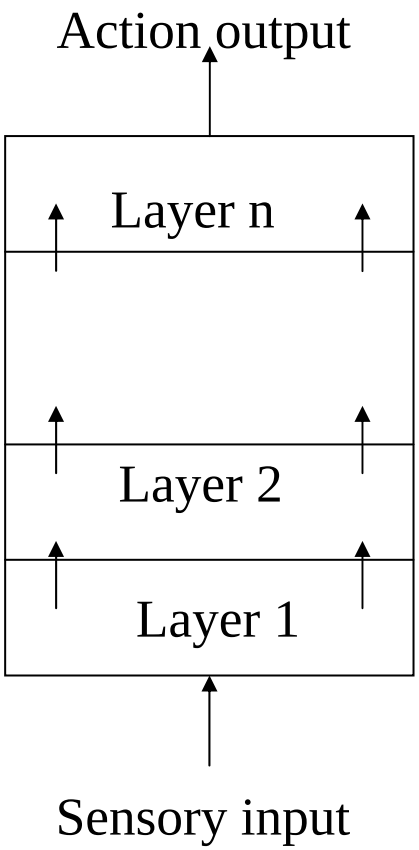
Modelling layer

- Endows the agent with reflective and predictive capabilities
- Entities are modelled as having a configuration, beliefs, desires and intentions
- Generates goals to resolve conflicts which are then propagated to the planning layer

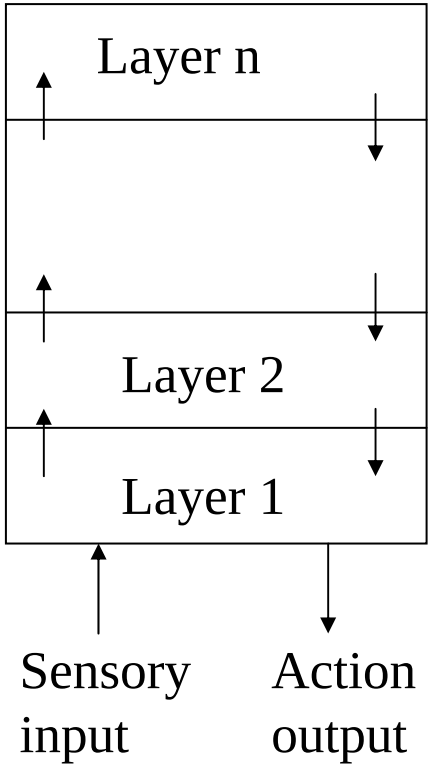
Control subsystem

- Decides which of the layers has control over the agent
- It is implemented via control rules which can either suppress sensor information between the control rules and the control layers or else censor action outputs from the control layers

Vertical layering



One-pass



Two-pass

Advantages

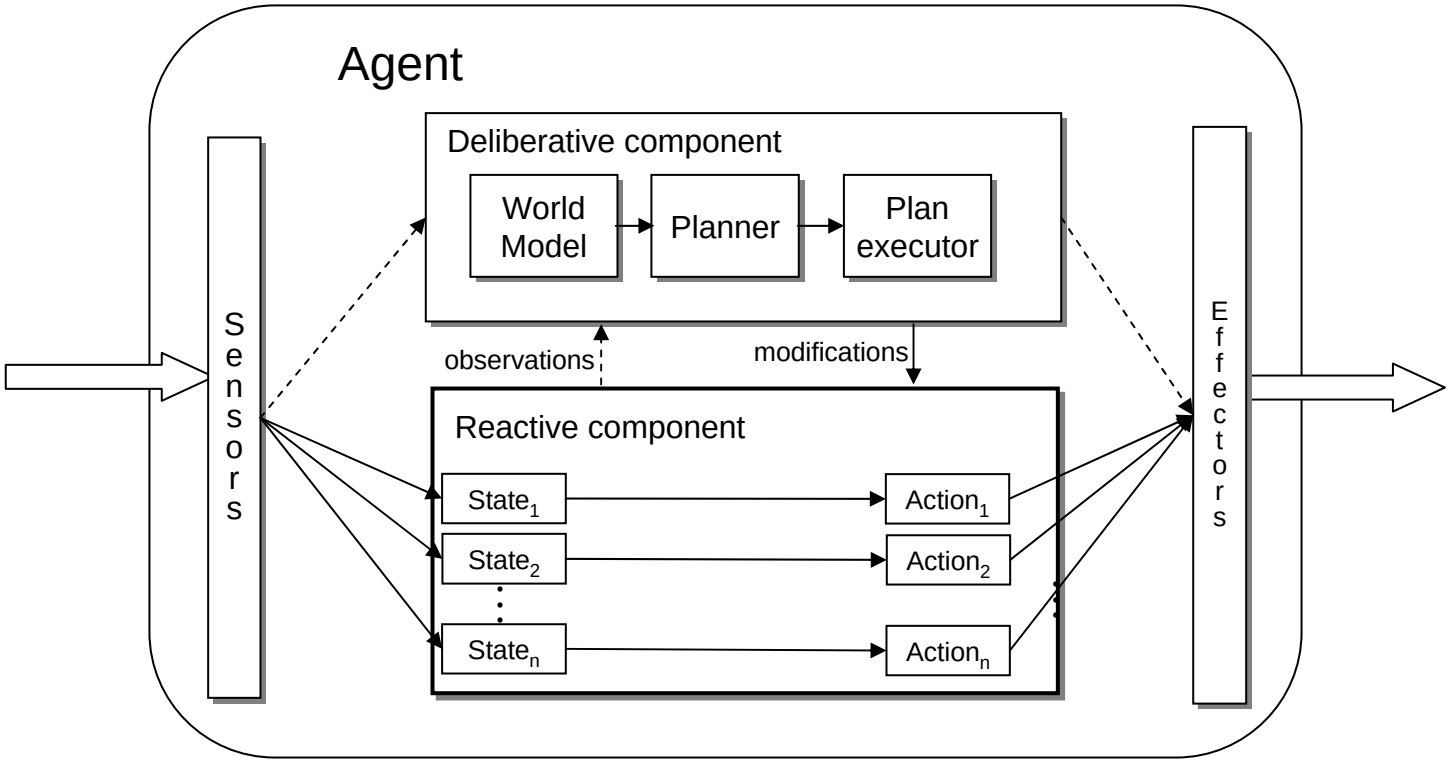
- Low complexity. If there are *n* layers there are *n*-1 interfaces between them. If each layer is capable of suggesting *m* possible actions then there are at most $m^2(n-1)$ interactions
- No central control, no bottleneck in the agent's decision making

Problems

- Less flexible
- Not fault tolerant

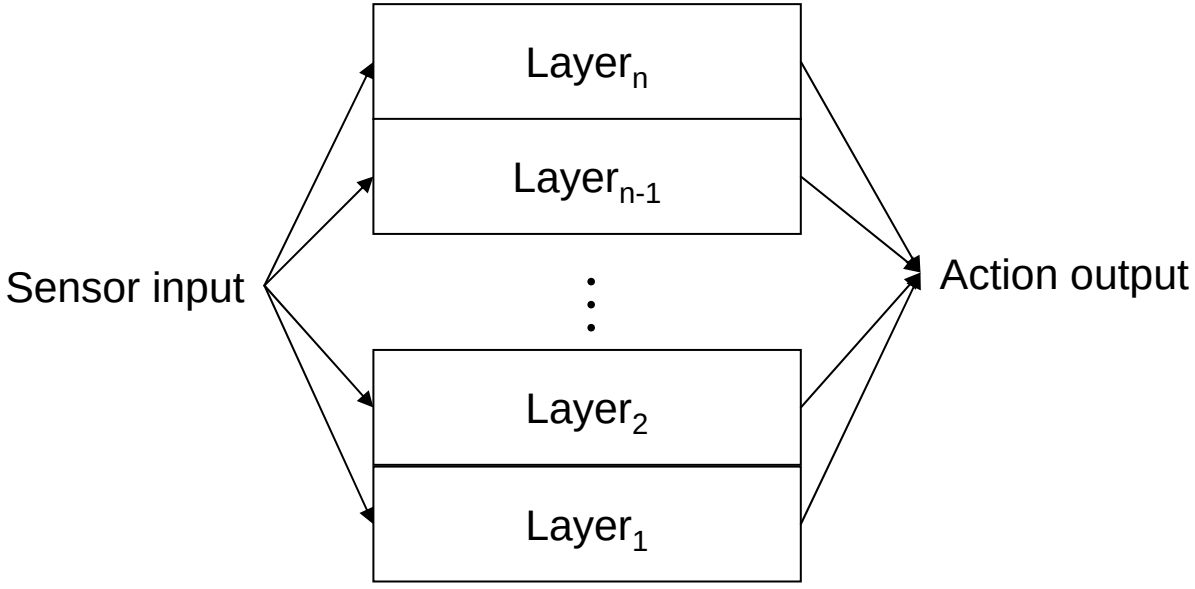
Agent Architectures

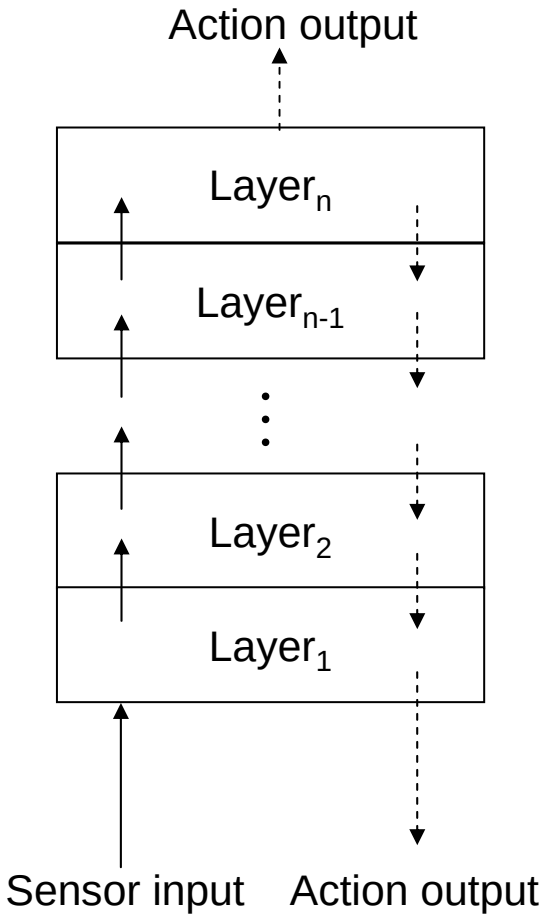
Hybrid Agent



Agent Architectures

Hybrid Agent

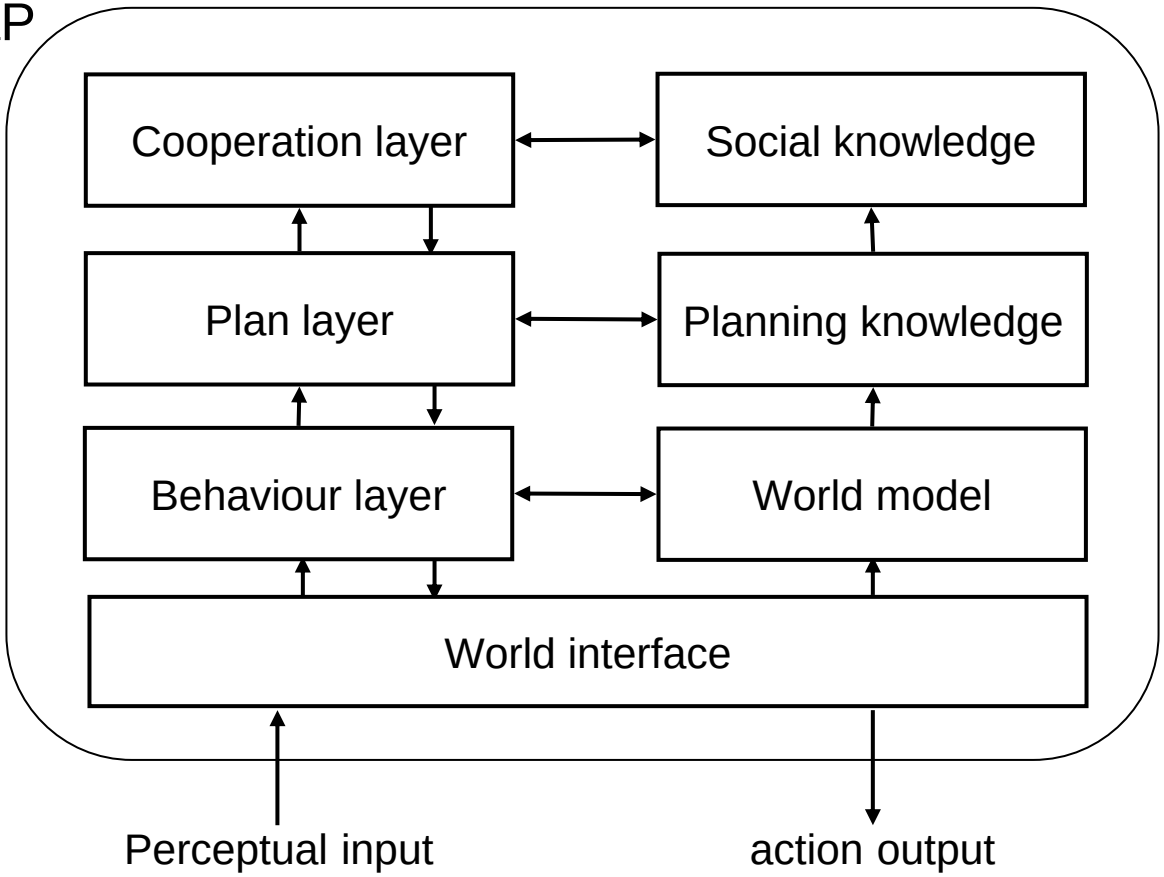




Agent Architectures

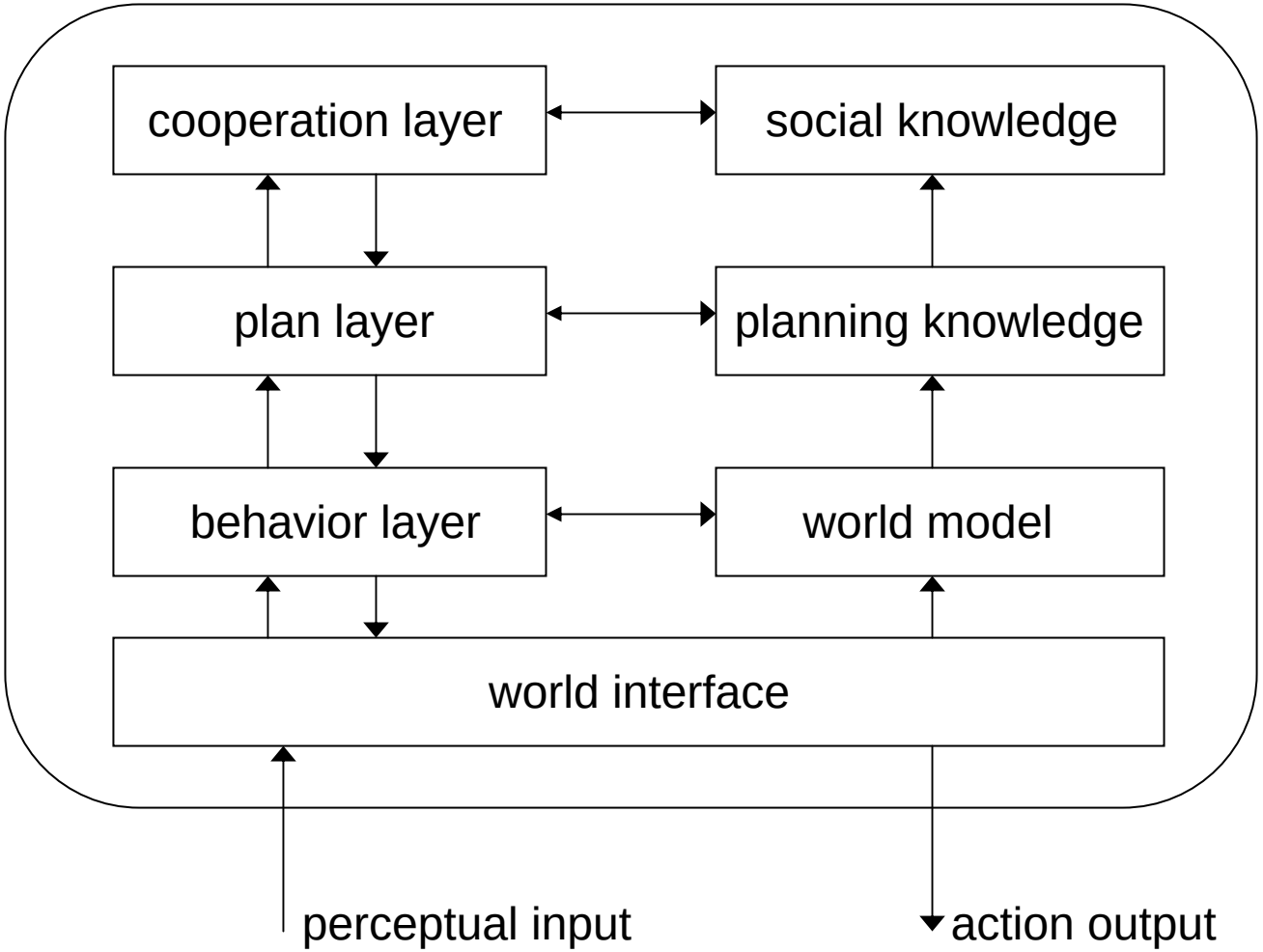
Hybrid Agent

example - InteRRaP

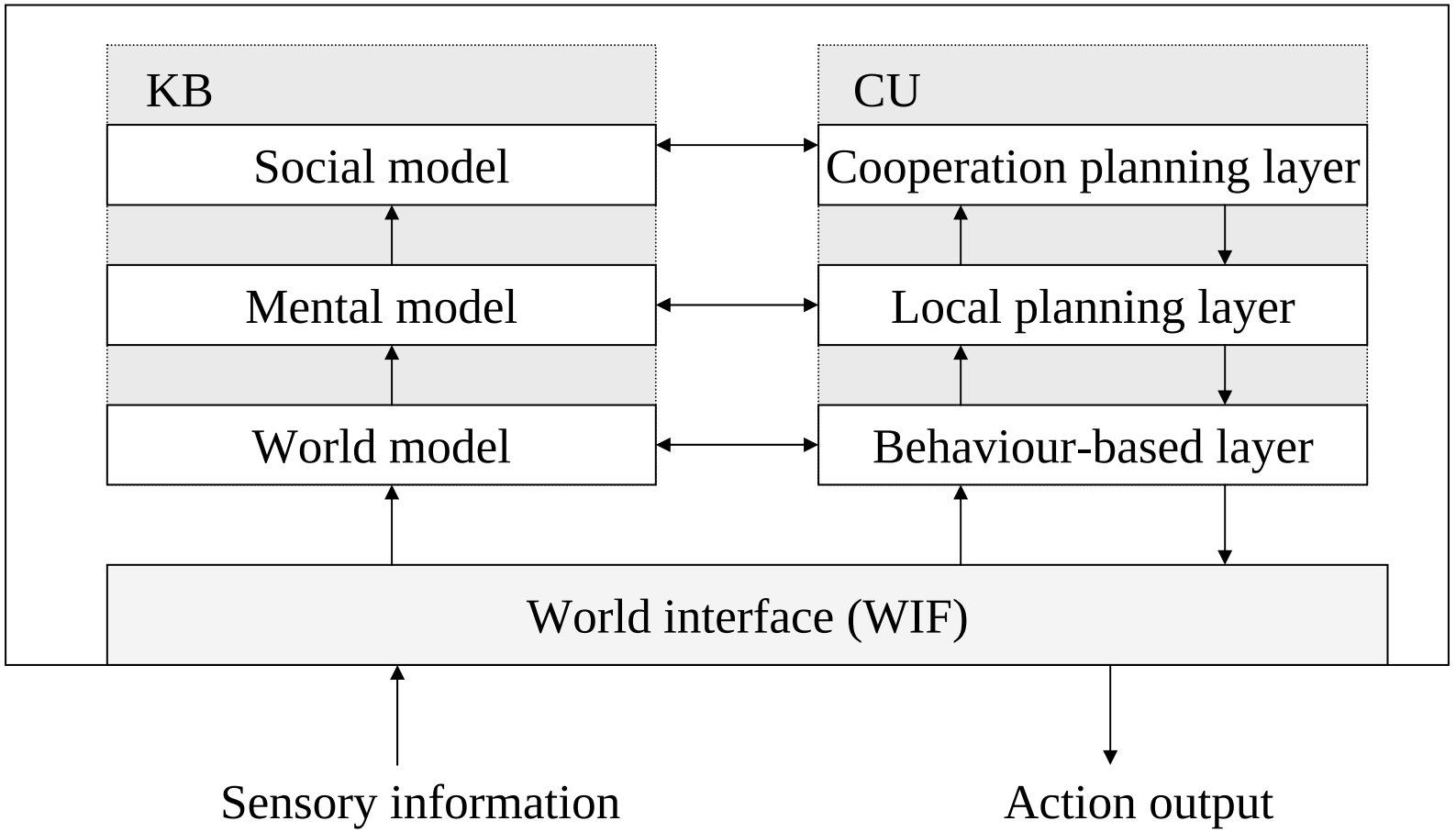


Müller –InteRRaP

- Vertically layered, two-pass architecture



InteRRaP


KB
CU
Social model
Cooperation planning layer
Mental model
Local planning layer
World model
Behaviour-based layer
World interface (WIF)
Sensory information
Action output

Each layer consists of two subprocesses

- Situation recognition and goal activation process (SG)
- Planning, scheduling and execution process (PS)

Two main types of interactions take place between the layers:

- Activation requests (bottom up) which are issued when a lower layer passes control to a higher layer. The request is issued by the PS of layer *i* to the SG of layer *i+1*
- Commitment postings (top down) are sent from layer *i* to *i-1* in order to achieve its goals. These are communicated between the PSs of the two layers

Status

- "toolbox" of agent architecture types available
- benchmarking of agent architectures?
- agent architecture design as an engineering discipline?
- (proven) standards for agent architectures?
- which architecture for which problem?
- agent architectures vs. related "non-agent" architectures (client/server, CORBA, etc.)?
- agent architectures vs. MA systems architectures?

Interacting Agents