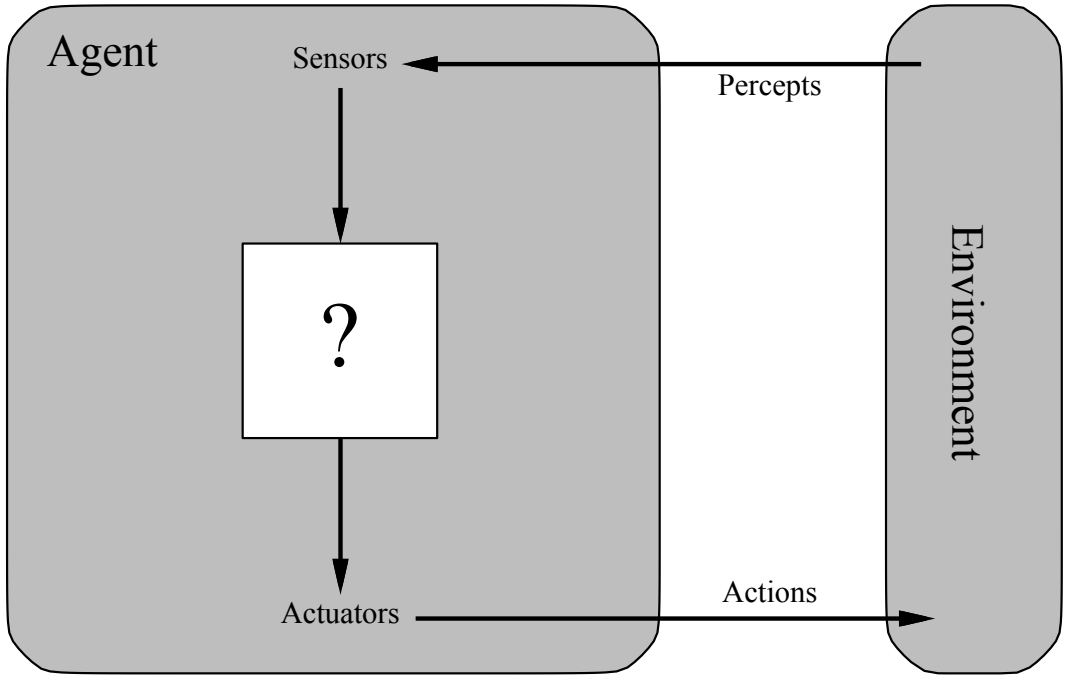
Artificial Intelligence / Applied Artificial Intelligence

Intelligent Agents

Outline

- Agents and environments
- PEAS (Performance measure, Environment, Actuators, Sensors)
- Environment types
- Agent types

Agents and environments



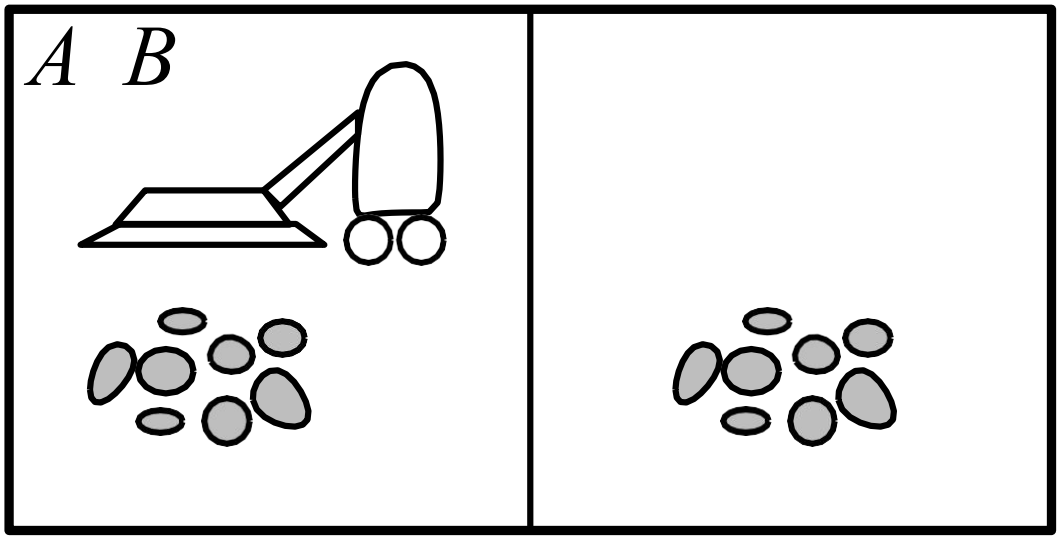
Agents include humans, robots, softbots, thermostats, etc.

The agent function maps from percept histories to actions:

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

The agent program runs on the physical architecture to produce f

Vacuum-cleaner world



Percepts: location and contents, e.g., [*A, Dirty*]

Actions: *Left*, *Right*, *Suck*, *NoOp*

A vacuum-cleaner agent

Percept sequence	Action
[*A, Clean*]	*Right*
[*A, Dirty*]	*Suck*
[*B, Clean*]	*Left*
[*B, Dirty*]	*Suck*
[*A, Clean*], [*A, Clean*]	*Right*
[*A, Clean*], [*A, Dirty*]	*Suck*
.	.

```
function Reflex-Vacuum-Agent( [location,status]) returns an action

    if status = Dirty then return Suck else if
    location = A then return Right else if
    location = B then return Left
```

What is the right function?
Can it be implemented in a small agent program?

PEAS

To design a rational agent, we must specify the task environment

Consider, e.g., the task of designing an

automated taxi: Performance measure??

Environment

??

Actuators??

Sensors??

PEAS

To design a rational agent, we must specify the task environment

Consider, e.g., the task of designing an automated taxi:

Performance measure?? safety, destination, profits, legality, comfort, . . .

Environment?? US streets/freeways, traffic, pedestrians, weather, . . .

Actuators?? steering, accelerator, brake, horn, speaker/display, . . .

Sensors?? video, accelerometers, gauges, engine sensors, keyboard, GPS, . . .

Internet shopping agent

Performance measure??

Environment

??

Actuators??

Sensors??

Internet shopping agent

Performance measure?? price, quality, appropriateness, efficiency
Environment?? current and future WWW sites, vendors, shippers Actuators?? display to user, follow URL, fill in form

Sensors?? HTML pages (text, graphics, scripts)

Environment types

Fully Observable vs. Partially Observable Environments

- **Fully Observable**: Agent's sensors provide complete state information at all times.

  Convenience: No need for internal state tracking.
- **Partially Observable**: Limited or noisy sensor data, missing state information.
- **Examples**:
  - **Fully Observable**: Chessboard (all pieces visible).
  - **Partially Observable**: Vacuum agent with local dirt sensor, self-driving car unable to see driver intentions.
- **Unobservable**: No sensor input; still, goals may sometimes be achievable.

Environment types

Single Agent vs. Multiagent Environments

- **Single-Agent**: Only one agent optimizing its performance (e.g., solving a crossword puzzle).
- **Multiagent**: Multiple agents whose actions impact each other (e.g., chess, taxi driving).
- **Key Distinction**: Does another entity maximize its performance based on the agent's behaviour?
- **Types of Multiagent Environments**:
  - **Competitive**: Agents have opposing goals (e.g., chess).
  - **Cooperative**: Agents share goals (e.g., avoiding collisions in traffic).
  - **Mixed**: Both cooperative and competitive elements (e.g., competing for parking spaces).
- **Challenges in Multiagent Systems**:
  - Need for communication in cooperative settings.
  - Randomized behaviour can prevent predictability in competitive scenarios.

Environment types

Deterministic vs. Stochastic Environments

- **Deterministic**: The next state is fully determined by the current state and agent's action.
- **Stochastic**: The next state has uncertainty, often modeled with probabilities.
- **Fully Observable & Deterministic**: No uncertainty for the agent.
- **Partially Observable Environments**: May appear stochastic due to missing information.
- **Examples**:
  - **Deterministic**: Classic vacuum world (fixed dirt and reliable suction).
  - **Stochastic**: Taxi driving (unpredictable traffic, tire blowouts).
- **Uncertain Environments**: Either partially observable or non-deterministic.
- **Nondeterministic vs. Stochastic**: Nondeterministic environments specify possible outcomes but not probabilities.

Environment types

Episodic vs. Sequential Environments

- **Episodic**: The agent's experience is divided into independent episodes.
  - Each episode consists of a percept and a single action.
  - Past actions do not affect future episodes.
  - **Example**: Detecting defective parts on an assembly line.
- **Sequential**: The current decision impacts future decisions.
  - Actions have long-term consequences.
  - **Examples**: Chess, taxi driving.
- **Complexity**: Episodic environments are simpler as they don't require planning ahead.

Environment types

Static vs. Dynamic Environments

- **Static Environment**
  - No changes unless the agent acts.
  - Predictable and stable.
  - Lower complexity.
  - **Example**: Chessboard (positions remain fixed until a move is made).
- **Dynamic Environment**
  - Changes occur autonomously.
  - Unpredictable and uncertain.
  - Requires continuous adaptation.
  - Higher complexity.
  - **Example**: Traffic conditions (affected by accidents, construction, rush hour).

Environment types

Discrete vs. Continuous Environments

- **Discrete Environment**:
  - Finite number of distinct states.
  - Time, percepts, and actions occur in discrete steps.
  - **Example**: Chess (finite board positions and moves).
- **Continuous Environment**:
  - Infinite possible states and smooth transitions over time.
  - Percepts and actions vary continuously.
  - **Example**: Taxi driving (speed, location, and steering angles change smoothly).

Environment types

The environment type largely determines the agent design

The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent

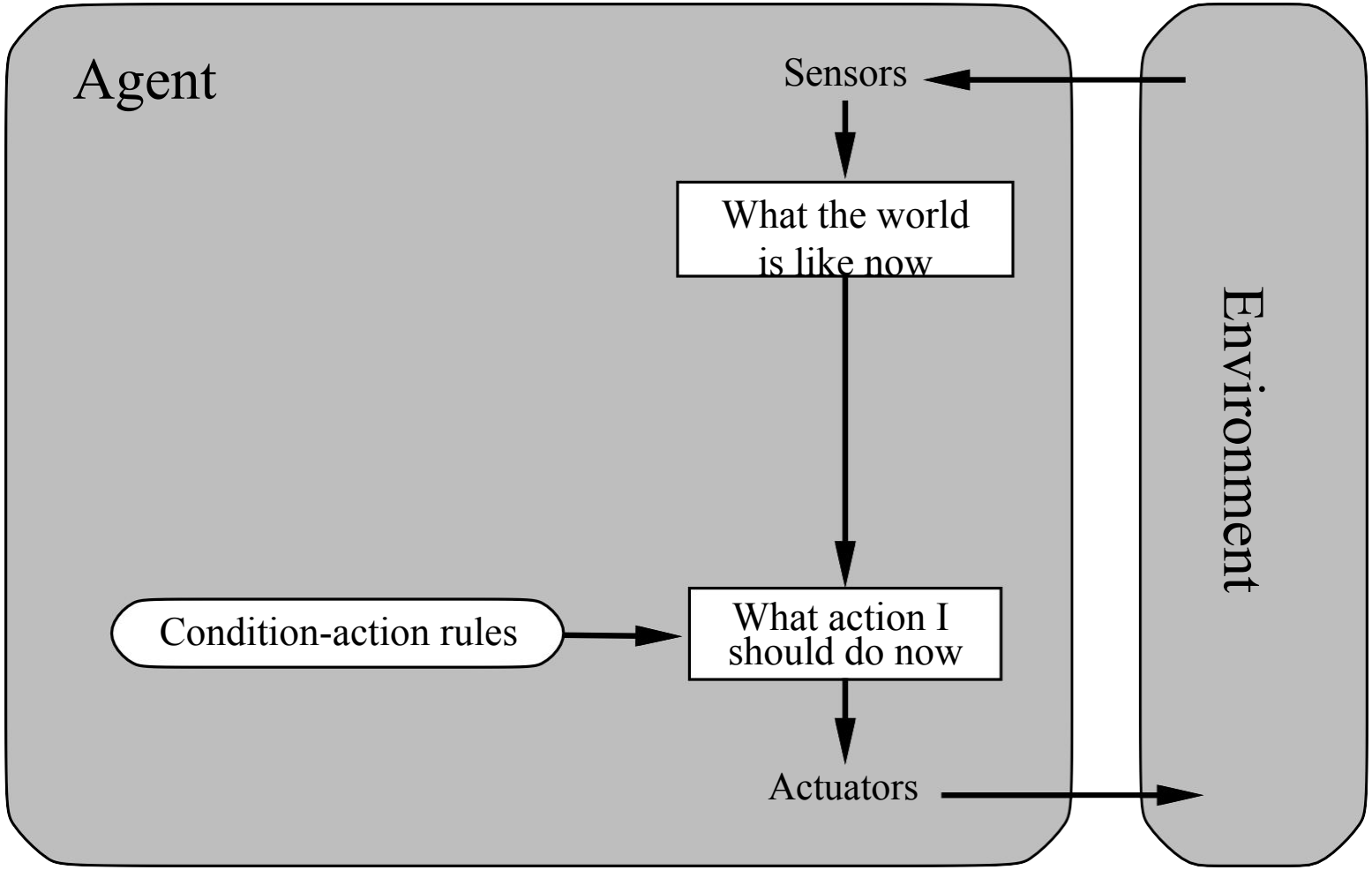
Agent types

Four basic types in order of increasing generality:

- simple reflex agents
- reflex agents with state
- goal-based agents
- utility-based agents

All these can be turned into learning agents

Simple reflex agents



Example

function Reflex-Vacuum-Agent([*location*,*status*]) returns an action

if *status* = *Dirty* then return *Suck* else if
location = *A* then return *Right* else if
location = *B* then return *Left*

Problems with simple reflex agents

Simple reflex agents fail in partially observable environments

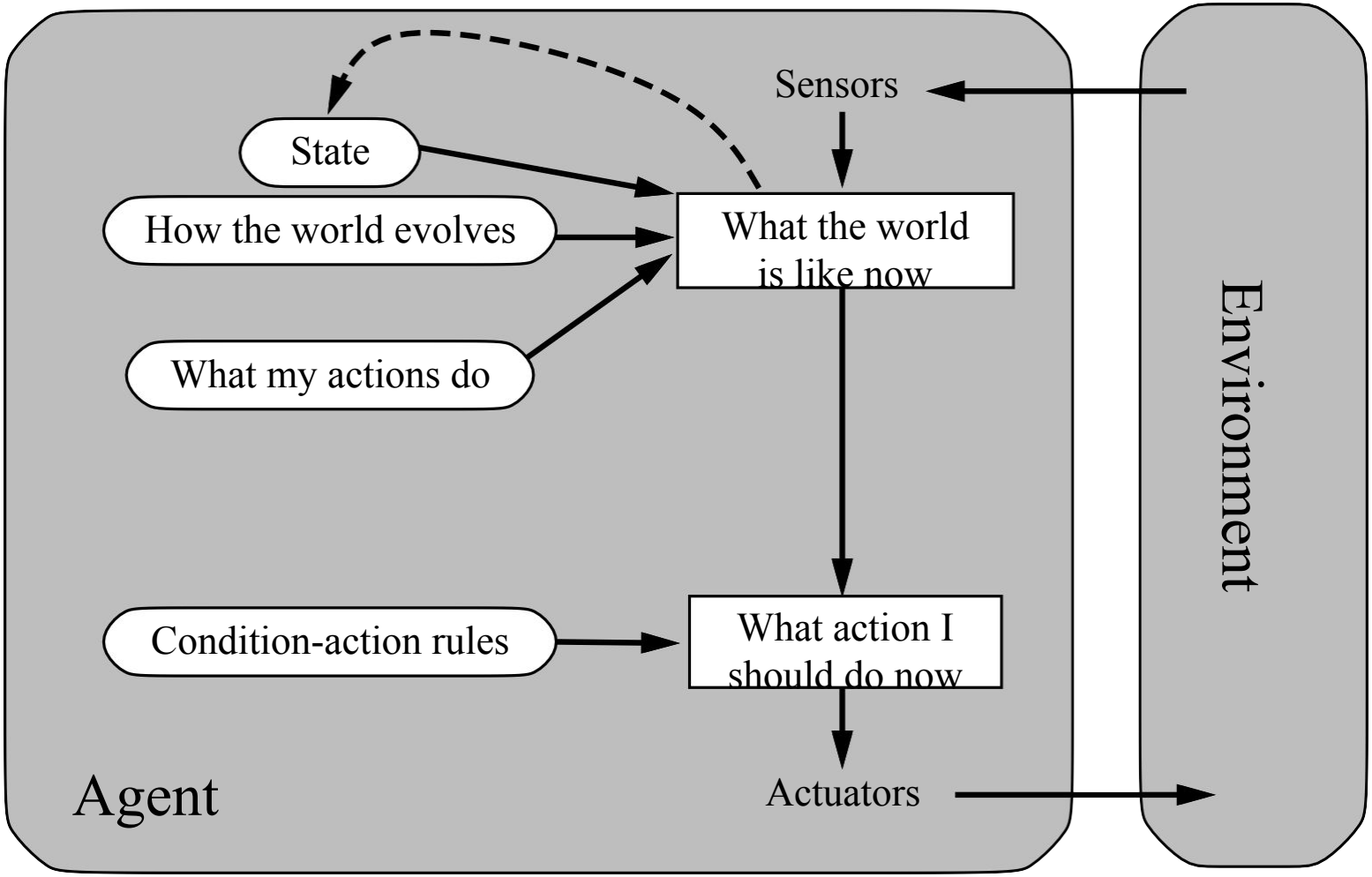
E.g., suppose location sensor is missing

Agent (presumably) *Suck*s if *Dirty*; what if *Clean*?

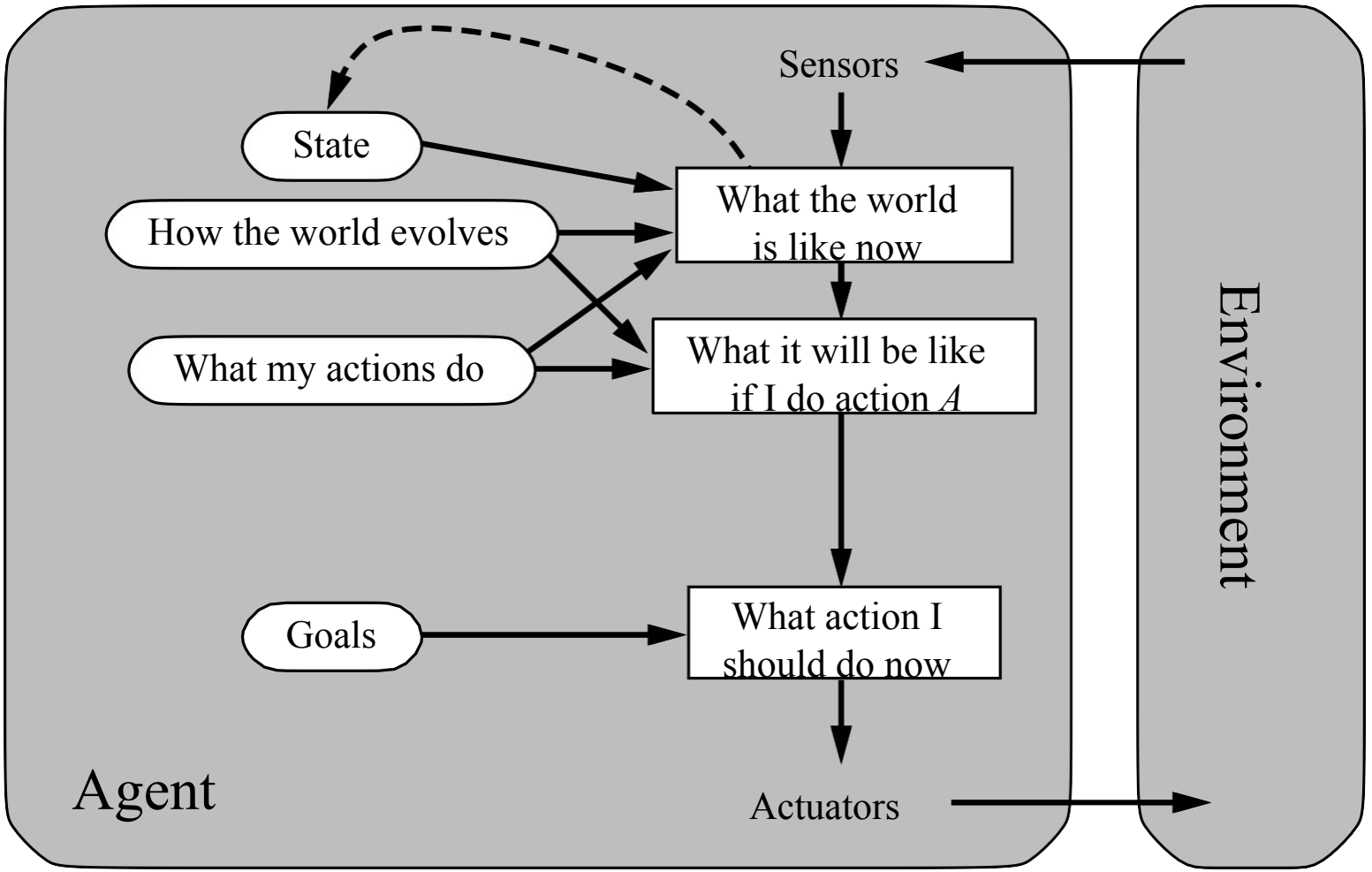
$\Rightarrow$ infinite loops are unavoidable

Randomization helps (why??), but not that much

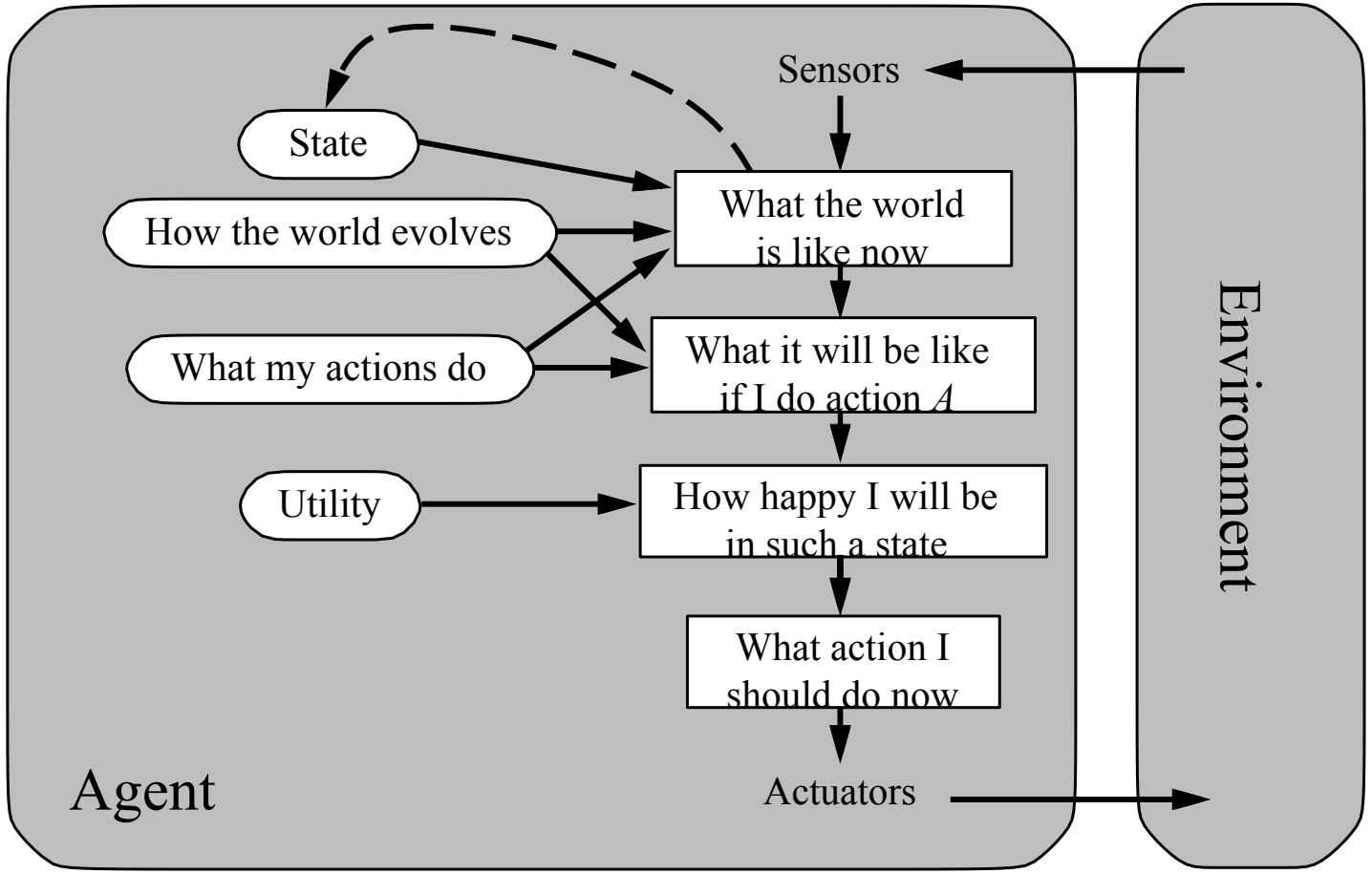
Reflex agents with state



Goal-based agents



Utility-based agents



Summary

Agents interact with environments through actuators and sensors
The agent function describes what the agent does in all

circumstances The performance measure evaluates the

environment sequence

A perfectly rational agent maximizes expected

performance Agent programs implement (some)

agent functions

PEAS descriptions define task environments

Environments are categorized along several

dimensions:

observable? deterministic? episodic? static? discrete?
single-agent?

Several basic agent architectures exist:
reflex, reflex with state, goal-based, utility-based