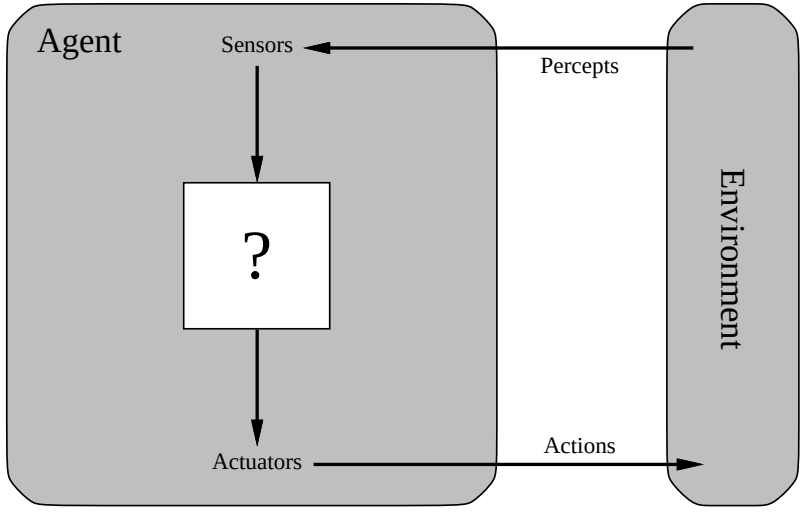
Intelligent Agents

Chapter 2

Outline

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

Agents and environments



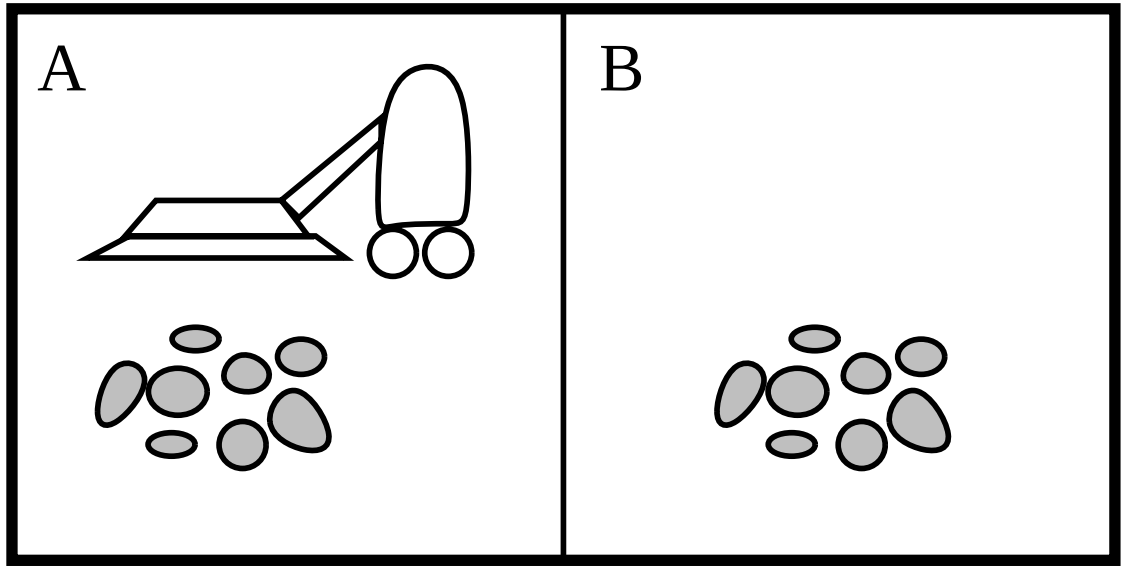
Agents include humans, robots, softbots, thermostats, etc.

The *agent function* maps from percept histories to actions:

$$f : \mathcal{P}^* \rightarrow \mathcal{A}$$

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
$[A, Clean]$	$Right$
$[A, Dirty]$	$Suck$
$[B, Clean]$	$Left$
$[B, Dirty]$	$Suck$
$[A, Clean], [A, Clean]$	$Right$
$[A, Clean], [A, Dirty]$	$Suck$
⋮	⋮

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

A vacuum-cleaner agent

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?

Rationality

- Fixed *performance measure* evaluates the environment sequence
  - one point per square cleaned up in time T?
  - one point per clean square per time step, minus one per move?
  - penalize for $> k$ dirty squares?
- *rational agent* chooses whichever action maximizes the *expected* value of the performance measure *given the percept sequence to date*
- Rational $\neq$ omniscient Rational $\neq$ clairvoyant
  Rational $\neq$ successful
- Rational $\Longrightarrow$ exploration, learning, autonomy

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:

Environment types

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??				
Deterministic??				
Episodic??				
Static??				
Discrete??				
Single-agent??				

Fully Observable: access to the complete (relevant) state of the world

Partially Observable: missing information

Environment types

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	yes (?)	yes	no	no
Deterministic??				
Episodic??				
Static??				
Discrete??				
Single-agent??				

Deterministic: the next state is completely determined by the current state and the action

Stochastic: Changes not known

Strategic: Deterministic except for the actions of the other agents

Environment types

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	yes (?)	yes	no	no
Deterministic??	yes	no	partly	no
Episodic??				
Static??				
Discrete??				
Single-agent??				

Episodic: task divided into atomic episodes

Sequential: Current decision may affect all future decisions

Environment types

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	yes (?)	yes	no	no
Deterministic??	yes	no	partly	no
Episodic??	no	no	no	no
Static??				
Discrete??				
Single-agent??				

Static: the world does not change while the agent is thinking

Dynamic: changes

Semidynamic: does not change but the performance is affected as time passes

Environment types

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	yes (?)	yes	no	no
Deterministic??	yes	no	partly	no
Episodic??	no	no	no	no
Static??	yes	semi	no	no
Discrete??				
Single-agent??				

Discrete: time, percepts, and actions are discrete

Continuous: time, percepts, and actions are continuous over time

Environment types

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	yes (?)	yes	no	no
Deterministic??	yes	no	partly	no
Episodic??	no	no	no	no
Static??	yes	semi	no	no
Discrete??	yes	yes	yes	no
Single-agent??				

Single-agent: one agent

Multi-agent: competitive or cooperating agents

Environment types

	Solitaire	Backgammon	Internet shopping	Taxi
Observable??	yes (?)	yes	no	no
Deterministic??	yes	no	partly	no
Episodic??	no	no	no	no
Static??	yes	semi	no	no
Discrete??	yes	yes	yes	no
Single-agent??	yes	no	yes (?)	no

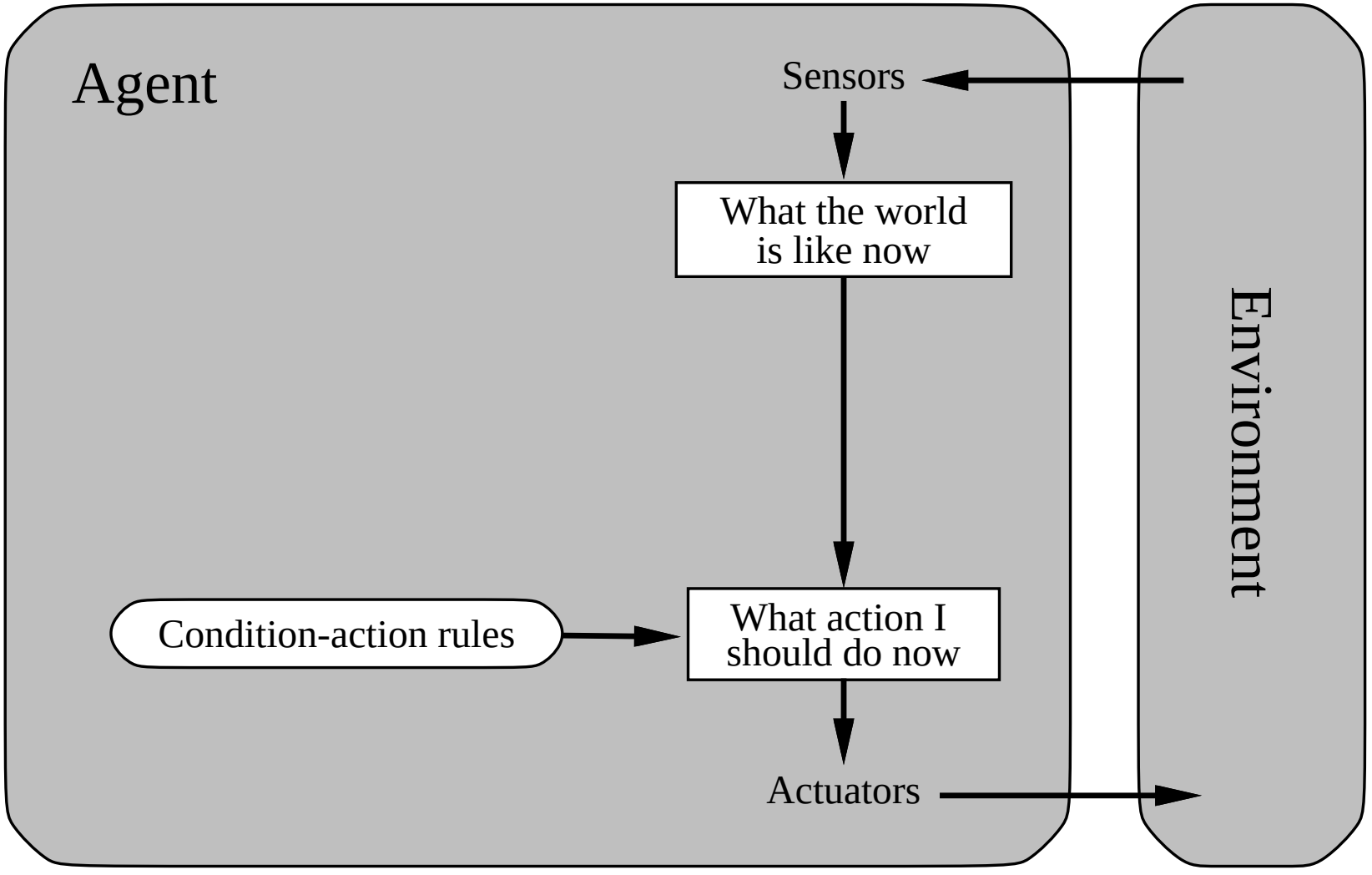
The environment type largely determines the agent design

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

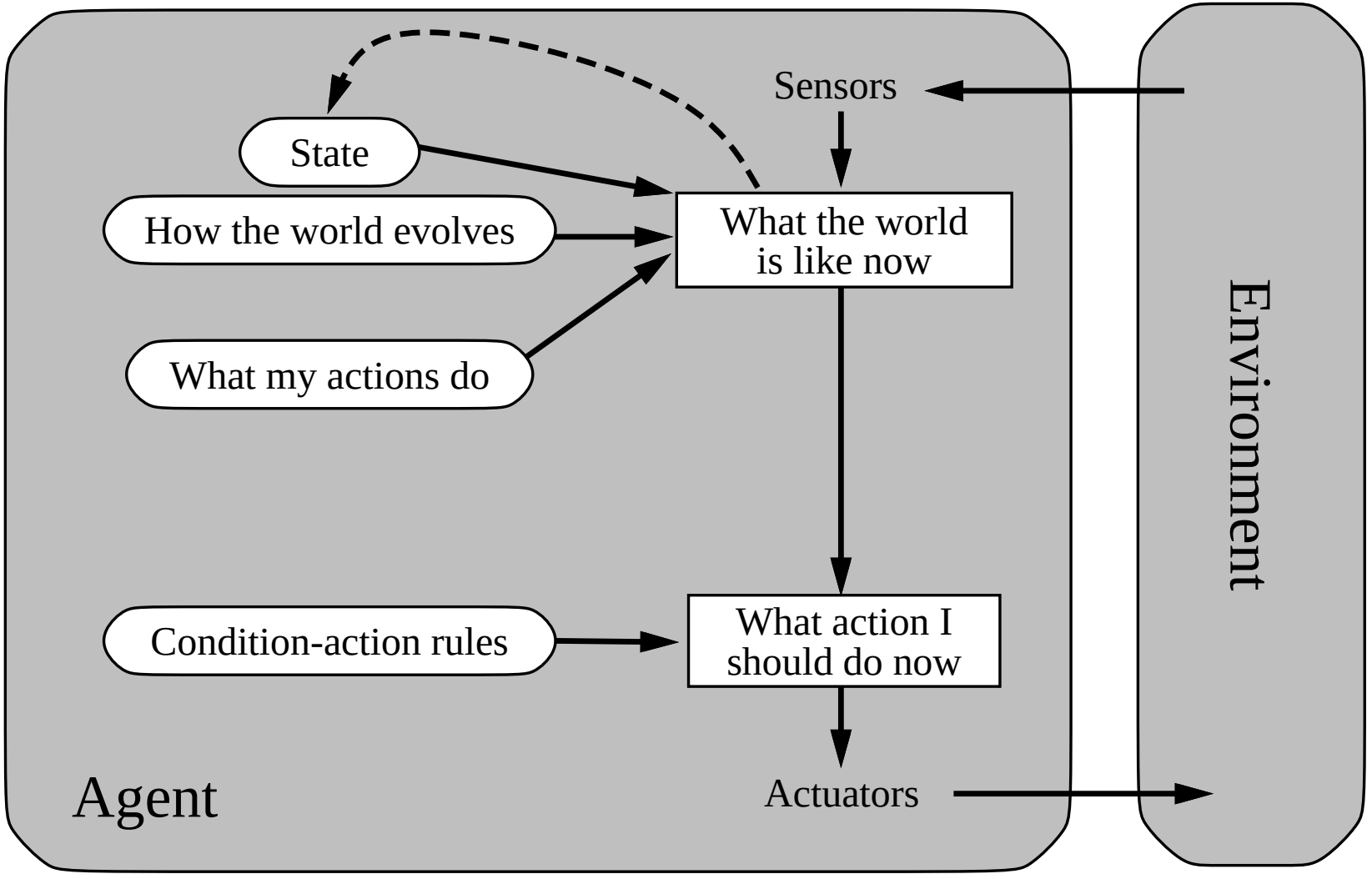
Agent types

- 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

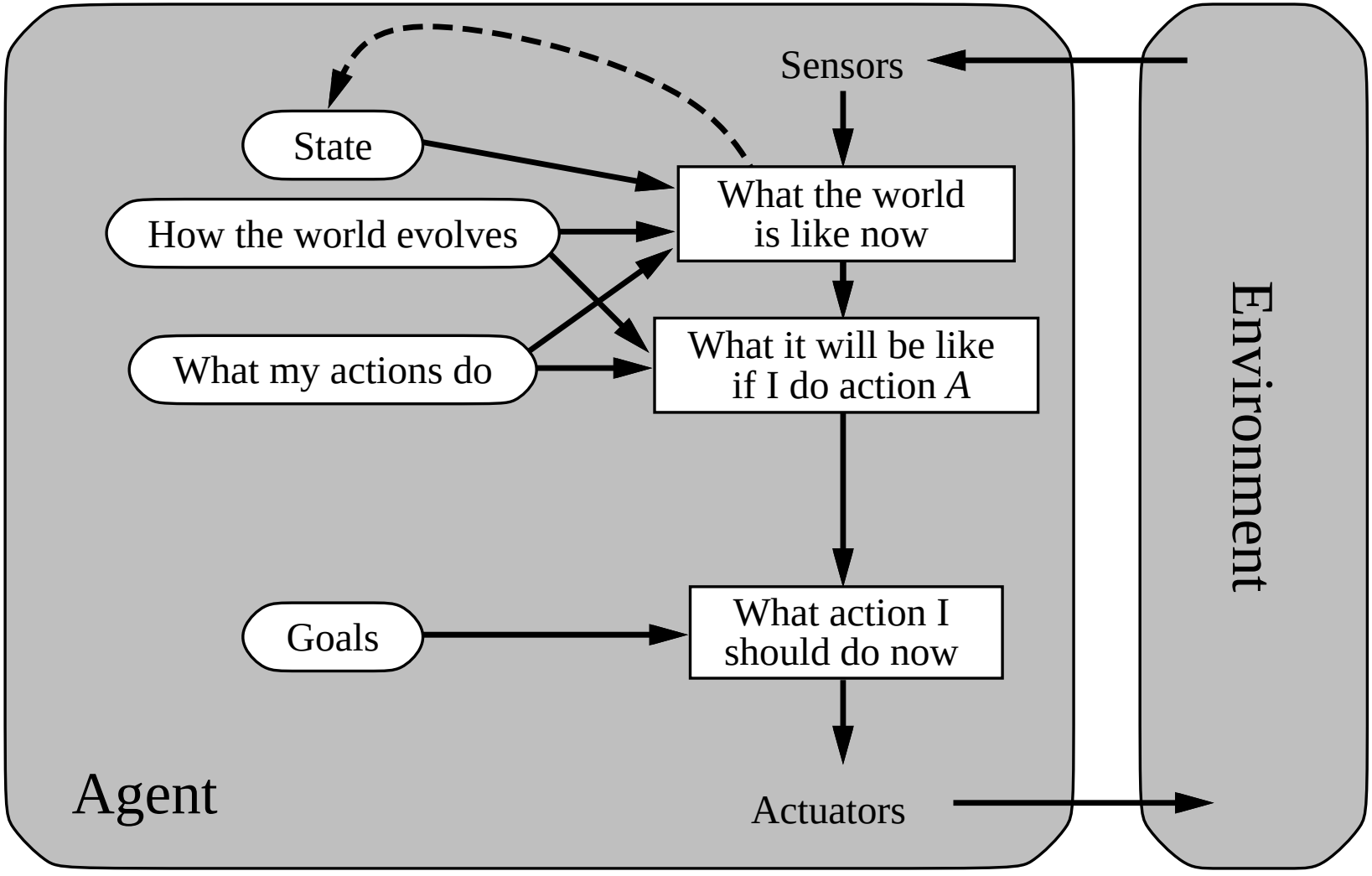
Simple reflex agents



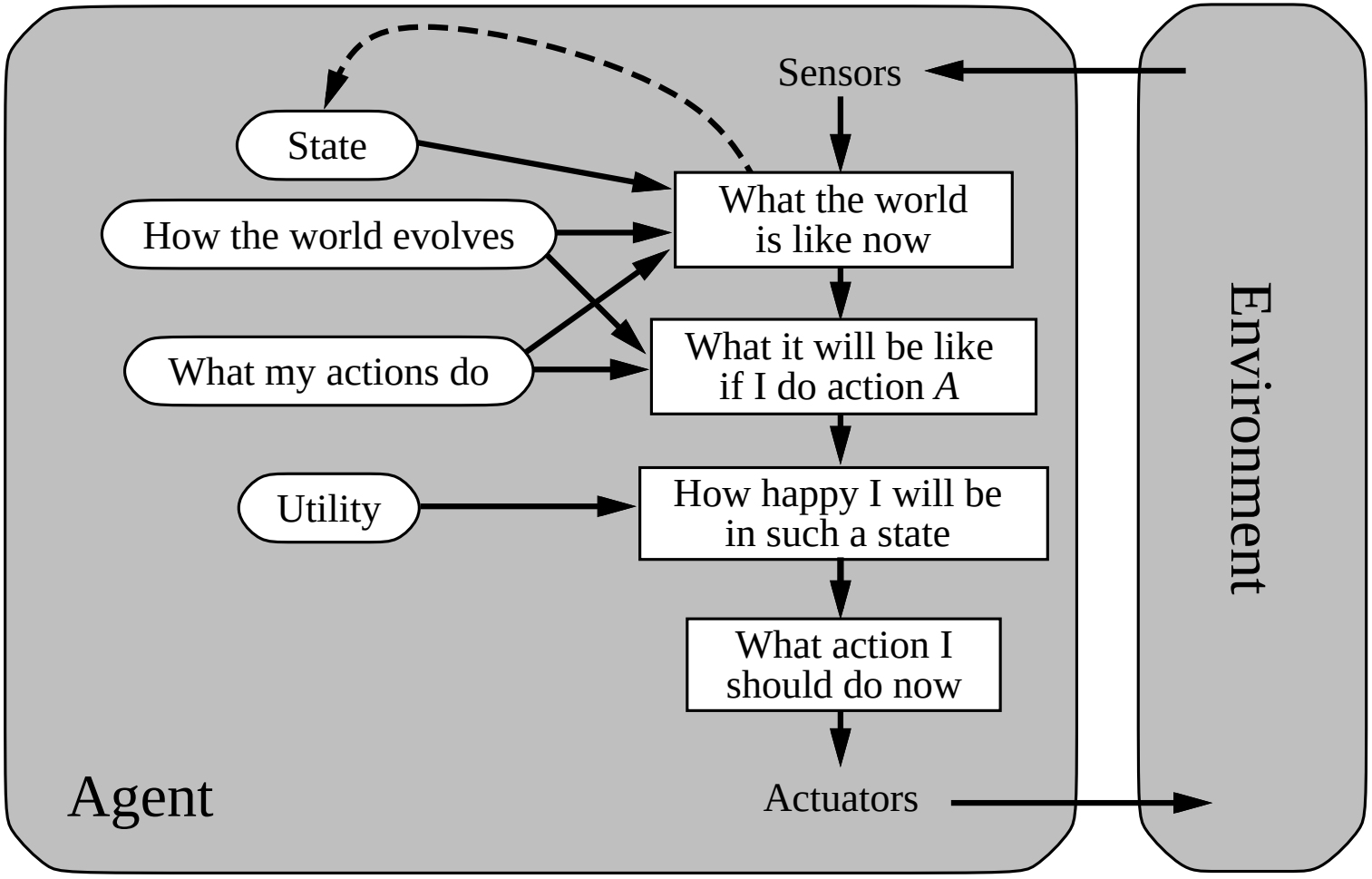
Reflex agents with state



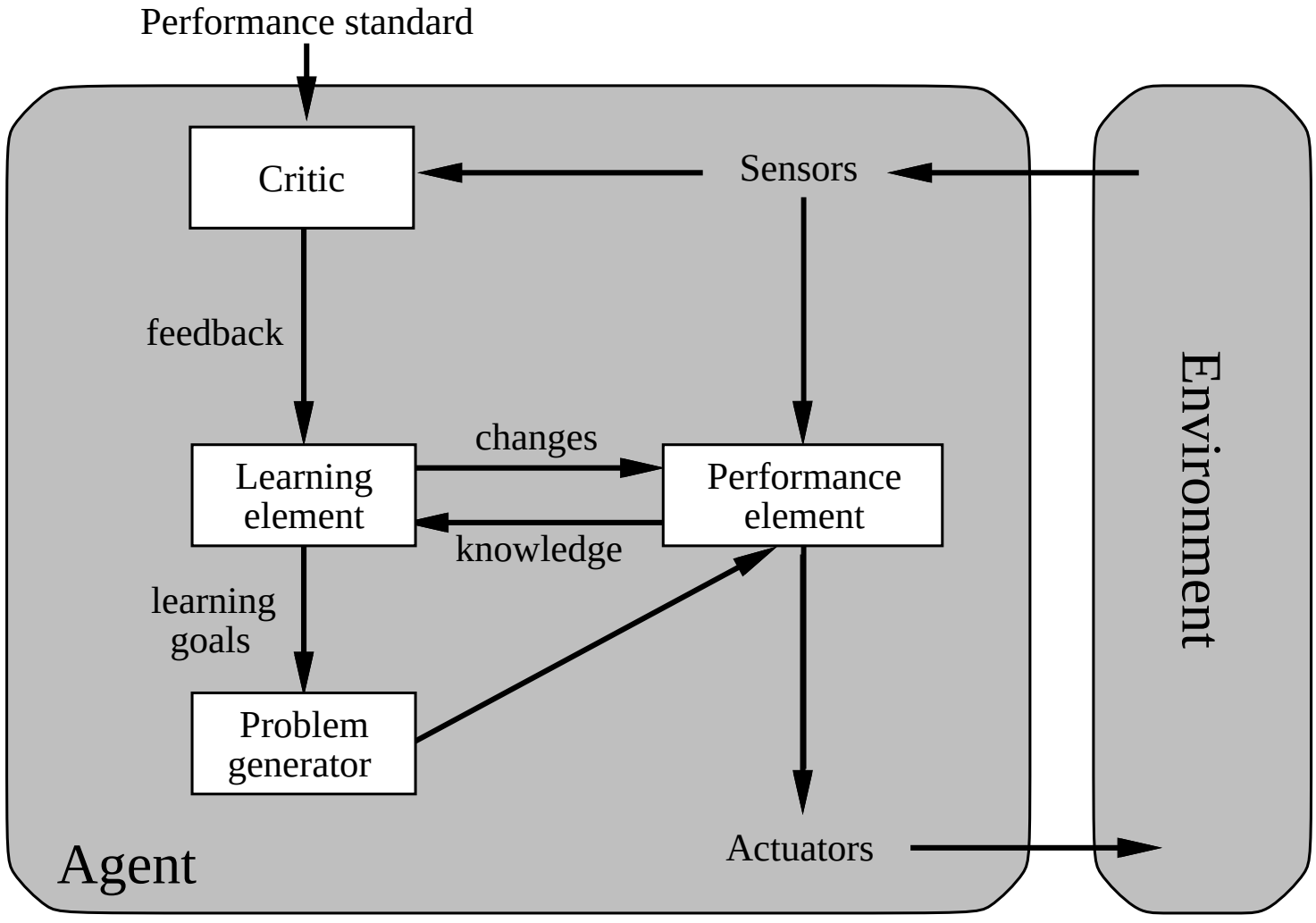
Goal-based agents



Utility-based agents



Learning agents



AIMA code

- The code for each topic is divided into four directories:
  - `agents`: code defining agent types and programs
  - `algorithms`: code for the methods used by the agent programs
  - `environments`: code defining environment types, simulations
  - `domains`: problem types and instances for input to algorithms
- Often run algorithms on domains rather than agents in environments.

AIMA code

located in:/classes/cs5811/common/aima-code/

```
(setq joe (make-agent
  :name 'joe :body (make-agent-body)
  :program (make-dumb-agent-program)))

(defun make-dumb-agent-program ()
  (let ((memory nil))
    #'(lambda (percept)
        (push percept memory)
        'no-op)))
```