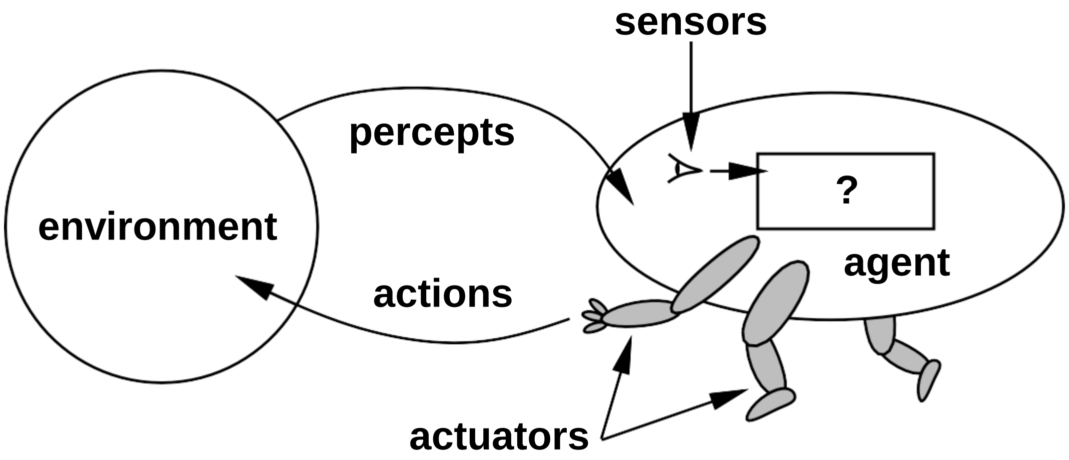
Intelligent Agents

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

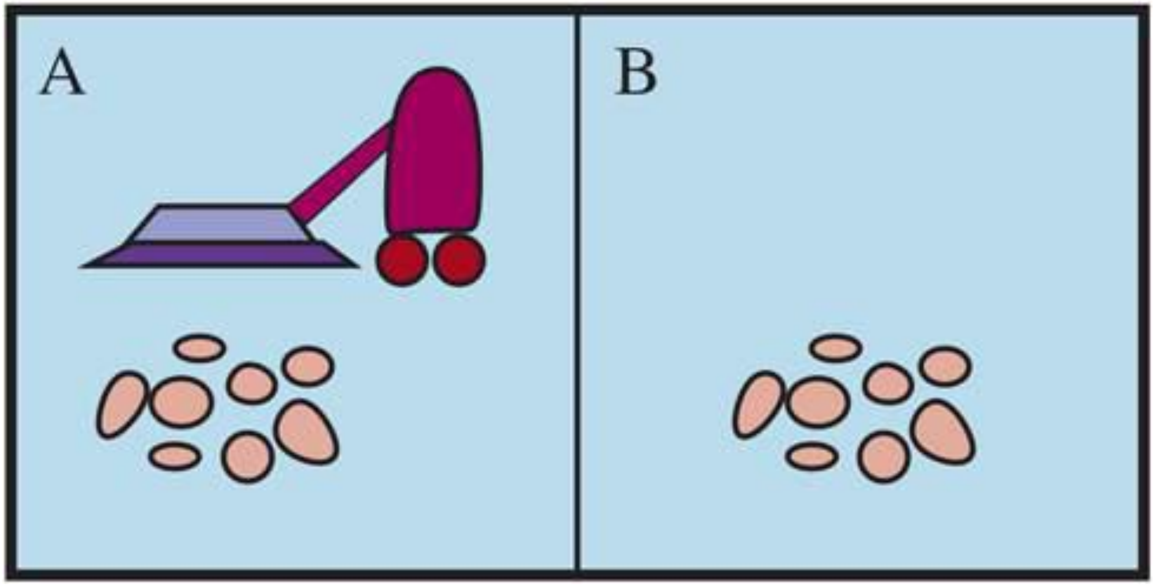
Agents and environments



- Agents include humans, robots, softbots, thermostats, etc.
- An agent can be anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators
- 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?

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?
- A rational agent chooses whichever action maximizes the expected value of the performance measure given the percept sequence to date
- Rational /= omniscient
  - percepts may not supply all relevant information
- Rational /= clairvoyant
  - action outcomes may not be as expected
- Hence, rational /= successful
- Rational ⇒ exploration, learning, autonomy

PEAS

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

Types of environments

- Partially observable: agent does not have full information about the state
- Fully observable: agent has full relevant information about their state
- Unobservable: the agent has no sensors at all
- Stochastic: uncertainty in the transition model
- Deterministic: taking an action in a state has a single outcome
- Single-agent: agent acts in the environments by itself
- Multi-agent (competitive or cooperative): agent acts in the environments along with other agents (adversaries or partners)

Types of environments

- Episodic: agent's experience is divided into atomic episodes
- Sequential: current decision could affect all future decisions
- Static: the environment does not change as the agent acts on it
- Dynamic: environments that change as the agent interacts with it
- Semidynamic: environment does not change but agent's performance score changes
- Discrete vs Continuous: distinction applies to the *state* of the environment, to the way *time* is handled, and to the *percepts* and actions of the agent
- Known physics: transition model is known to the agent and it can use that when planning a path
- Physics are unknown: the agent will need to take actions deliberately to learn the unknown dynamics

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	Semi	No
Discrete??	Yes	Yes	Yes	
Single-agent??	No Yes	No	Yes (except auctions)	No

The environment type largely determines the agent design

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

PEAS Example - automated taxi

- Consider, e.g., the task of designing an automated taxi:
- Performance measure??
- Environment??
- Actuators??
- Sensors??

PEAS Example - automated taxi

- 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, . . .

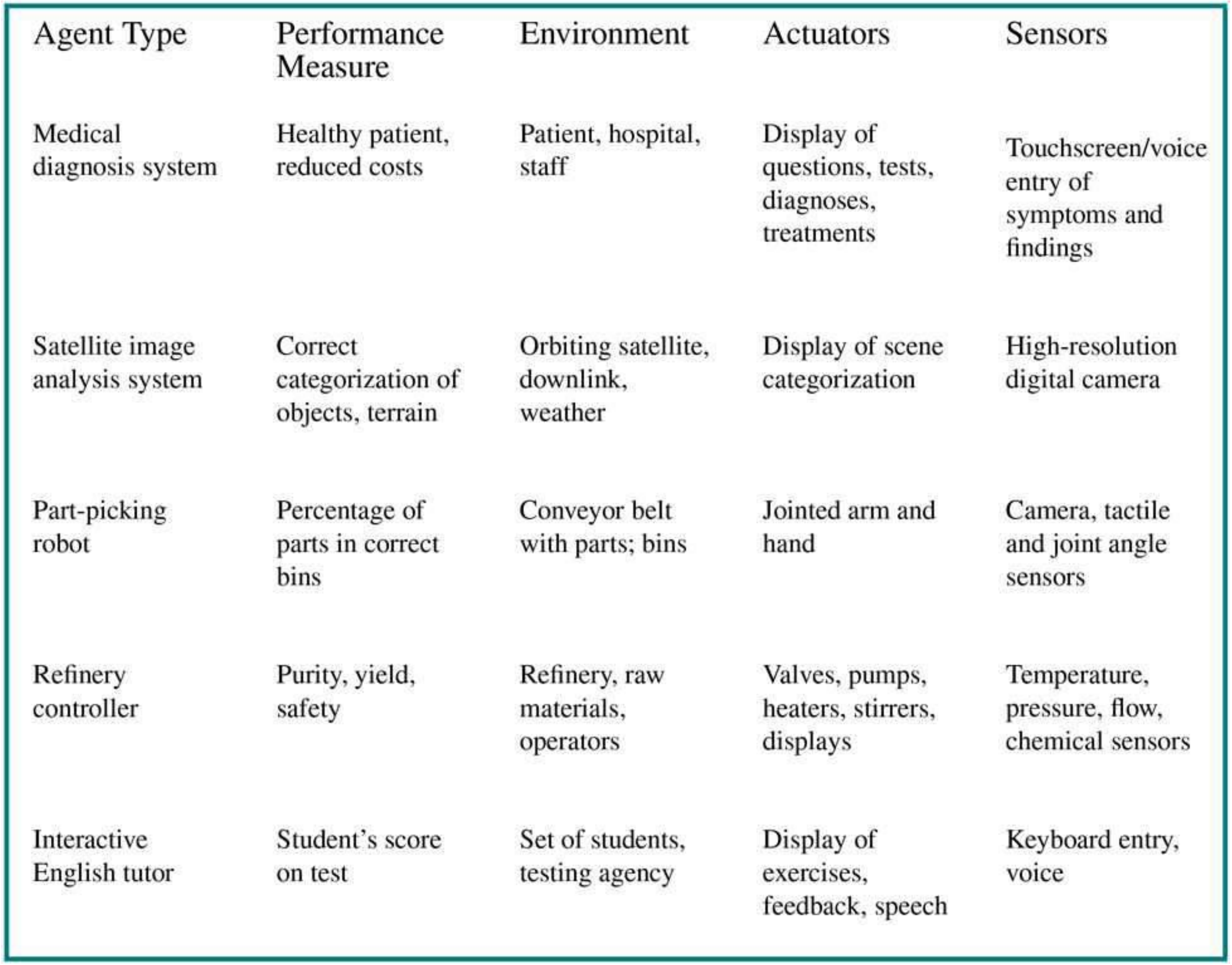
PEAS Example - Internet shopping agent

- Performance measure??
- Environment??
- Actuators??
- Sensors??

PEAS Example - 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)

PEAS Other Examples

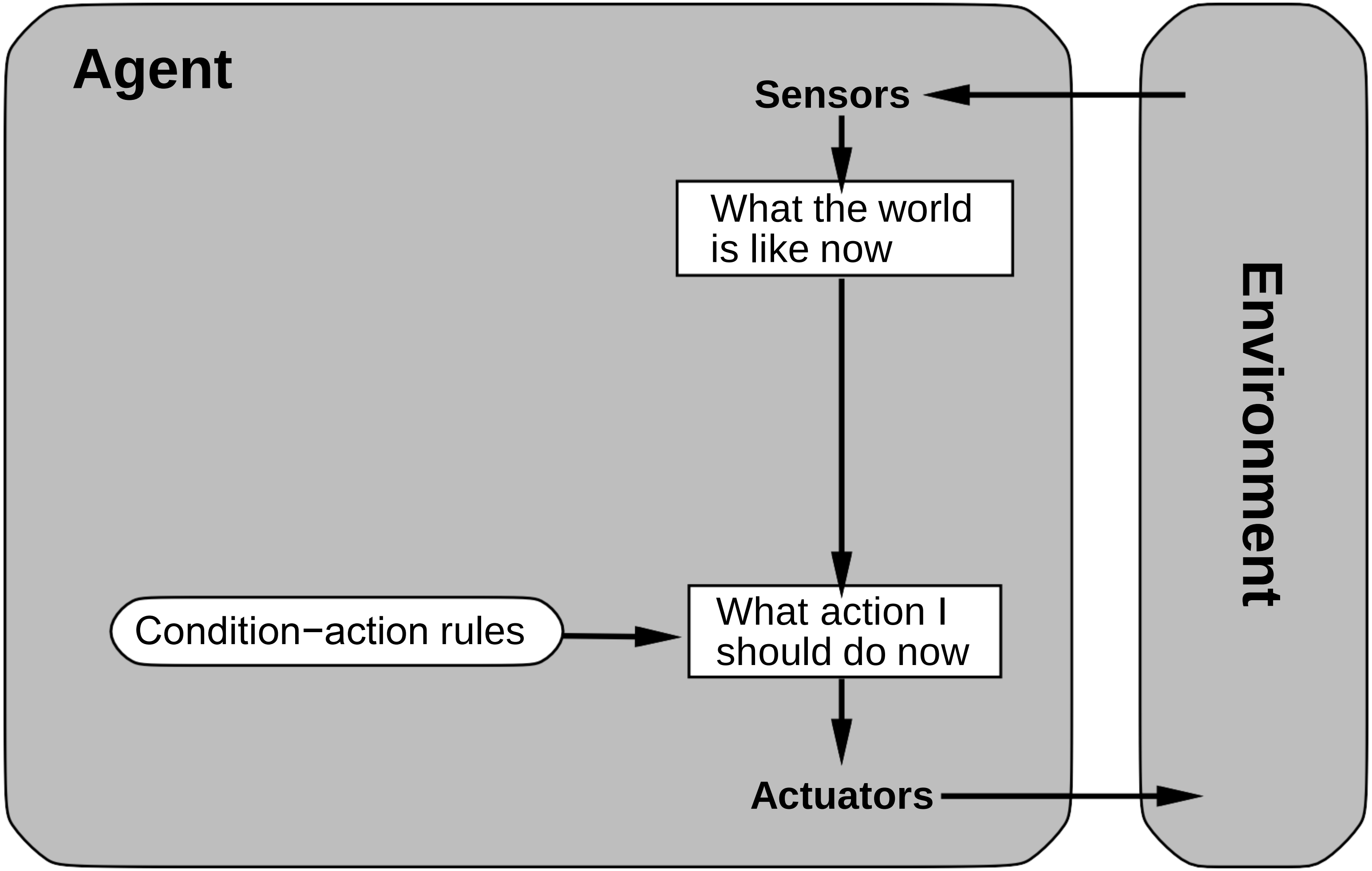


Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient, reduced costs	Patient, hospital, staff	Display of questions, tests, diagnoses, treatments	Touchscreen/voice entry of symptoms and findings
Satellite image analysis system	Correct categorization of objects, terrain	Orbiting satellite, downlink, weather	Display of scene categorization	High-resolution digital camera
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, tactile and joint angle sensors
Refinery controller	Purity, yield, safety	Refinery, raw materials, operators	Valves, pumps, heaters, stirrers, displays	Temperature, pressure, flow, chemical sensors
Interactive English tutor	Student's score on test	Set of students, testing agency	Display of exercises, feedback, speech	Keyboard entry, voice

Agent types

- Basic types in order of increasing generality:
  - simple reflex agents
  - reflex agents with state
  - model-based agents
  - goal-based agents
  - utility-based agents
- All these can be turned into learning agents

Simple reflex agents

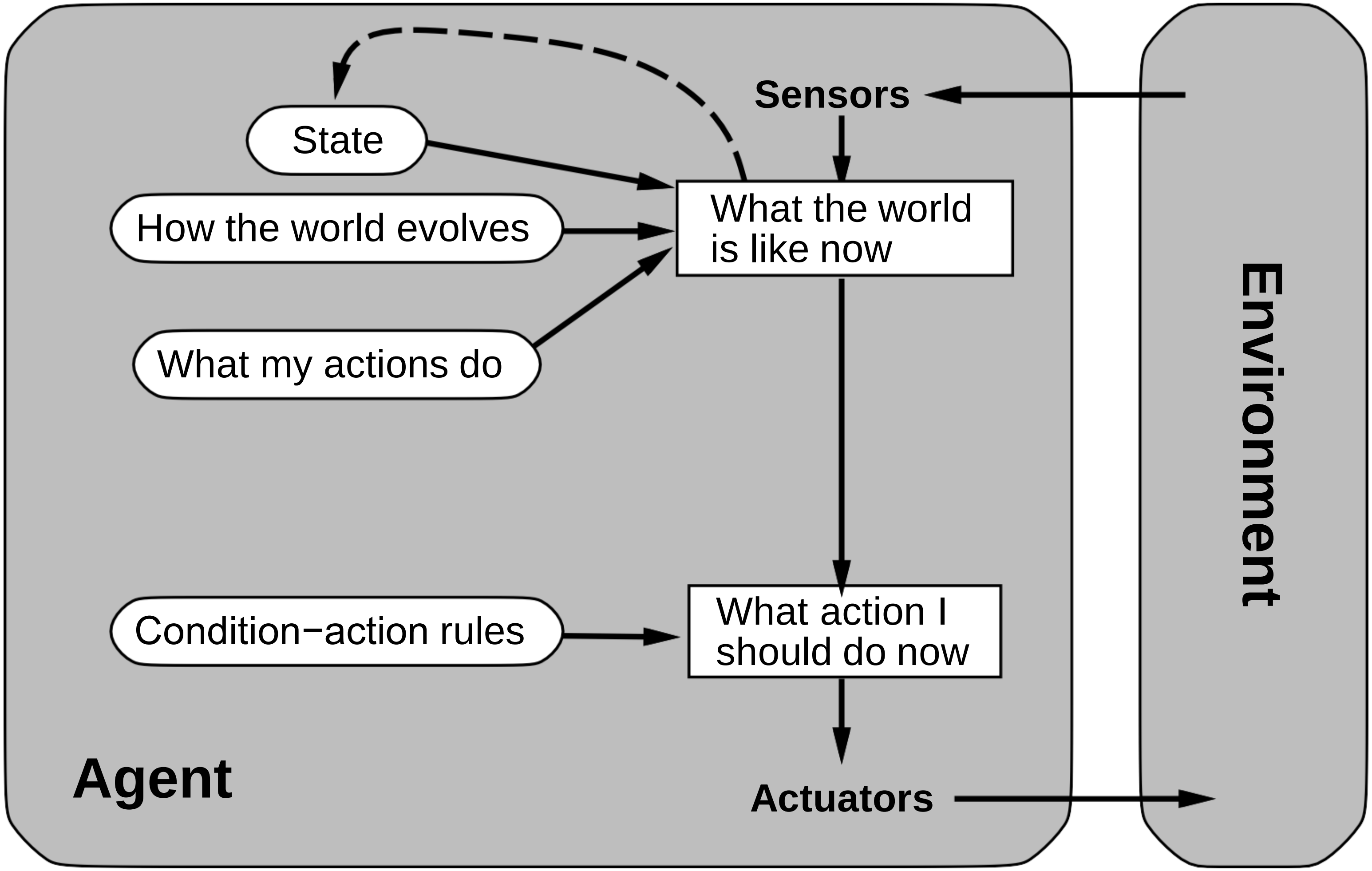


Example

```
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
```

```
(setq joe (make-agent :name ’joe :body (make-agent-body)
                      :program (make-reflex-vacuum-agent-program))

(defun make-reflex-vacuum-agent-program ()
  #’(lambda (percept)
      (let ((location (first percept)) (status (second percept)))
        (cond ((eq status ’dirty) ’Suck)
              ((eq location ’A) ’Right)
              ((eq location ’B) ’Left)))))
```

Reflex agents with state

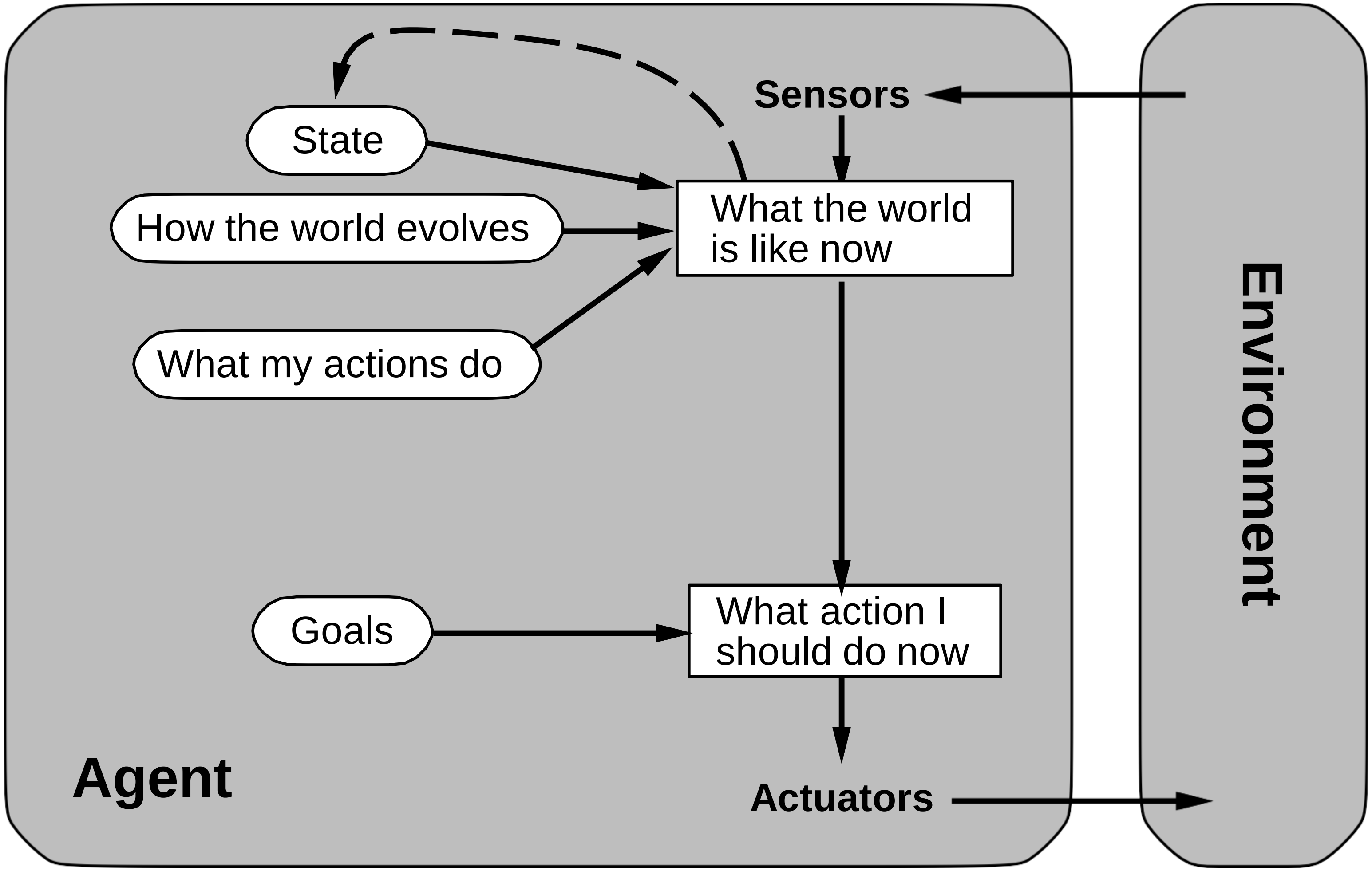


Example

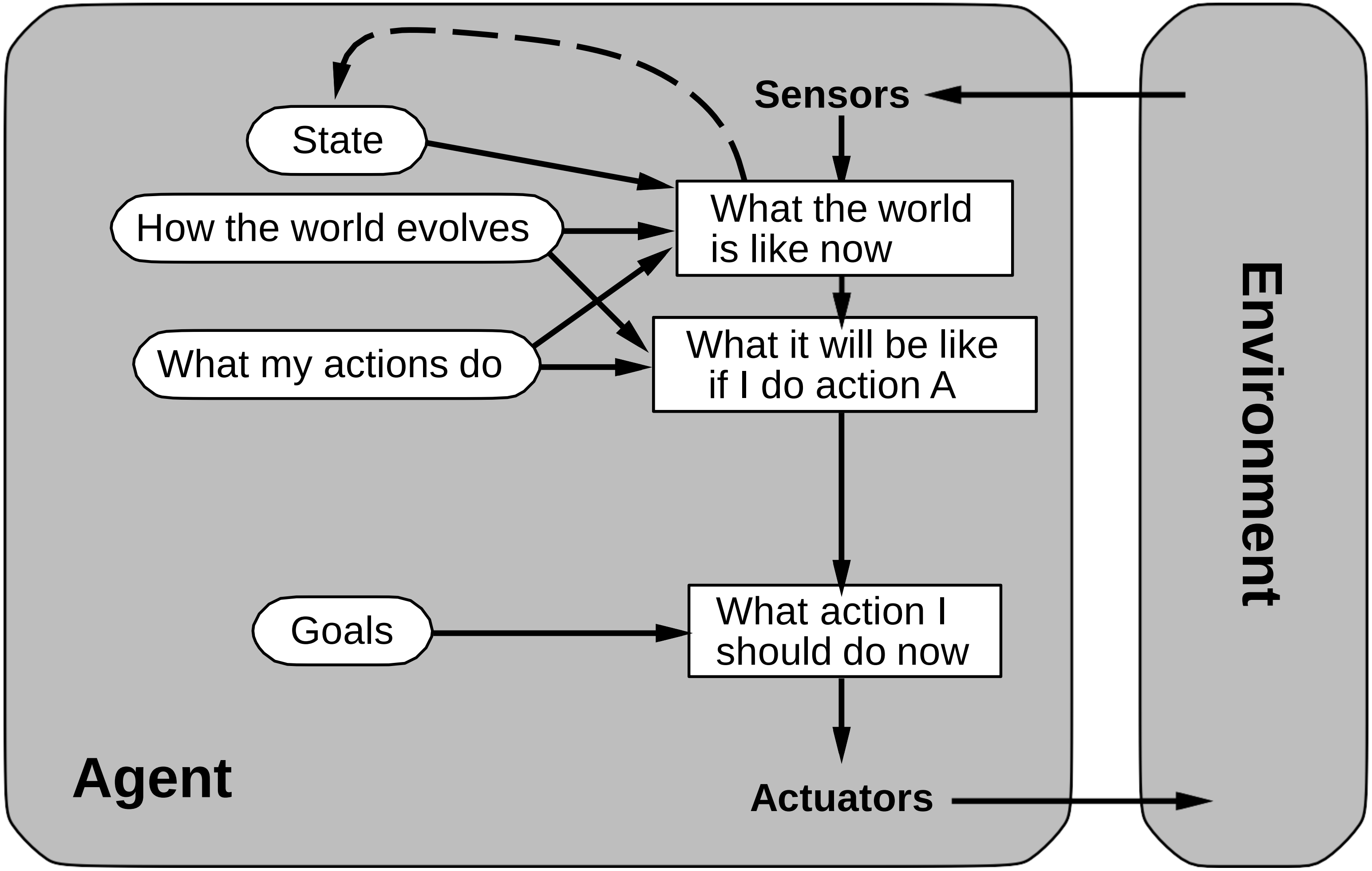
```
function Reflex-Vacuum-Agent( [location,status]) returns an action
static: last_A, last_B, numbers, initially ∞

  if status = Dirty then . . .
```

```
(defun make-reflex-vacuum-agent-with-state-program ()
  (let ((last-A infinity) (last-B infinity))
  #’(lambda (percept)
      (let ((location (first percept)) (status (second percept)))
        (incf last-A) (incf last-B)
        (cond
         ((eq status ’dirty)
          (if (eq location ’A) (setq last-A 0) (setq last-B 0))
          ’Suck)
         ((eq location ’A) (if (> last-B 3) ’Right ’NoOp))
         ((eq location ’B) (if (> last-A 3) ’Left ’NoOp)))))))
```

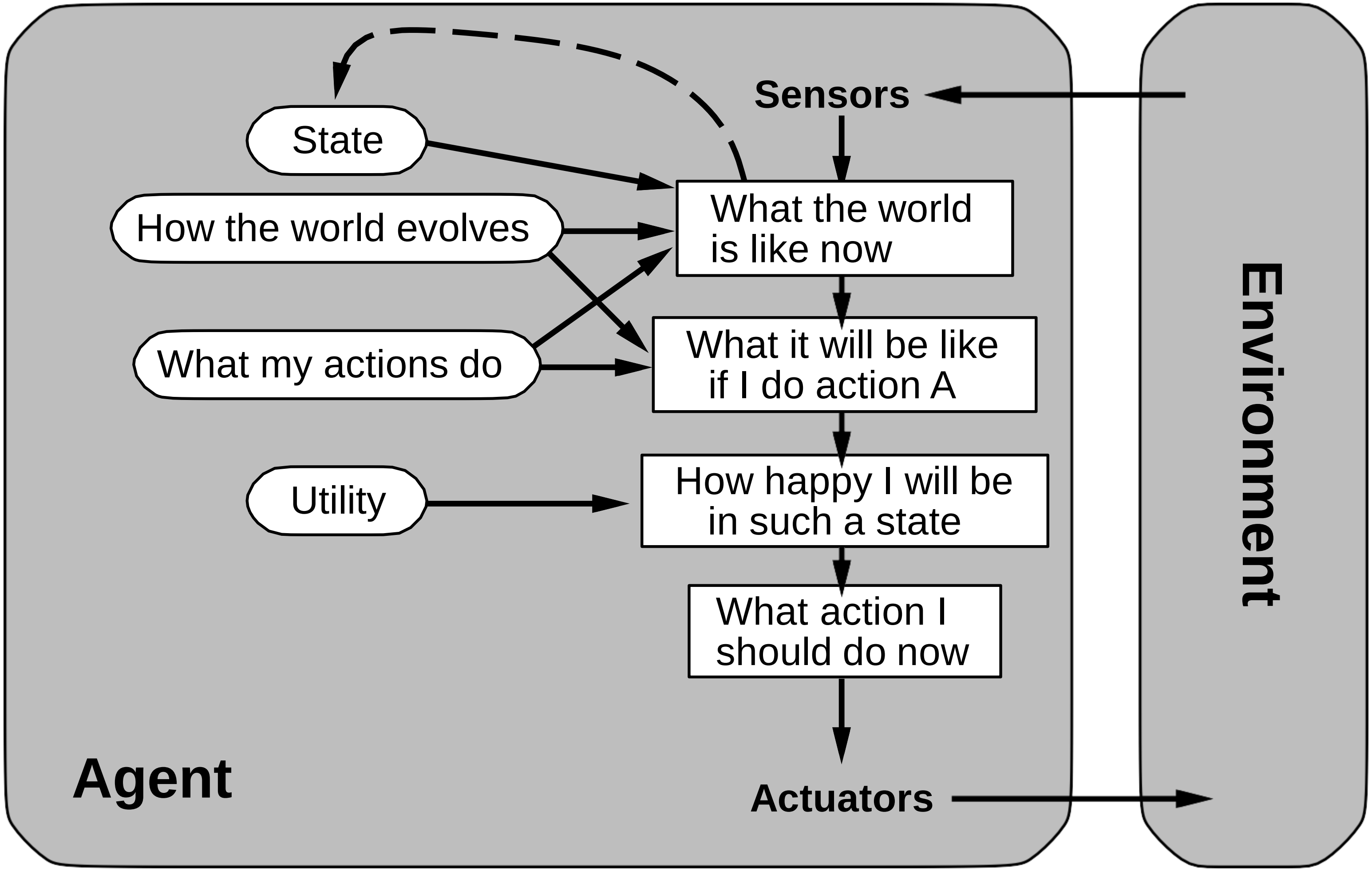
Model-based agents



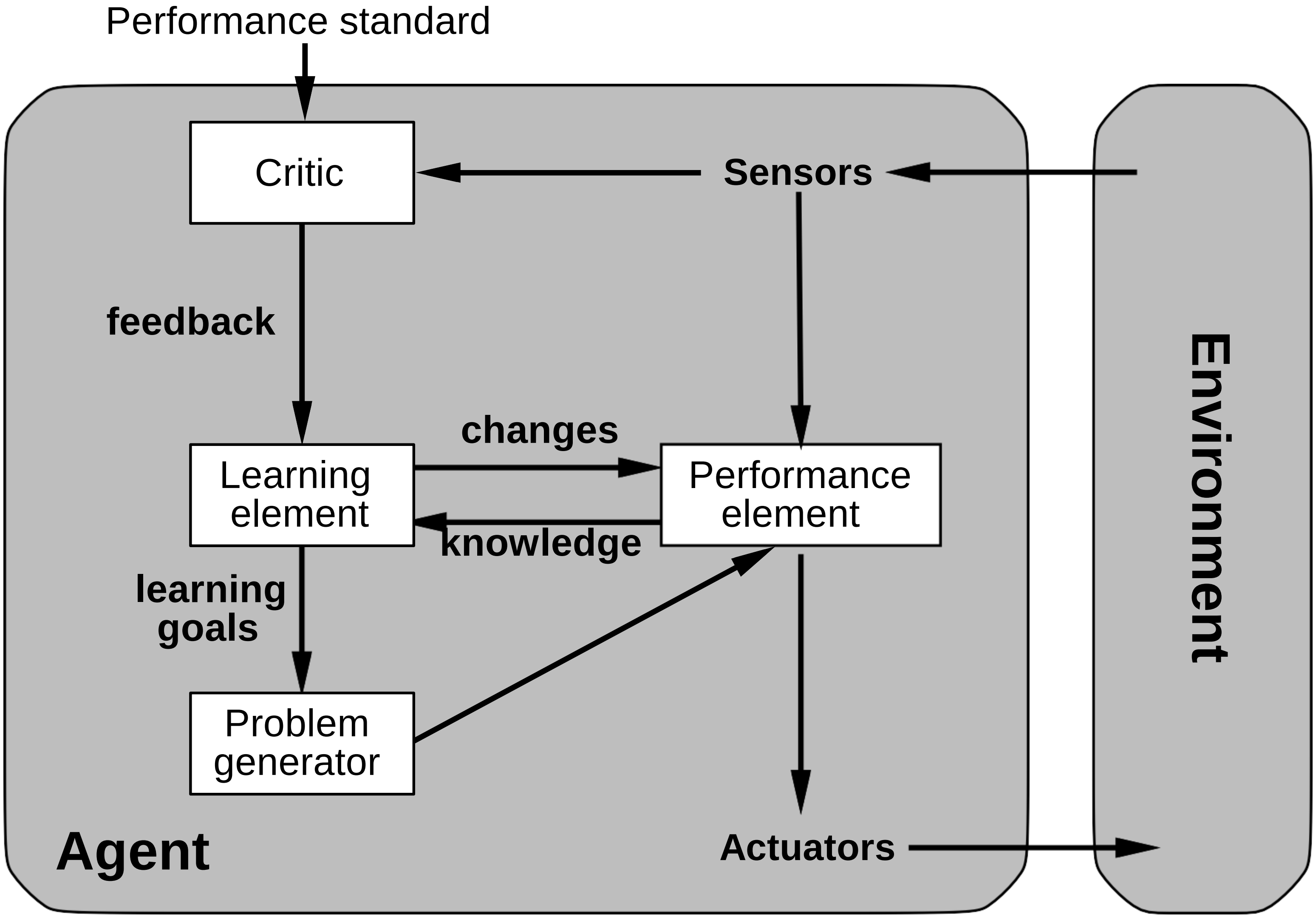
Goal-based agents


Sensors
State
How the world evolves
What my actions do
What the world is like now
What it will be like if I do action A
Goals
What action I should do now
Environment
Agent
Actuators

Utility-based agents


Sensors
State
How the world evolves
What my actions do
Utility
What the world is like now
What it will be like if I do action A
How happy I will be in such a state
What action I should do now
Actuators
Agent
Environment

Learning 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