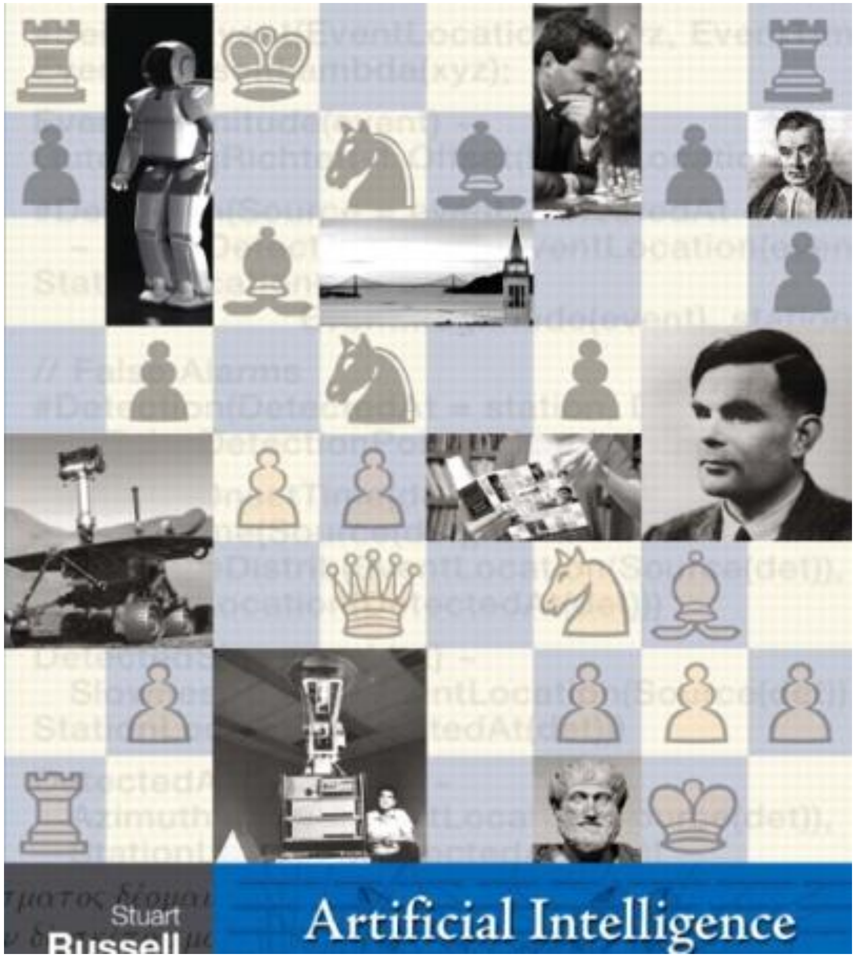
Lecture 2: Intelligent Agents

Russell and Norvig Chapter 2



CS-4820/5820

Tu/Th 12:15 PM-1:30 PM

Room: Centennial Hall 106

Instructor: Adham Atyabi

Office: Engineering 243

Office Hours: Mon 9:00 AM-14:00 PM.

Email: aatyabi@uccs.edu

Teaching Assistant: Ali Al Shami (aalshami@uccs.edu)

Outline

- Agents and environments
- Rationality
- PEAS (**P**erformance measure, **E**nvironment, **A**ctuators, **S**ensors)
- Environment types
- Agent types

Agents

- An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators

- Therefor, “an agent gets percepts one at a time, and maps this percept sequence to actions (one action at a time)”

- Percept refers to the agent’s perceptual inputs at a given time instant; an agent’s perceptual sequence is the complete history of everything the agent has ever perceived.

- In general, an agent’s choice of action at any given instant can depend on the entire precept sequence observed to date, but not on anything it hasn’t perceived.

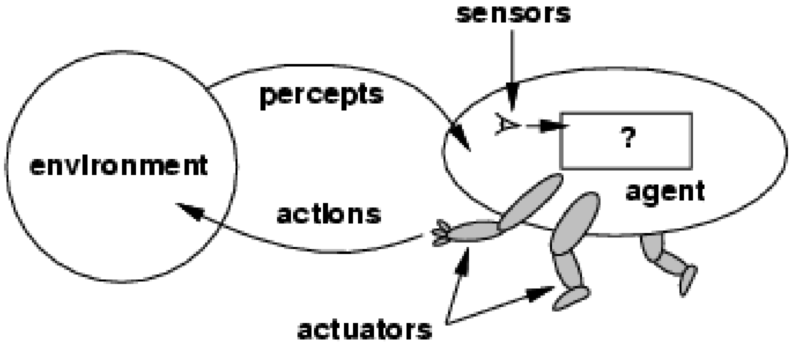
Agents

- Human agent:
  - eyes, ears, and other organs for sensors;
  - hands, legs, mouth, and other body parts for actuators

- Robotic agent:
  - cameras and infrared range finders for sensors;
  - various motors for actuators

- Software agent:
  - Keystrokes, file contents, received network packages as sensors
  - Displays on the screen, files, sent network packets as actuators

Agents and environments

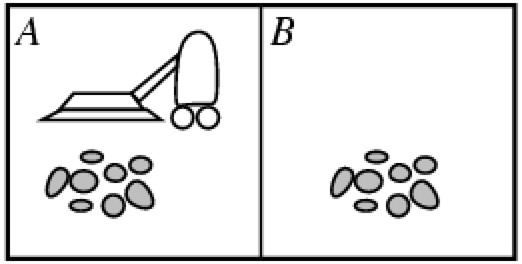


- The agent function maps from percept histories to actions:

  [*f*: P* → A]

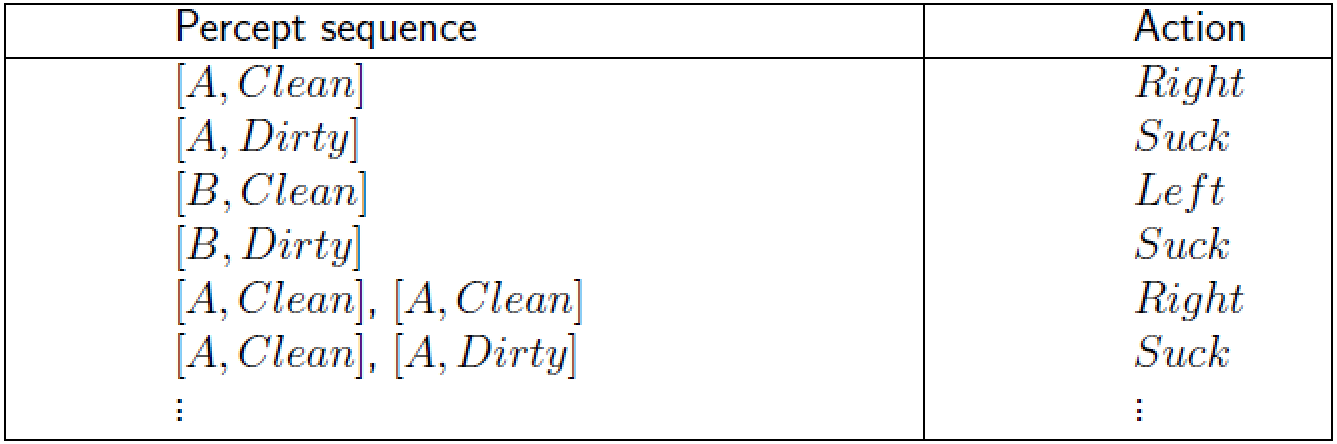
- The agent program runs on the physical architecture to produce *f*
- agent = architecture + program

Vacuum-cleaner world

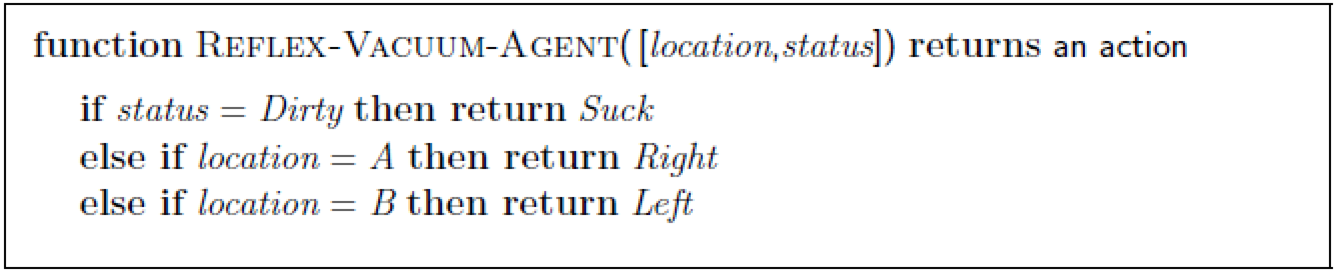


- Two locations: A and B
- Percepts: location and contents, e.g., [A,Dirty]
- Actions: *Left*, *Right*, *Suck*, *NoOp*

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?

Examples of agents in different types of applications

Agent type	Percepts	Actions	Goals	Environment
Medical diagnosis system	Symptoms, findings, patient's answers	Questions, tests, treatments	Healthy patients, minimize costs	Patient, hospital
Satellite image analysis system	Pixels of varying intensity, color	Print a categorization of scene	Correct categorization	Images from orbiting satellite
Part-picking robot	Pixels of varying intensity	Pick up parts and sort into bins	Place parts in correct bins	Conveyor belts with parts
Refinery controller	Temperature, pressure readings	Open, close valves; adjust temperature	Maximize purity, yield, safety	Refinery
Interactive English tutor	Typed words	Print exercises, suggestions, corrections	Maximize student's score on test	Set of students

Examples of agents (Cont..)

Agent Type	**Percepts**	**Actions**	**Goals**	**Environments**
Bin-Picking Robot	Images	Grasp objects; Sort into bins	Parts in correct bins	Conveyor belt
Medical Diagnosis	Patient symptoms	Tests and treatments	Healthy patients	Patient & hospital
Softbot	Web pages	ftp, mail, telnet	Collect info on a subject	Internet

Rational agents

- An agent should strive to "do the right thing", based on what it can perceive and the actions it can perform. The right action is the one that will cause the agent to be most successful.

- But what does it mean to do the right thing? We use a performance measure to evaluate any given sequence of environment states.

- **Performance measure**: An objective criterion for success of an agent's behavior e.g., performance measure of a vacuum-cleaner agent could be the amount of dirt cleaned up, amount of time taken, amount of electricity consumed, amount of noise generated, etc. A more suitable measure would reward the agent for having a clean floor

- Performance is assessed in terms of environment states and not agent states; self-assessment is often susceptible to self-delusion.

- Here is a relevant rule of thumb: "*It is advisable to design performance measures according to what one actually wants in the environment, as opposed to how one believes that agent should behave*".

Rational agents

- What is rational at any given time depends on four things
  - The performance measure that defines the criterion of success
  - The agent's prior knowledge of the environment
  - The actions that the agent can perform
  - The agent's percept sequence to date

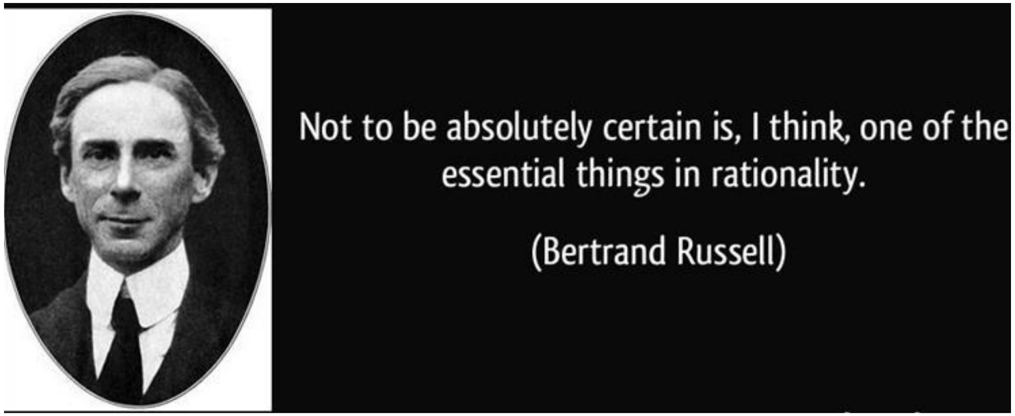
- Rational Agent: For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.

Rational agents

- Rationality is distinct from omniscience (all-knowing with infinite knowledge). "*An omniscient agent knows the actual outcome of its actions and can act accordingly. Percepts may not supply all relevant information*".

- "*rationality is not the same thing as clairvoyance (action outcomes may be unexpected) nor perfection (we maximize expected performance, not actual performance).*"

- Agents can perform actions in order to modify future percepts so as to obtain useful information (information gathering, exploration)



Not to be absolutely certain is, I think, one of the essential things in rationality.

(Bertrand Russell)

- An agent is autonomous if its behavior is determined by its own experience (with ability to learn and adapt)

PEAS

- PEAS: Performance measure, Environment, Actuators, Sensors
- Must first specify the setting for intelligent agent design

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

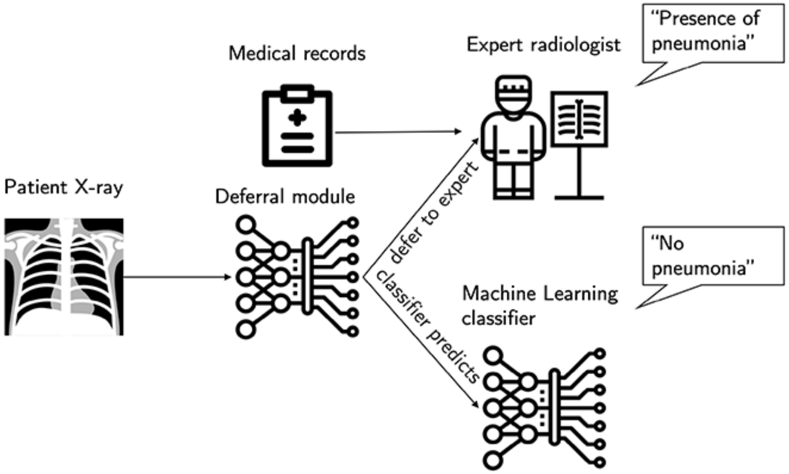
  - Performance measure?
  - Environment?
  - Actuators?
  - Sensors?

PEAS for an automated taxi driver

- **Performance measure**: Safe, fast, legal, comfortable trip, maximize profits
- **Environment**: Roads, other traffic, pedestrians, customers
- **Actuators**: Steering wheel, accelerator, brake, signal, horn
- **Sensors**: Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard
- How about a medical diagnosis system?
  - Performance measure?
  - Environment?
  - Actuators?
  - Sensors?

PEAS for a medical diagnosis system

- **Agent**: Medical diagnosis system
- **Performance measure**: Healthy patient, minimize costs, lawsuits
- **Environment**: Patient, hospital, staff
- **Actuators**: Screen display (questions, tests, diagnoses, treatments, referrals)
- **Sensors**: Keyboard (entry of symptoms, findings, patient's answers)



PEAS for a refinery controller

- **Agent**: a refinery controller
- **Performance measure**: maximize purity, yield, safety
- **Environment**: refinery, operators
- **Actuators**: valves, pumps, heaters, displays
- **Sensors**: temperature, pressure, chemical sensors

PEAS for Part-picking robot

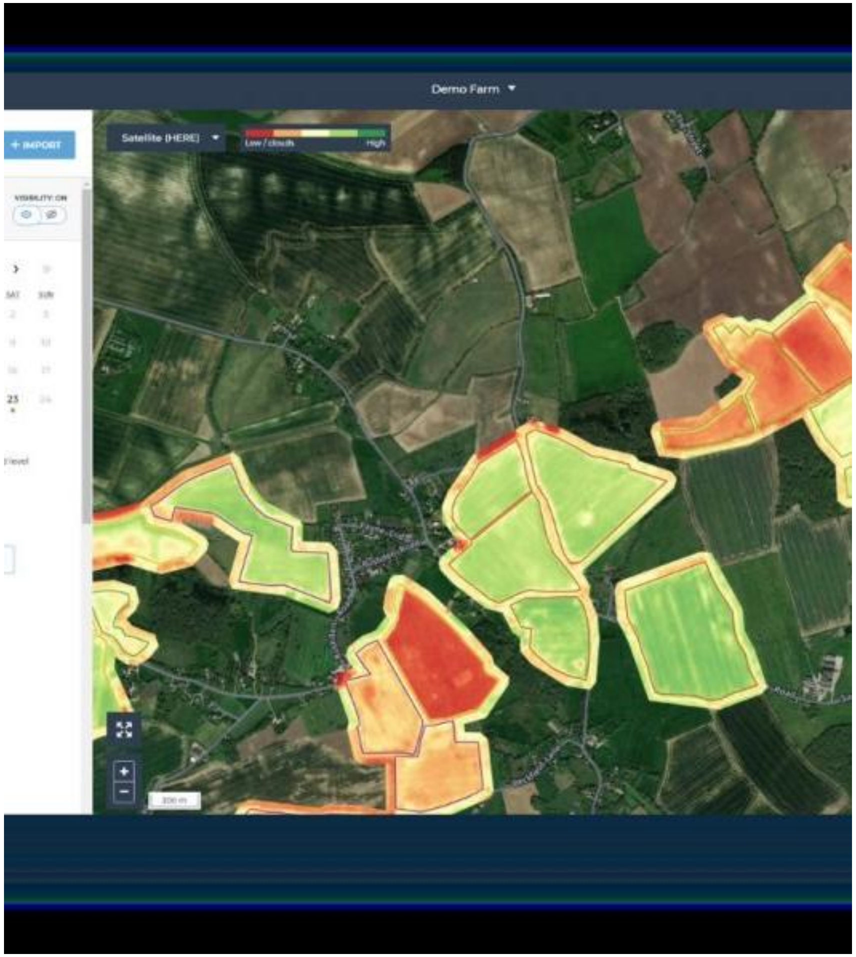
- **Agent**: Part-picking robot
- **Performance measure**: Percentage of parts in correct bins
- **Environment**: Conveyor belt with parts, bins
- **Actuators**: Jointed arm and hand
- **Sensors**: Camera, joint angle sensors

PEAS for Interactive English tutor

- **Agent**: Interactive English tutor
- **Performance measure**: Maximize student's score on test
- **Environment**: Set of students
- **Actuators**: Screen display (exercises, suggestions, corrections)
- **Sensors**: Keyboard

PEAS for a satellite image analysis system

- **Agent**: Satellite image analysis system
- **Performance measure**: correct image categorization
- **Environment**: downlink from orbiting satellite
- **Actuators**: display categorization of scene
- **Sensors**: color pixel arrays



Environment types

- Fully observable vs. partially observable
- Deterministic vs. stochastic
- Episodic vs. sequential
- Static vs. dynamic
- Discrete vs. continuous
- Single agent vs. multiagent

Environment types

- Fully observable (vs. partially observable): An agent's sensors give it access to the complete state of the environment at each point in time.

- Deterministic (vs. stochastic): The next state of the environment is completely determined by the current state and the action executed by the agent. (If the environment is deterministic except for the actions of other agents, then the environment is strategic)

- Episodic (vs. sequential): The agent's experience is divided into atomic "episodes" (each episode consists of the agent perceiving and then performing a single action), and the choice of action in each episode depends only on the episode itself.

Environment types (Cont.)

- Static (vs. dynamic): The environment is unchanged while an agent is deliberating. (The environment is semidynamic if the environment itself does not change with the passage of time but the agent's performance score does)

- Discrete (vs. continuous): A limited number of distinct, clearly defined percepts and actions.

- Single agent (vs. multiagent): An agent operating by itself in an environment.

Environment types (Cont.)

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable?			
Deterministic?			
Episodic?			
Static?			
Discrete?			
Single agent?			

Environment types (Cont.)

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable?	Yes	Yes	No
Deterministic?			
Episodic?			
Static?			
Discrete?			
Single-agent?			

Environment types (Cont.)

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable?	Yes	Yes	No
Deterministic?	Strategic	Strategic	No
Episodic?			
Static?			
Discrete?			
Single?			

Environment types (Cont.)

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable?	Yes	Yes	No
Deterministic?	Strategic	Strategic	No
Episodic?	No	No	No
Static?			
Discrete?			
Single?			

Environment types (Cont.)

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable?	Yes	Yes	No
Deterministic?	Strategic	Strategic	No
Episodic?	No	No	No
Static?	Semi	Yes	No
Discrete?			
Single?			

Environment types (Cont.)

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable?	Yes	Yes	No
Deterministic?	Strategic	Strategic	No
Episodic?	No	No	No
Static?	Semi	Yes	No
Discrete?	Yes	Yes	No
Single?			

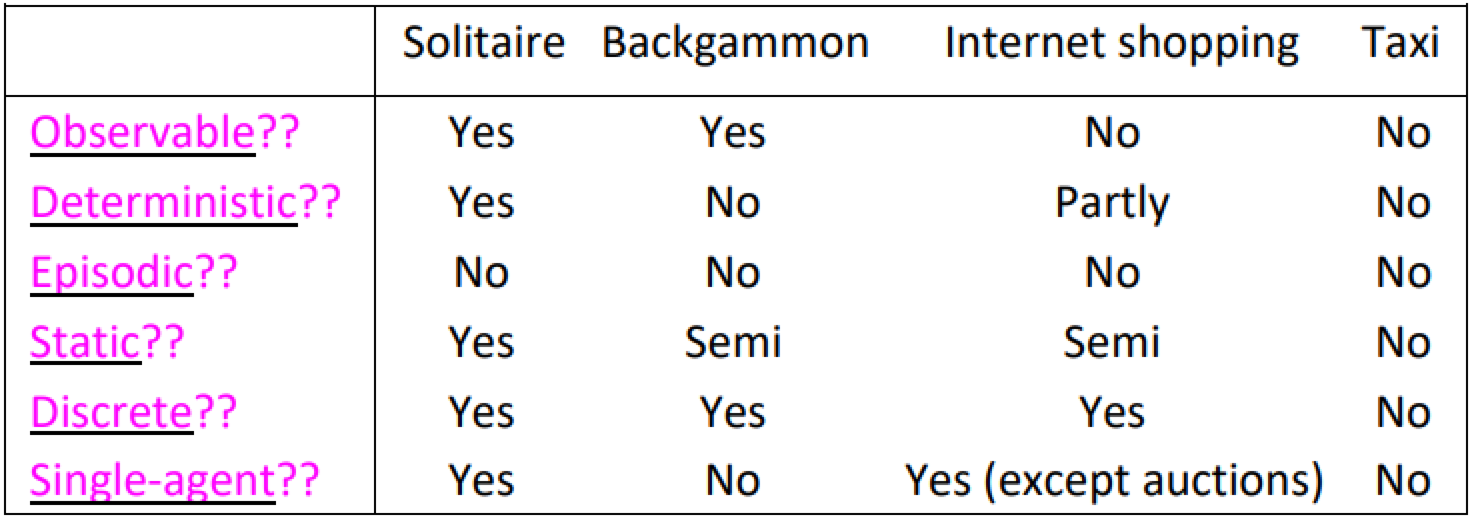
Environment types (Cont.)

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable?	Yes	Yes	No
Deterministic?	Strategic	Strategic	No
Episodic?	No	No	No
Static?	Semi	Yes	No
Discrete?	Yes	Yes	No
Single agent?	No	No	No

- The environment type largely determines the agent design

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

Environment types, another example



	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	No
Single-agent??	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

Agent functions and programs

- An agent is completely specified by the agent function mapping percept sequences to actions
- One agent function (or a small equivalence class) is rational

The goal is to find a way to implement the rational agent function concisely

Agent types

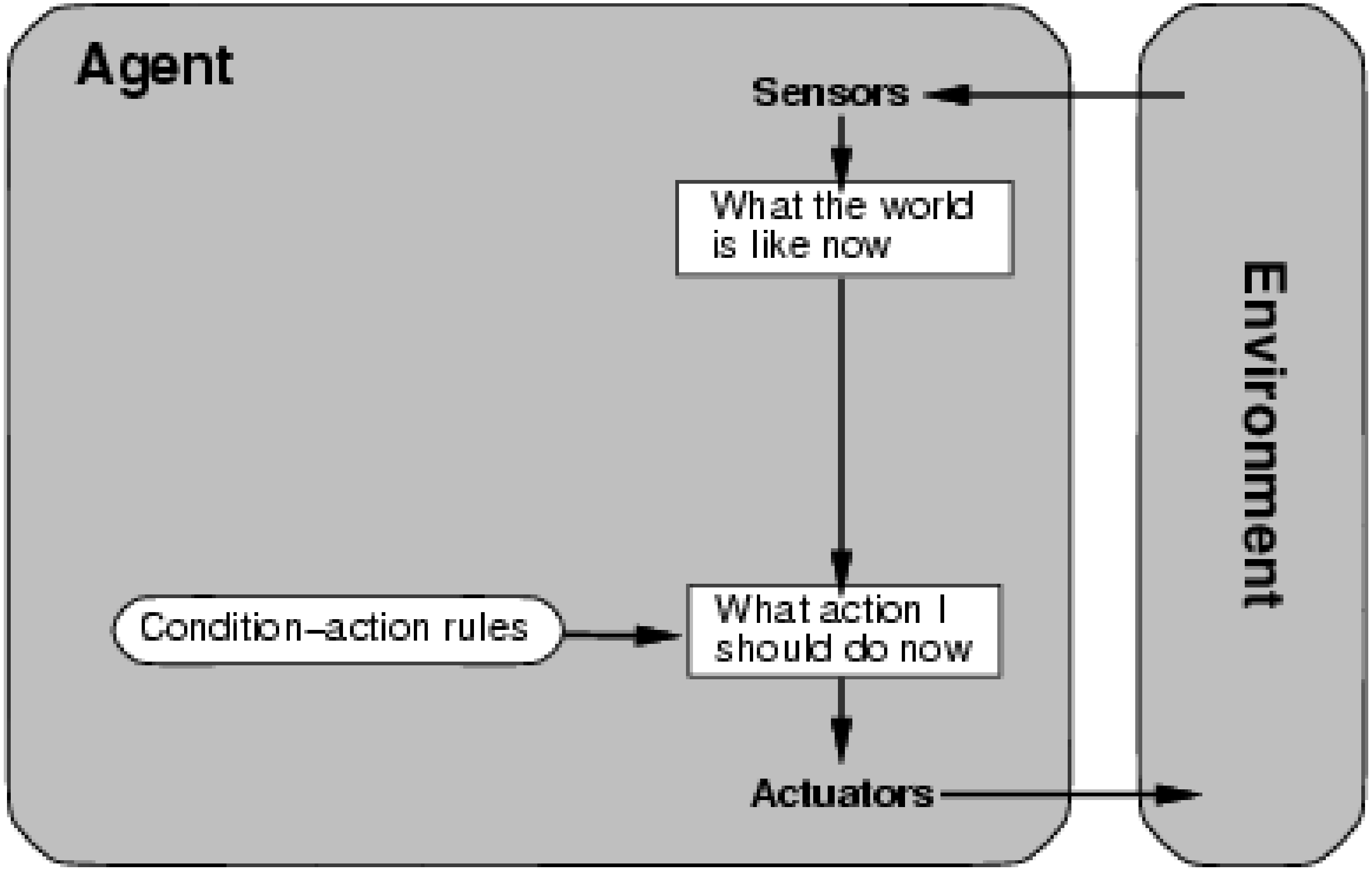
- Four basic agent types in order of increasing generality:

More sophisticated

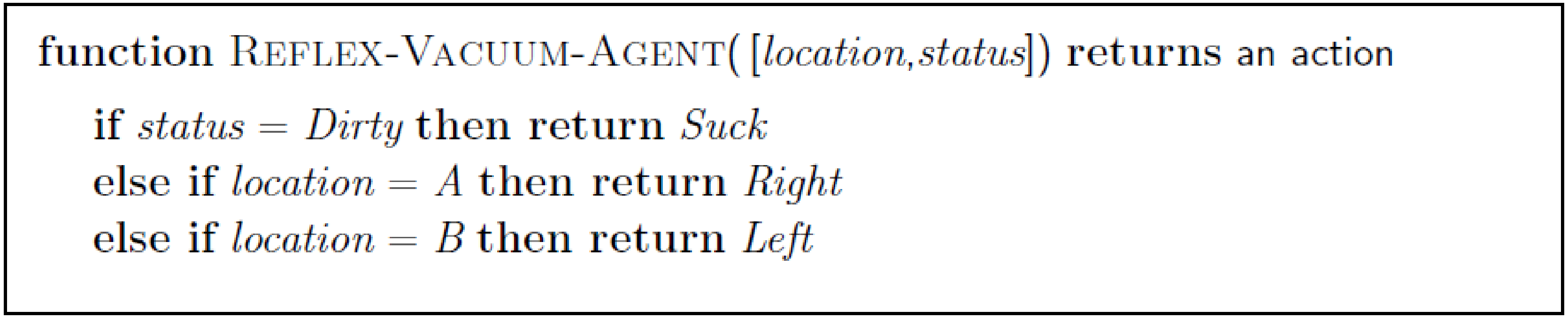
  - ***Table-driven agent***
  - ***Simple reflex agents***
  - ***Model-based reflex agents***
  - ***Goal-based agents***
  - ***Utility-based agents***

- All these can be turned into learning agents

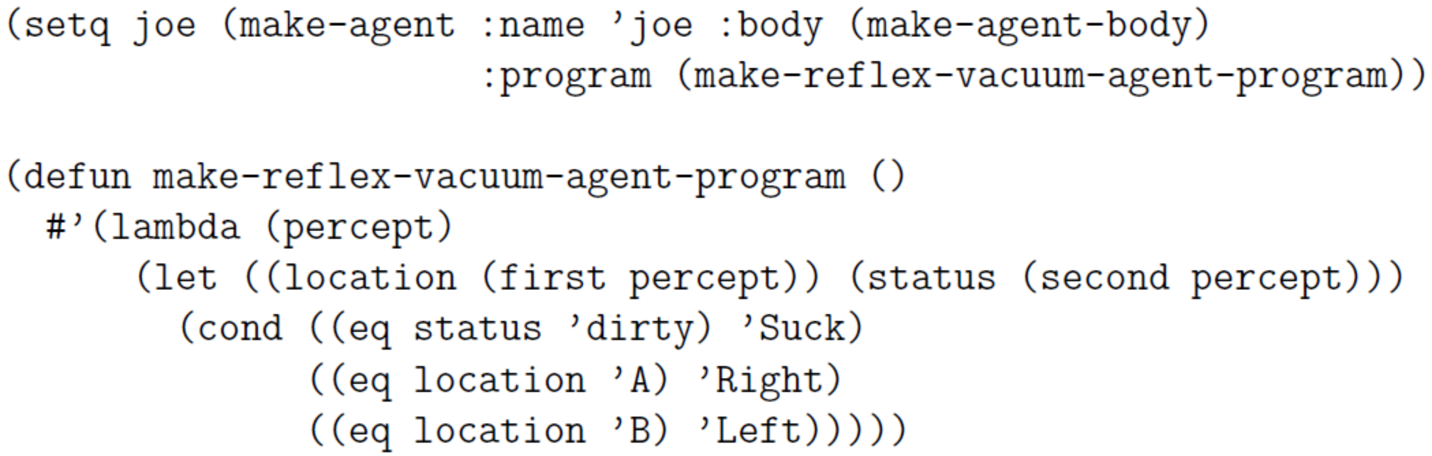
Simple reflex agents



Simple reflex agents: Vacuum cleaner 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
```

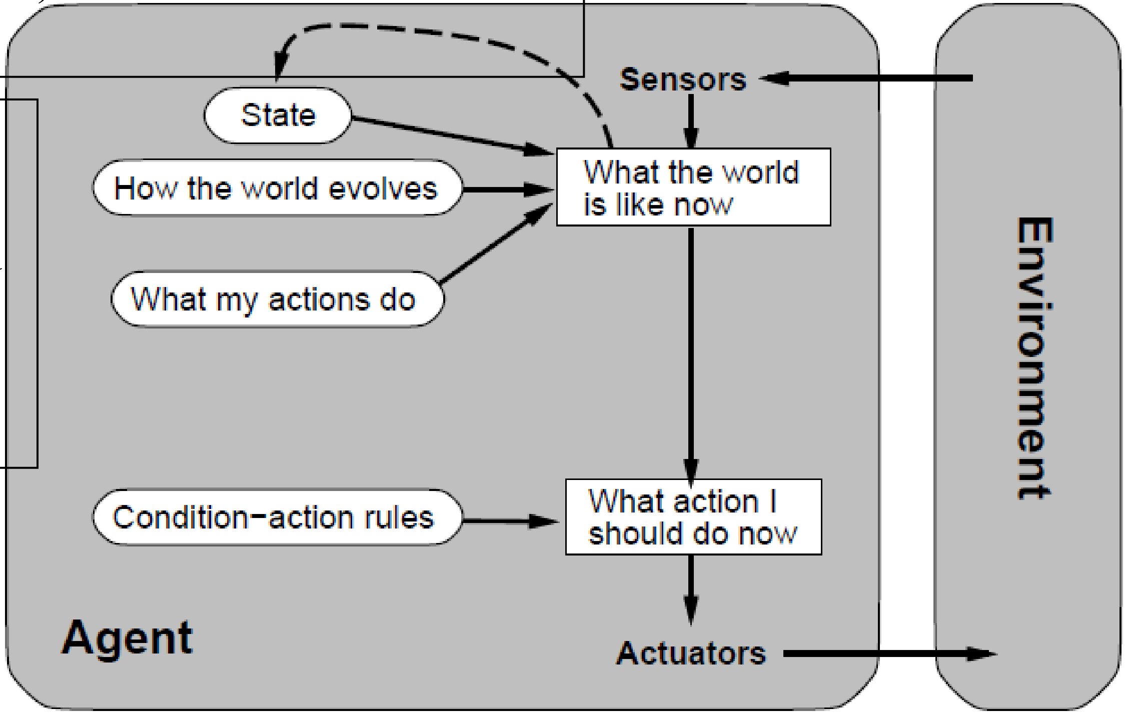


```
(setq joe (make-agent :name 'joe :body (make-agent-body)
                      :program (make-reflex-vacuum-agent-program))

(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)))))
```

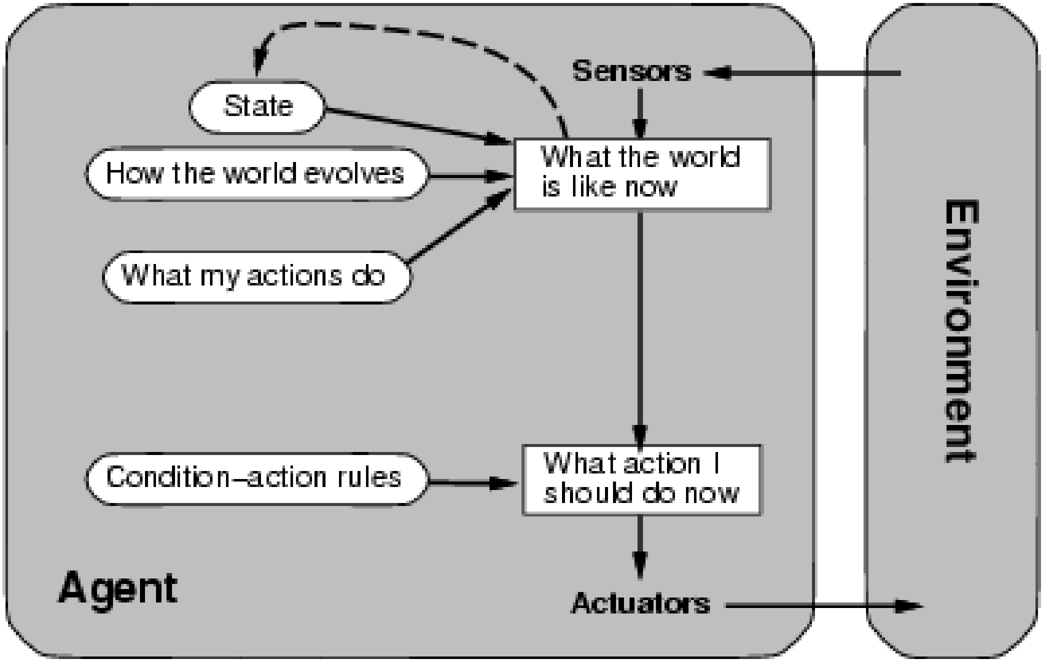
Simple reflex agents with state

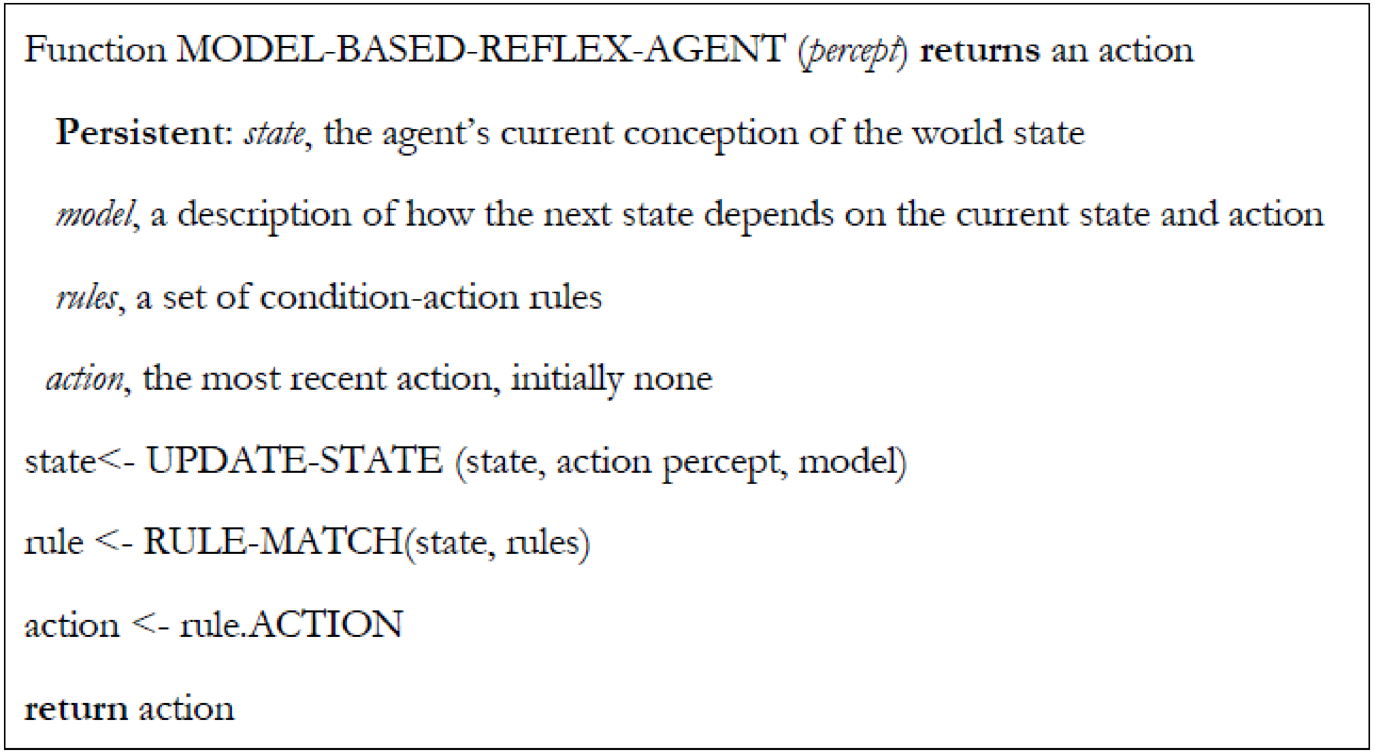
```
function REFLEX-AGENT-WITH-STATE (percept) returns action
    static: state, a description of the current world state
            rules, a set of condition-action rules

    state ← UPDATE-STATE (state, percept)
    rule ← RULE-MATCH (state, rules)
    action ← RULE-ACTION [rule]
    state ← UPDATE-STATE (state, action)
    return action
```

A reflex agent with internal state works by finding a rule whose condition matches the current situation (as defined by the percept and the stored internal state) and then doing the action associated with that rule.



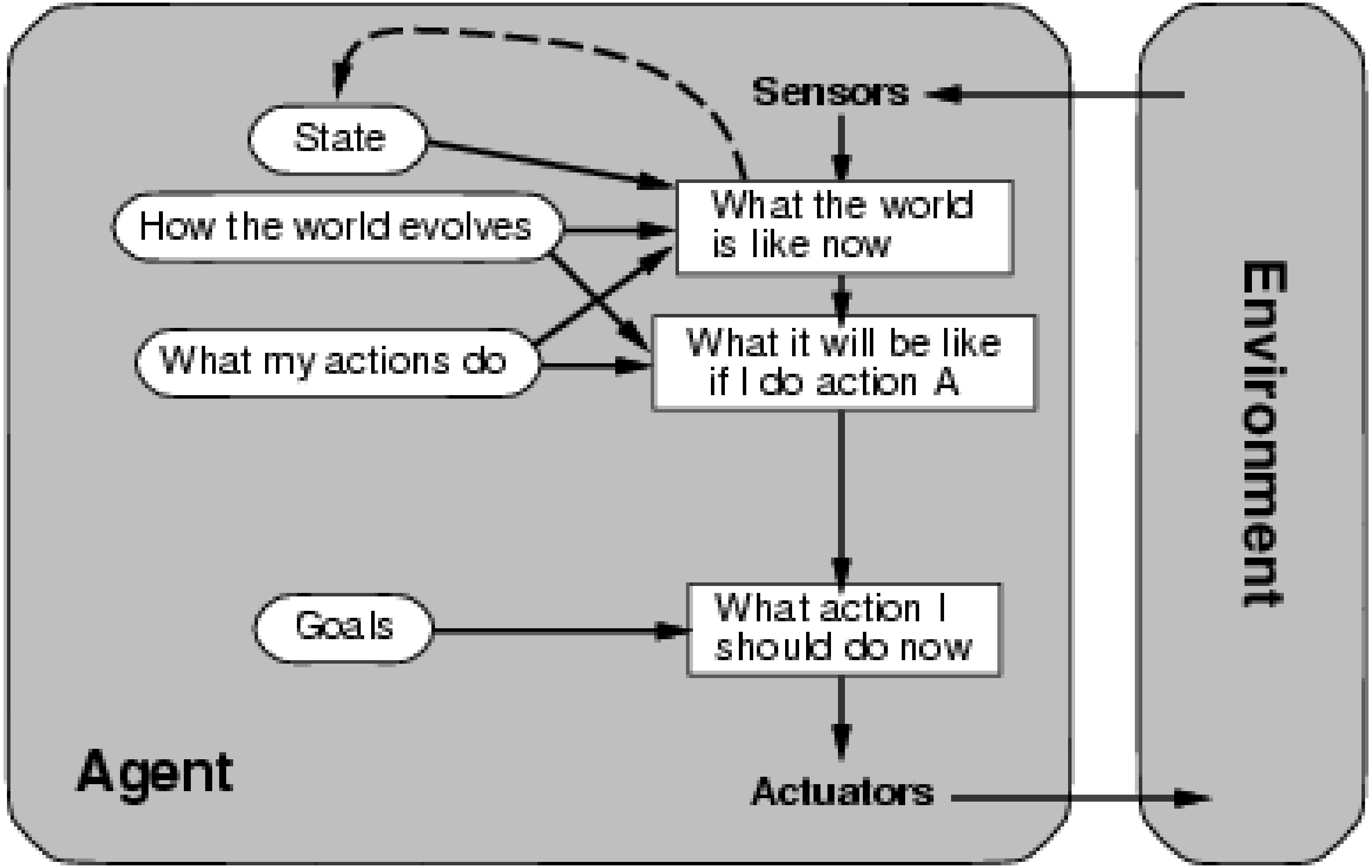
Model-based reflex agents



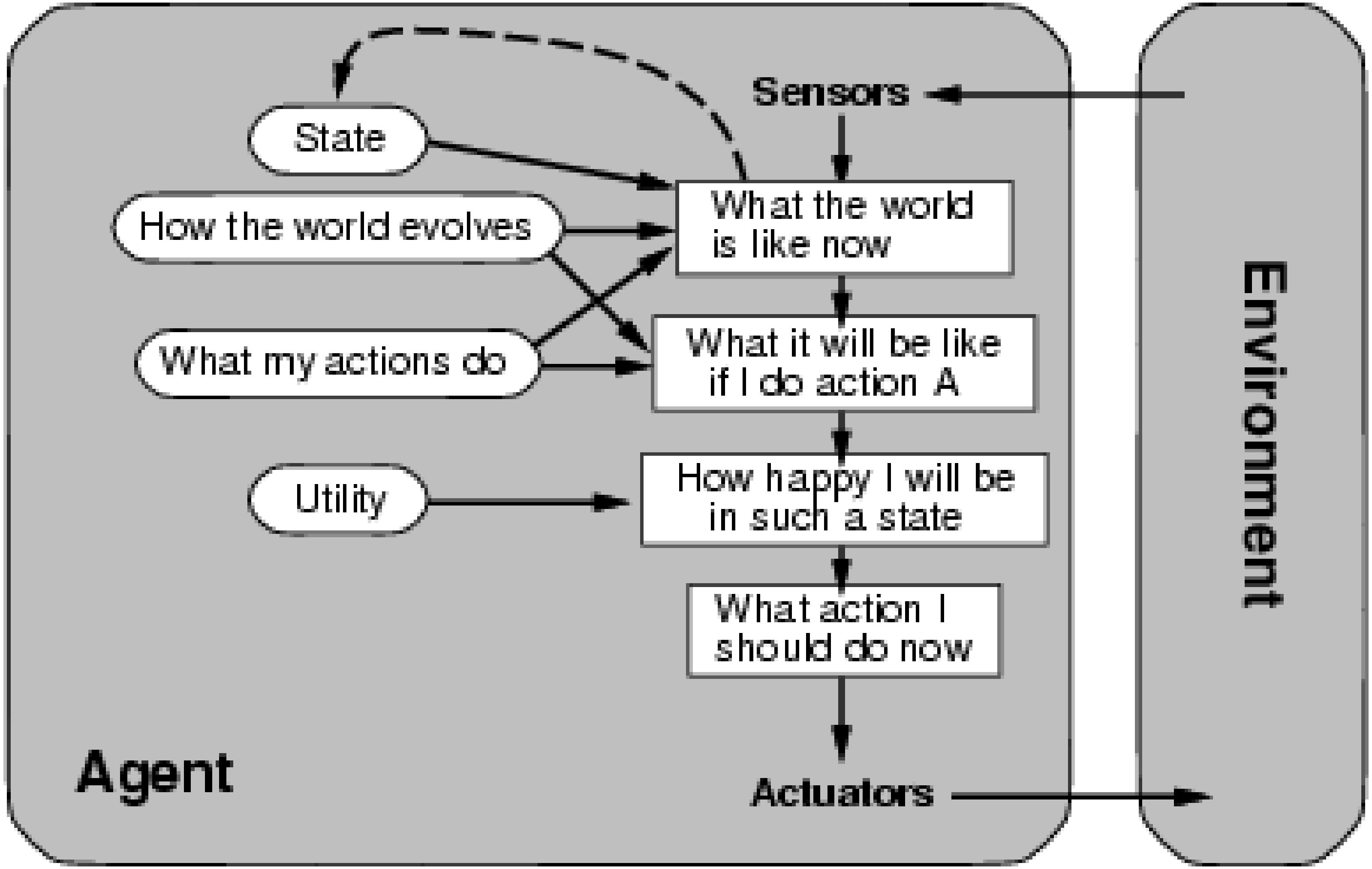


```
Function MODEL-BASED-REFLEX-AGENT (percept) returns an action
  Persistent: state, the agent's current conception of the world state
  model, a description of how the next state depends on the current state and action
  rules, a set of condition-action rules
 action, the most recent action, initially none
state<- UPDATE-STATE (state, action percept, model)
rule <- RULE-MATCH(state, rules)
action <- rule.ACTION
return action
```

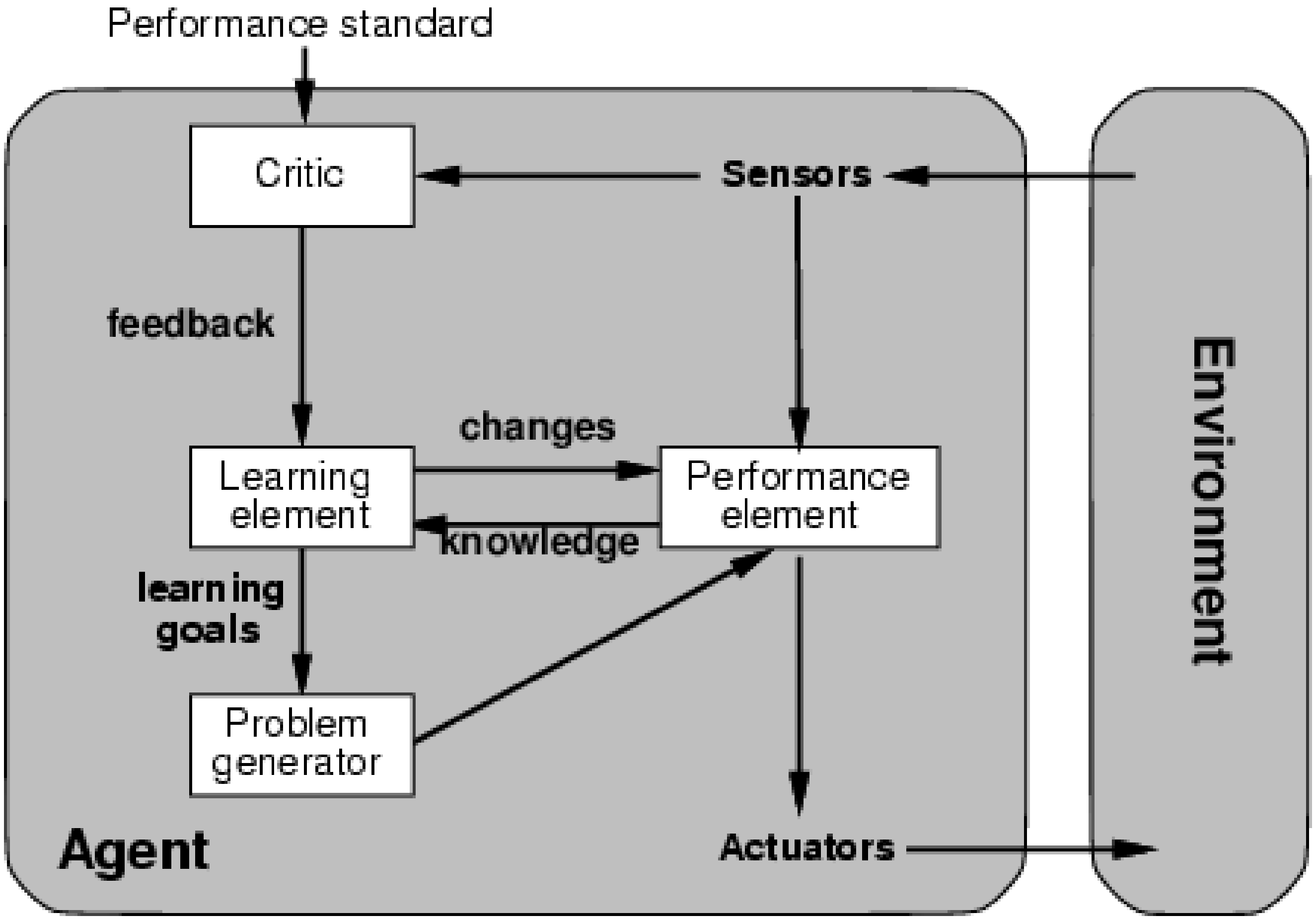
Goal-based agents


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

Utility-based agents


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

UCCS University of Colorado Colorado Springs

University of Colorado

Boulder | Colorado Springs | Denver | Anschutz Medical Campus