


# Chapter 2

# Intelligent Agents



Jane Hsu  
National Taiwan University

Acknowledgements: This presentation is created by Jane hsu based on the lecture slides from *The Artificial Intelligence: A Modern Approach* by Russell & Norvig, a PowerPoint version by Min-Yen Kan, as well as various materials from the web.

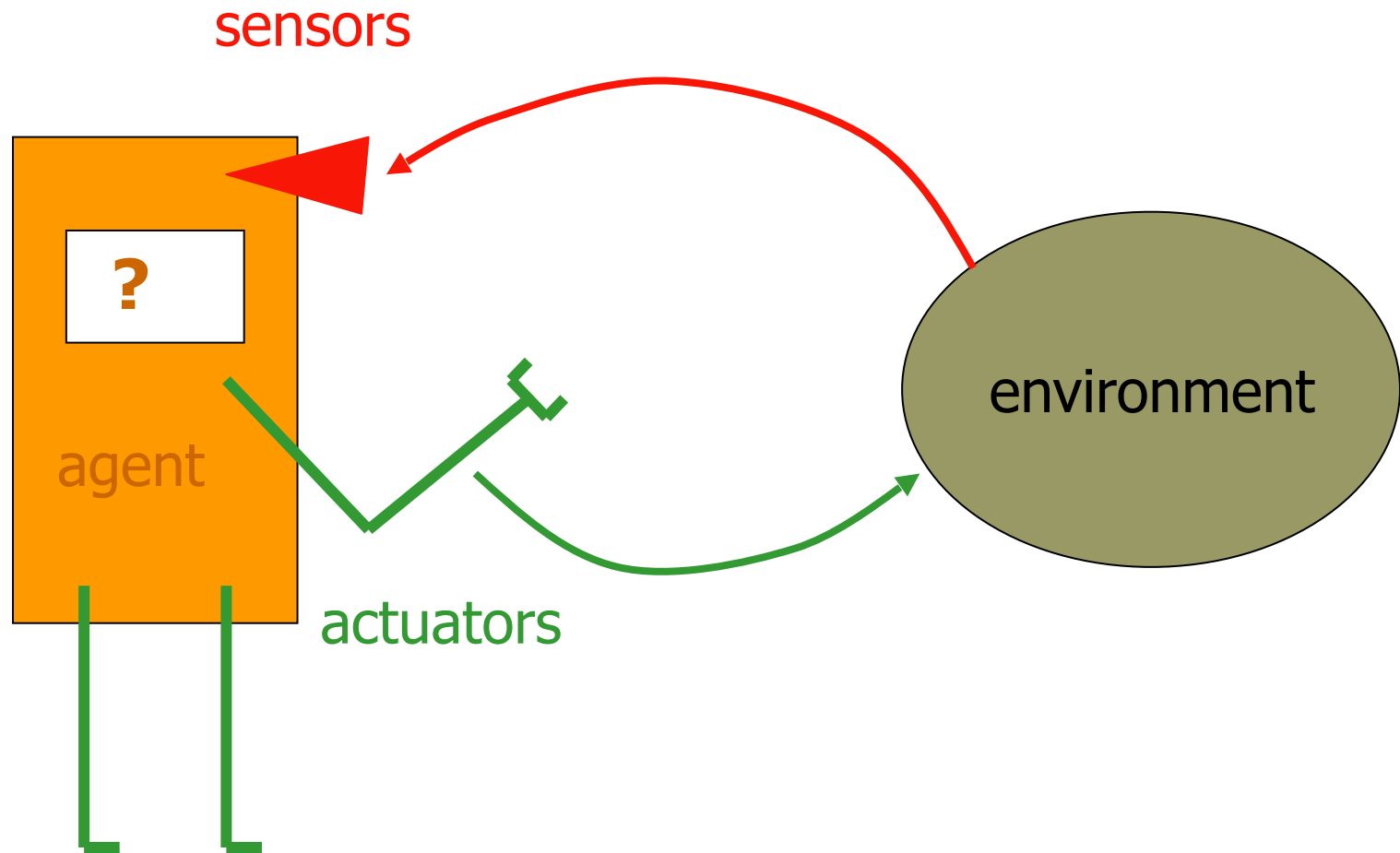
# Outline

---

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

# Agent

---



# Agents

---

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

- Human agent

- Sensors: eyes, ears, and other organs
- Actuators: hands, legs, mouth, and other body parts

- Robotic agent

- Sensors: cameras and infrared range finders
- Actuators: various motors, and robotic arms

- Software agent

- Sensors: ??
- Actuators: ??

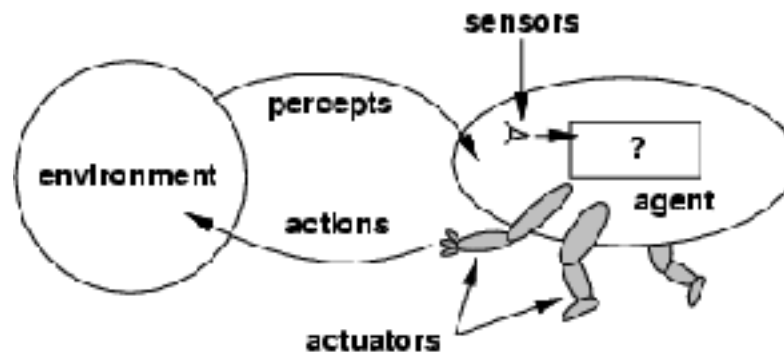
# Agency According to Marvin Minsky

---

- “... where many simpler things come together and their combined actions enable expressive and knowledgeable behavior. ... I see an *assistant* as *an intelligent machine composed of many agencies of agents*, not an agent being an intelligent machine.”

*--- The Society of Mind*

# Agents and Environments



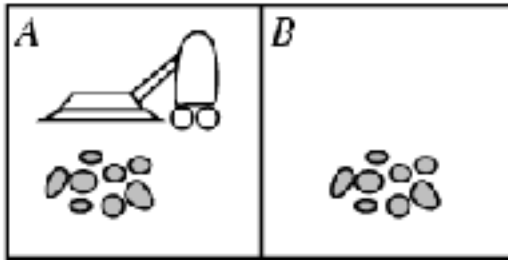
- The **agent function** maps from percept histories to actions:

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

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

# Vacuum-Cleaner World

---

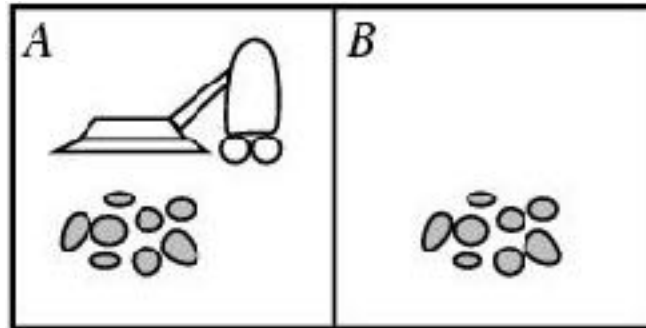


- Percepts: location and contents, e.g.,  $[A, \textit{Dirty}]$
- Actions: *Left*, *Right*, *Suck*, *NoOp*



# The Vacuum-Cleaner Agent

---

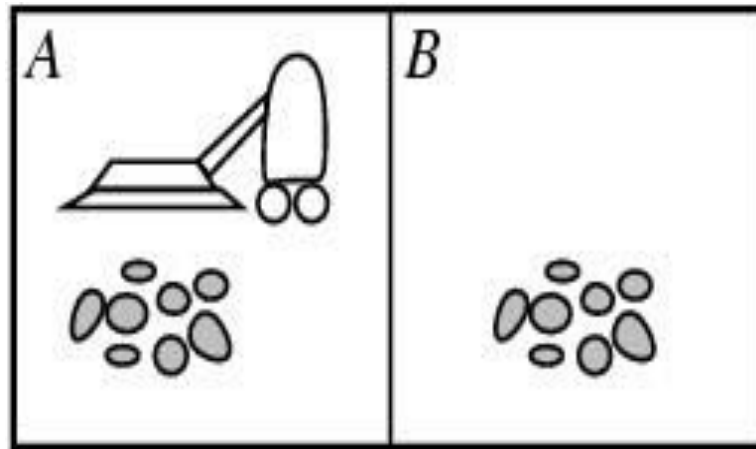


| Percept sequence      | Action |
|-----------------------|--------|
| [A,Clean]             | Right  |
| [A, Dirty]            | Suck   |
| [B, Clean]            | Left   |
| [B, Dirty]            | Suck   |
| [A, Clean],[A, Clean] | Right  |
| [A, Clean],[A, Dirty] | Suck   |
| ...               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



# A Reflex Vacuum-Cleaner Agent

---



```
function REFLEX-VACUUM-AGENT ([location, status]) return 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?

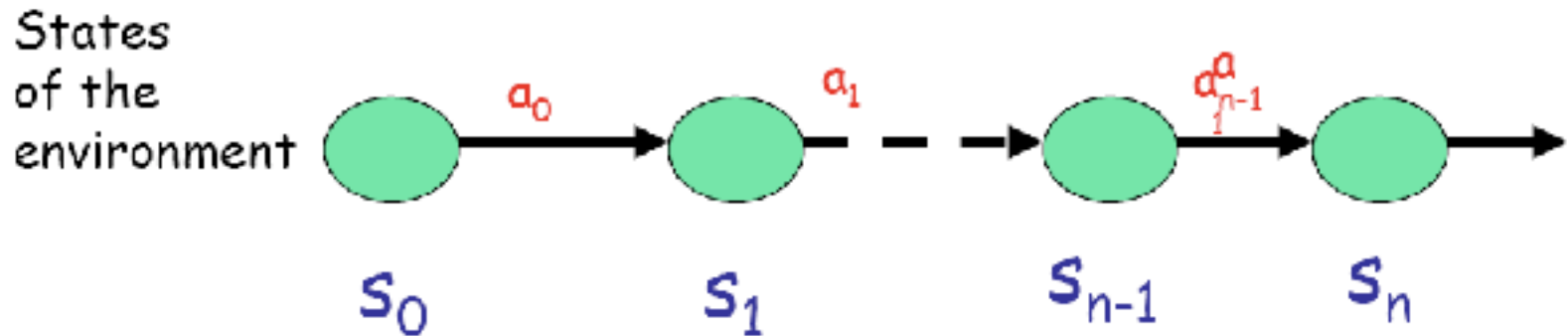
# Rational Agents

---

- 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
- **Performance measure**: An objective criterion for success of an agent's behavior, e.g.
  - amount of dirt cleaned up
  - amount of time taken
    - one point per square cleaned up (in time T)
    - one point per clean square per time step, minus one per move
    - penalty for  $> k$  dirty squares
  - amount of electricity consumed
  - amount of noise generated, etc.

# Specifying Performance Measures

- Performance measures are external



$[s_0, s_1, \dots, s_n]$  is the environment state history generated by the agent.

- A function from state history to real value  
 $V: S^* \rightarrow R$

# Rational Agents

---

- 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.

# Rationality

---

- Rationality is distinct from *omniscience* (all-knowing with infinite knowledge)
  - Percepts may not supply all relevant info
- Rationality is distinct from *clairvoyant*
  - Action outcomes may not be as expected
- Rationality is **exploration, learning, autonomy**
  - Agents can perform actions to obtain useful information
  - Agents can learn and adapt
  - Agent actions are determined by its own experience

# Task Environment

---

- PEAS: **P**erformance measure, **E**nvironment, **A**ctuators, **S**ensors
- Must first specify the setting for intelligent agent design
- Consider, e.g., the task of designing an automated taxi driver:
  - **P**erformance measure
  - **E**nvironment
  - **A**ctuators
  - **S**ensors

# PEAS: Automated Taxi Driver

---

- Must first specify the setting for intelligent agent design
  - **P**erformance measure: Safe, fast, legal, comfortable trip, maximize profits
  - **E**nvironment: Roads, other traffic, pedestrians, customers
  - **A**ctuators: Steering wheel, accelerator, brake, signal, horn
  - **S**ensors: Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard

# PEAS: Medical Diagnosis System

---

- **P**erformance measure: Healthy patient, minimize costs, lawsuits
- **E**nvironment: Patient, hospital, staff
- **A**ctuators: Screen display (questions, tests, diagnoses, treatments, referrals)
- **S**ensors: Keyboard (entry of symptoms, findings, patient's answers)



# PEAS: Part-Picking Robot

---

- **P**erformance measure: Percentage of parts in correct bins
- **E**nvironment: Conveyor belt with parts, bins
- **A**ctuators: Jointed arm and hand
- **S**ensors: Camera, joint angle sensors

# PEAS: Interactive English Tutor

---

- **P**erformance measure: Maximize student's score on test
- **E**nvironment: Set of students
- **A**ctuators: Screen display (exercises, suggestions, corrections)
- **S**ensors: Keyboard

# 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**)

# Environment Types

---

- **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. e.g.
  - to spot defective parts on an assembly line
  - How about chess, taxi driving?

# Environment types

---

- **Static** (vs. dynamic): The environment is unchanged while an agent is deliberating. (The environment is **semi-dynamic** 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. multi-agent): An agent operating by itself in an environment.

# Environment Types

---

|                  | Chess with<br>a clock | Chess without<br>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

# 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
- Aim: find a way to implement the rational agent function concisely
  - Table-lookup
  - Condition-action rules

# Table-Lookup Agent

---

## □ Drawbacks:

- Huge table
- Take a long time to build the table
- No autonomy
- Even with learning, need a long time to learn the table entries

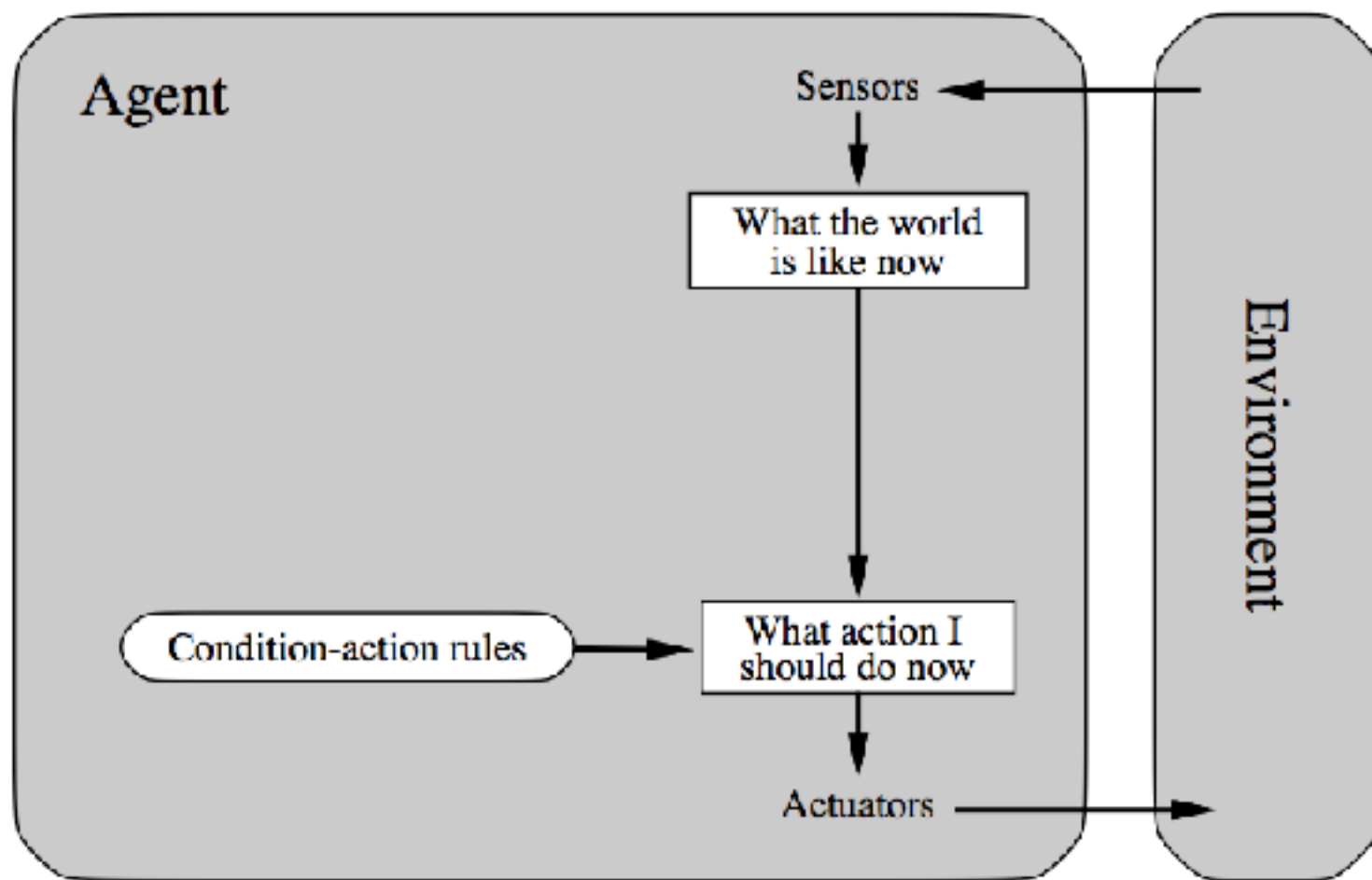


# Agent Types

---

- Four basic types in order of increasing generality:
  - Simple reflex agents
  - Model-based reflex agents
  - Goal-based agents
  - Utility-based agents

# Simple Reflex Agents



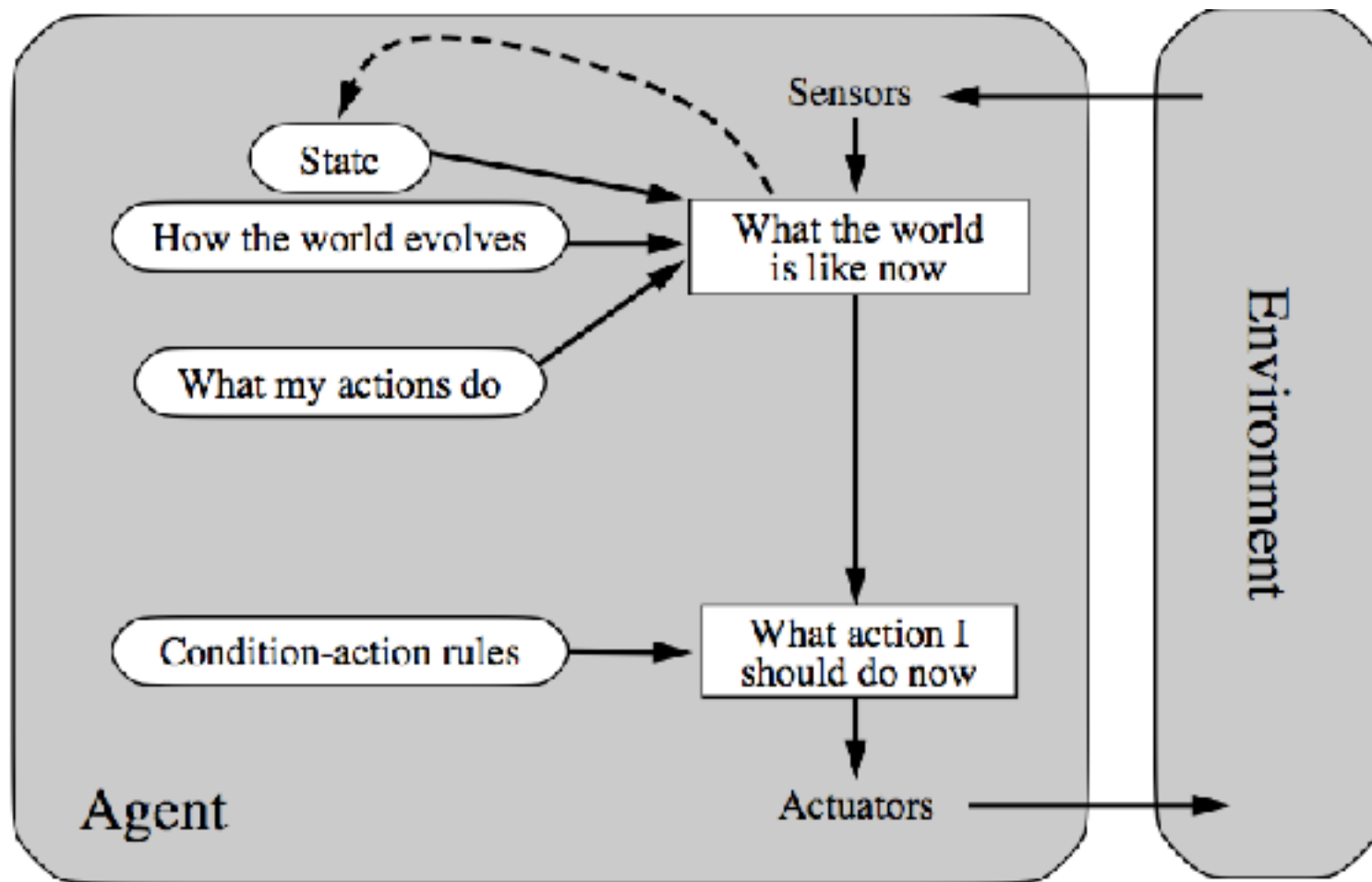
# Simple Reflex Agents

---

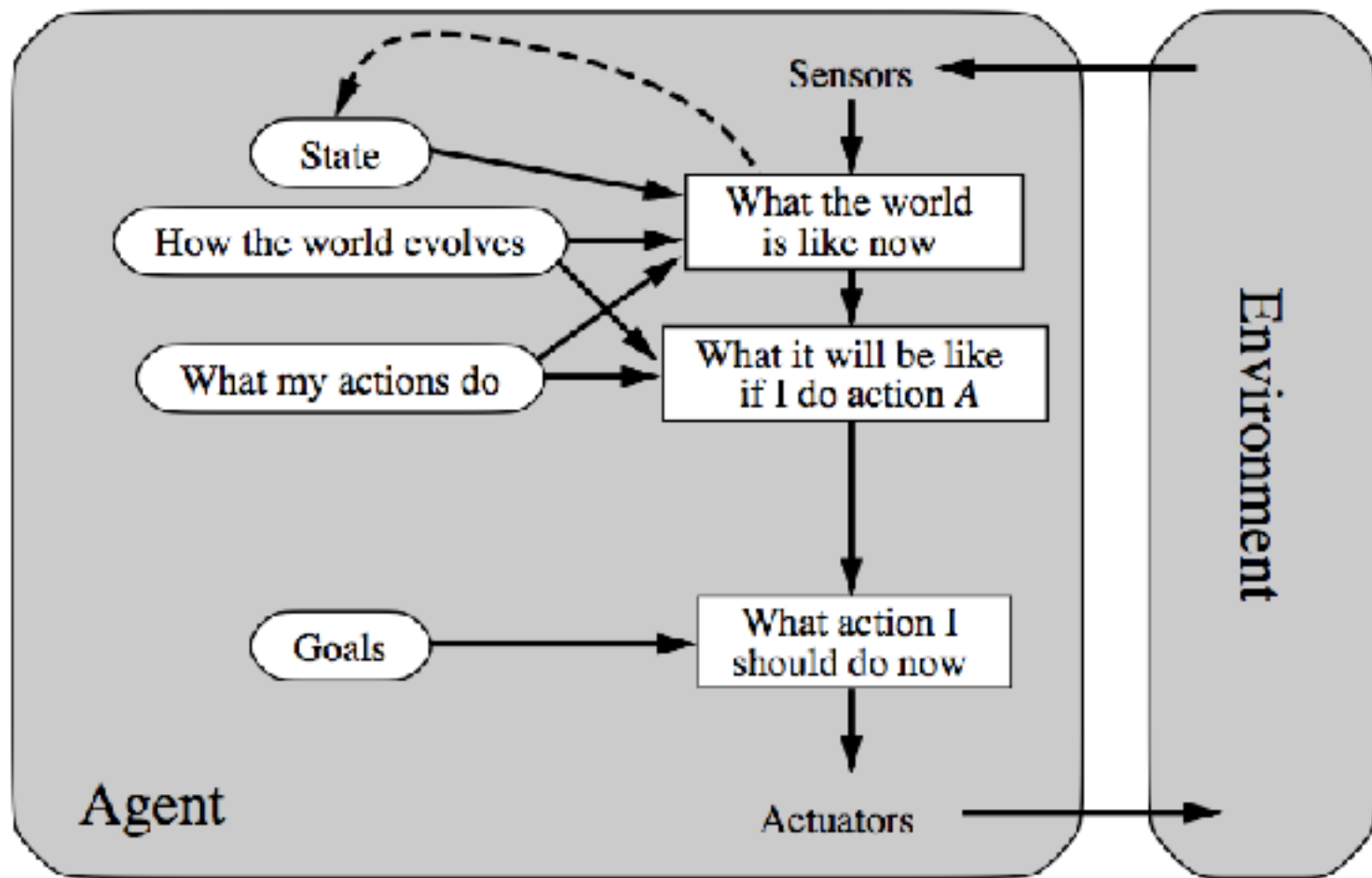
```
function REFLEX-VACUUM-AGENT([location,status]) returns an action
  if status = Dirty then return Suck
  else if location = A then return Right
  else if location = B then return Left
```

```
(setq joe (make-agent :body (make-agent-body)
  :program
    #'(lambda (percept)
      (destructuring-bind (location status) percept
        (cond ((eq status 'Dirty) 'Suck)
              ((eq location 'A) 'Right)
              ((eq location 'B) 'Left))))))
```

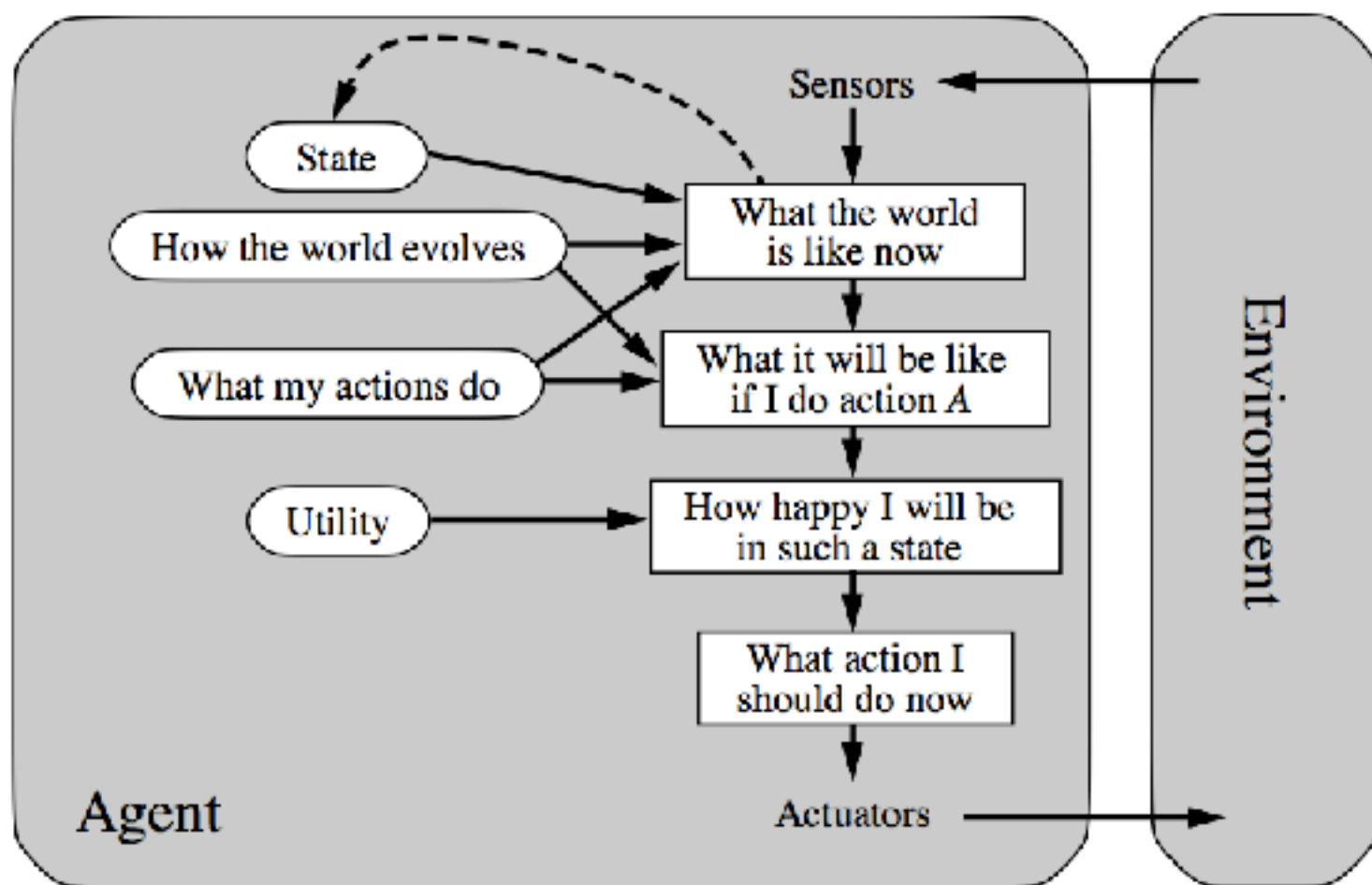
# Model-Based Reflex Agents



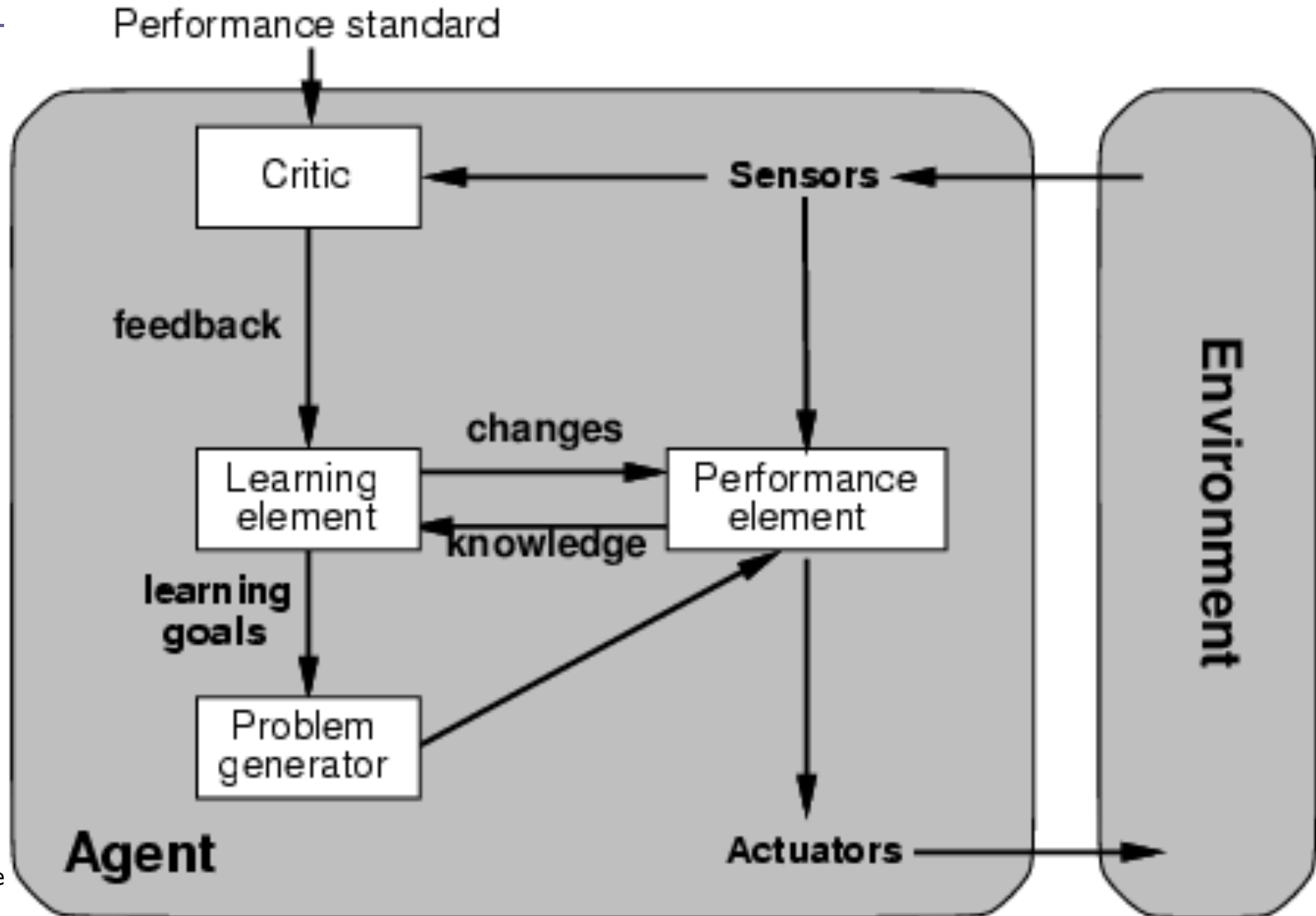
# Goal-Based Agents



# Utility-Based Agents

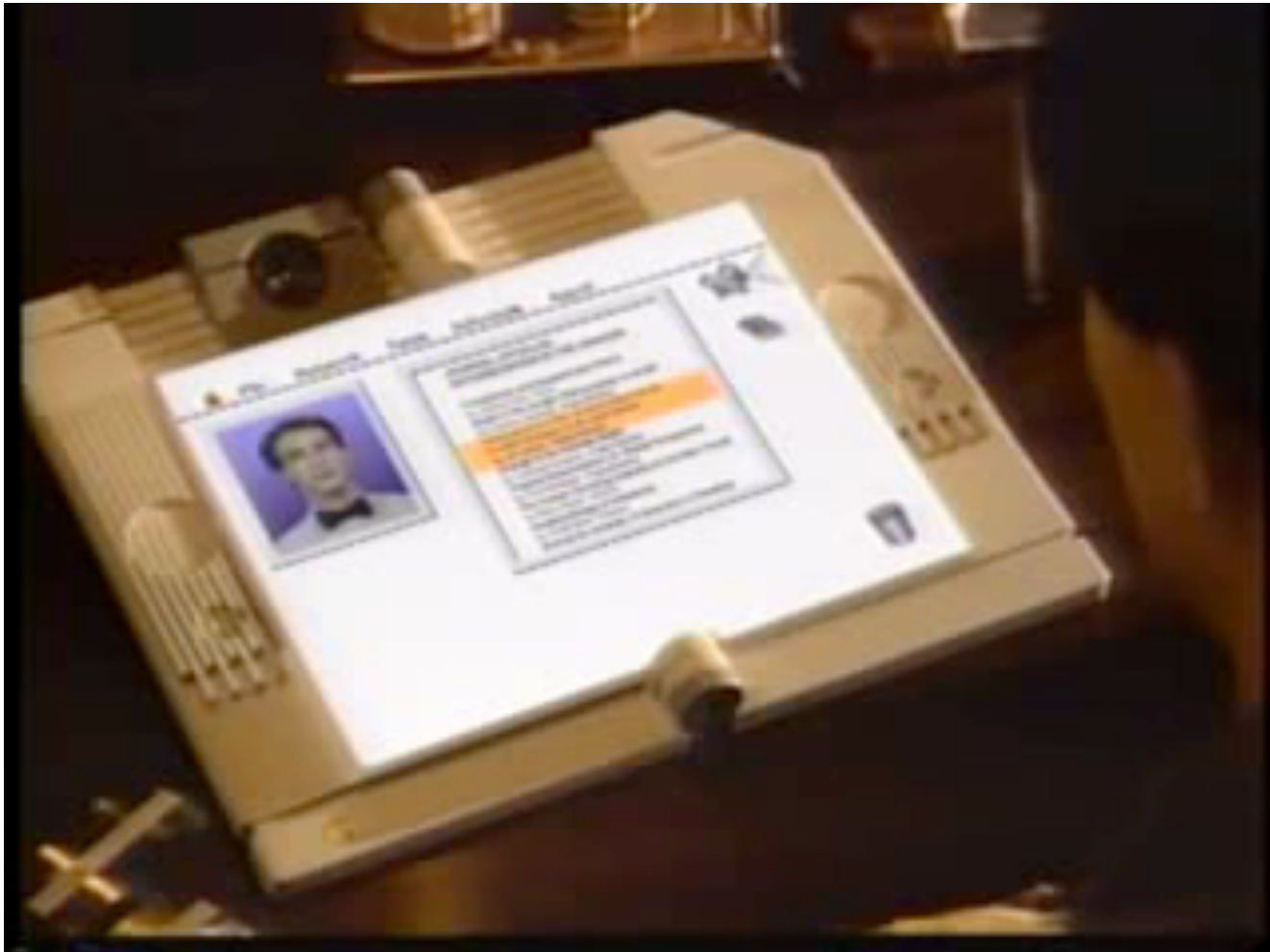


# Learning Agents



# Knowledge Navigator (Apple, 1987)

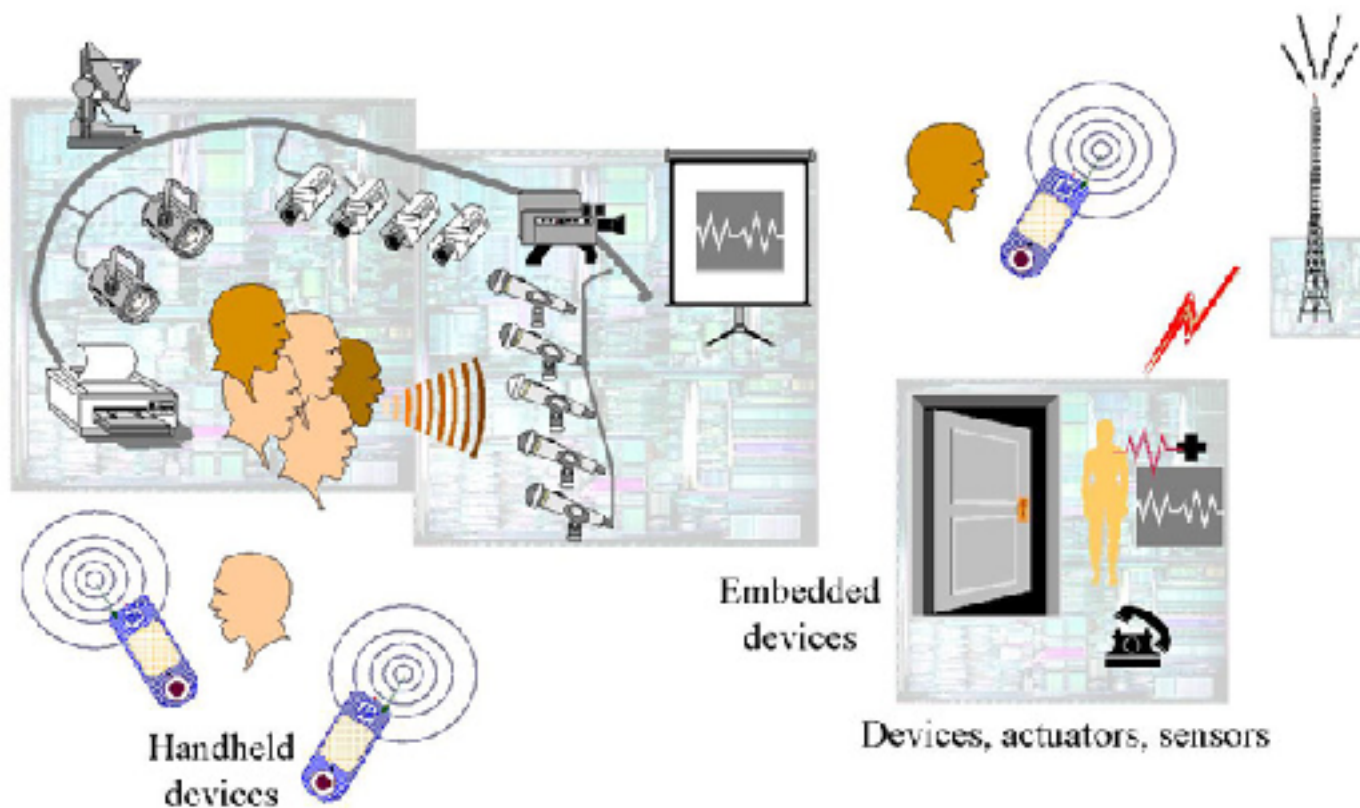
---



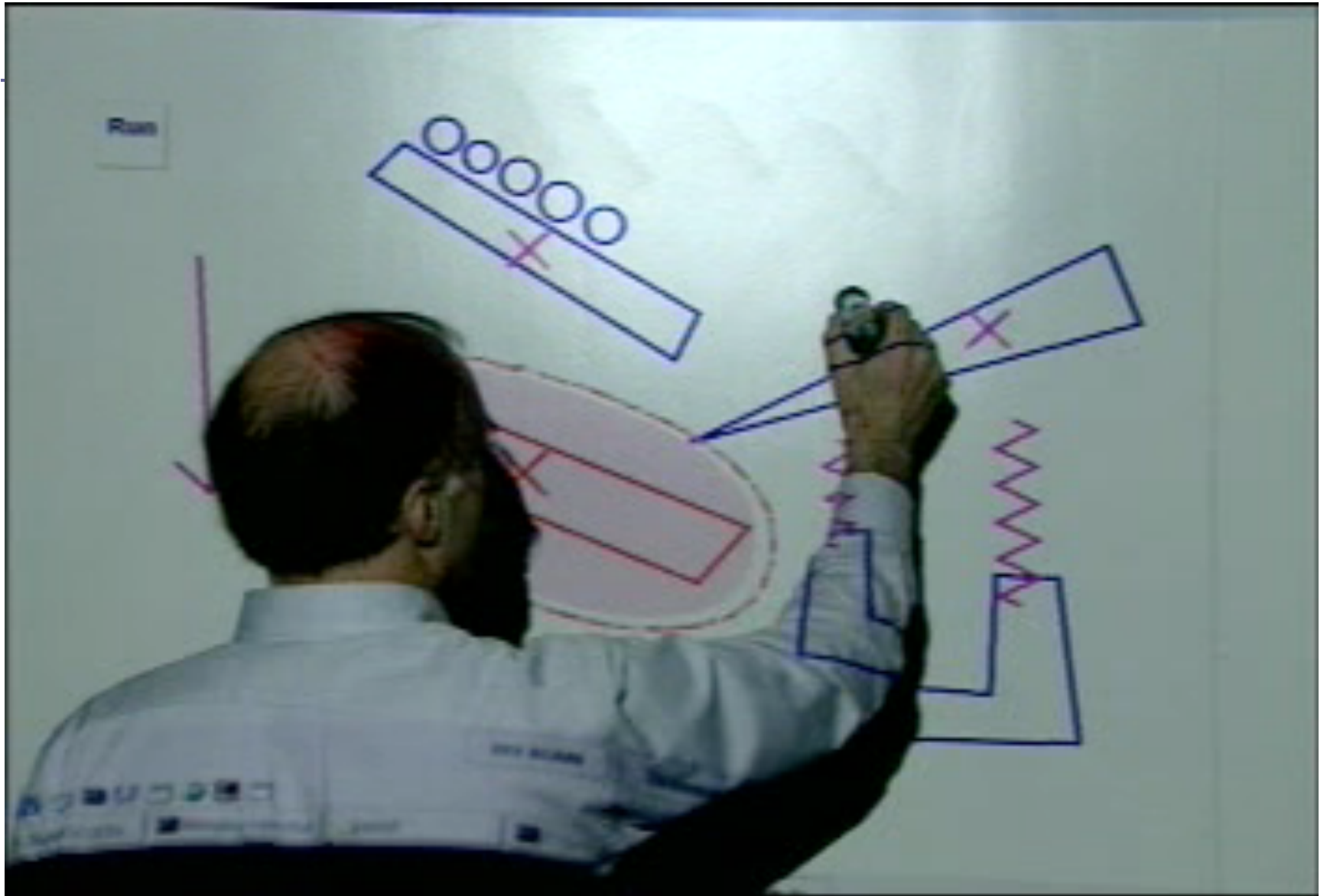


# Ubiquitous AI

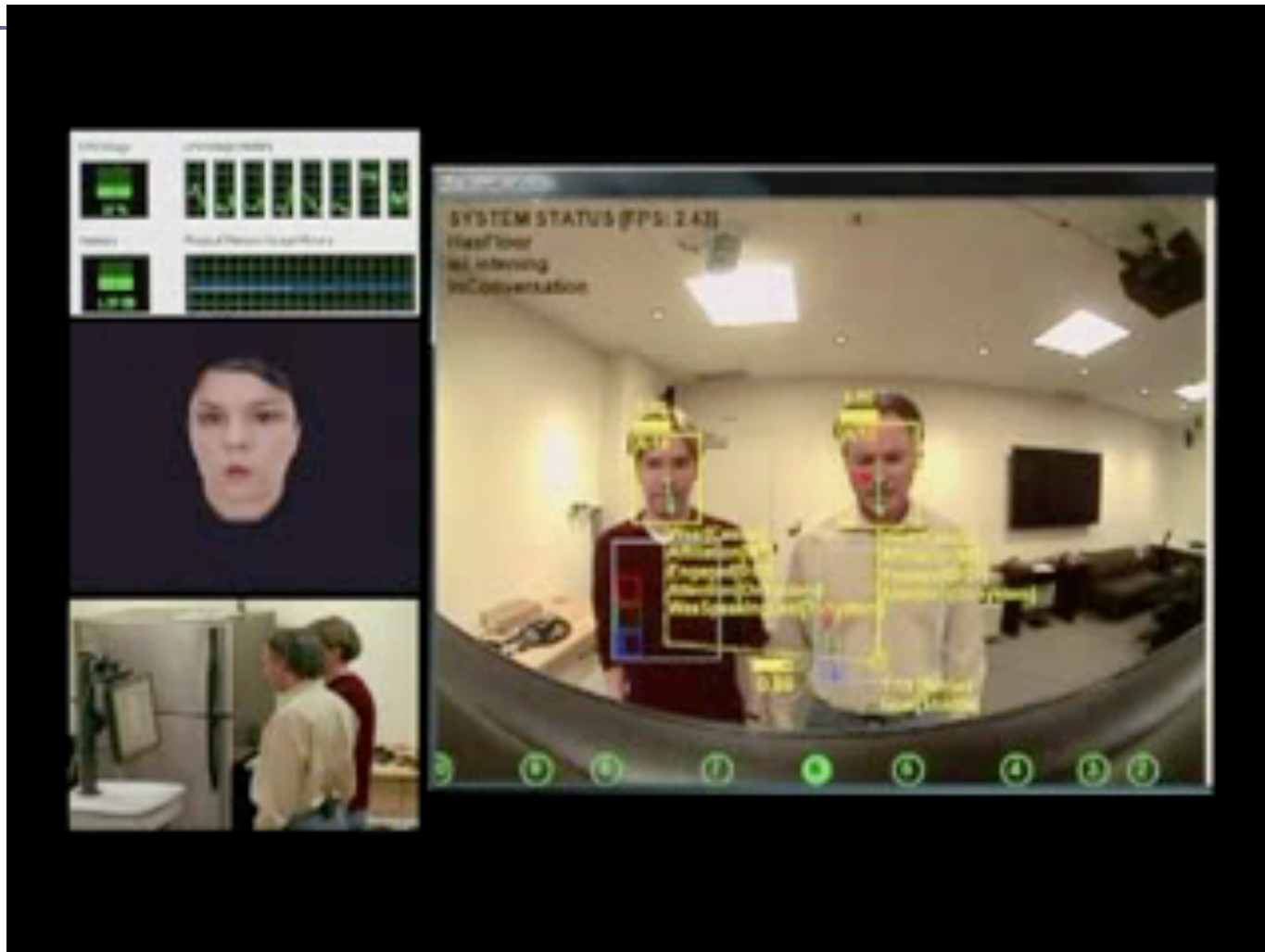
## □ Project Oxygen (MIT CSAIL)



# Intelligent Room [2002]



# Situated Interaction (Microsoft Research)



# IBM Watson Plays Jeopardy!

