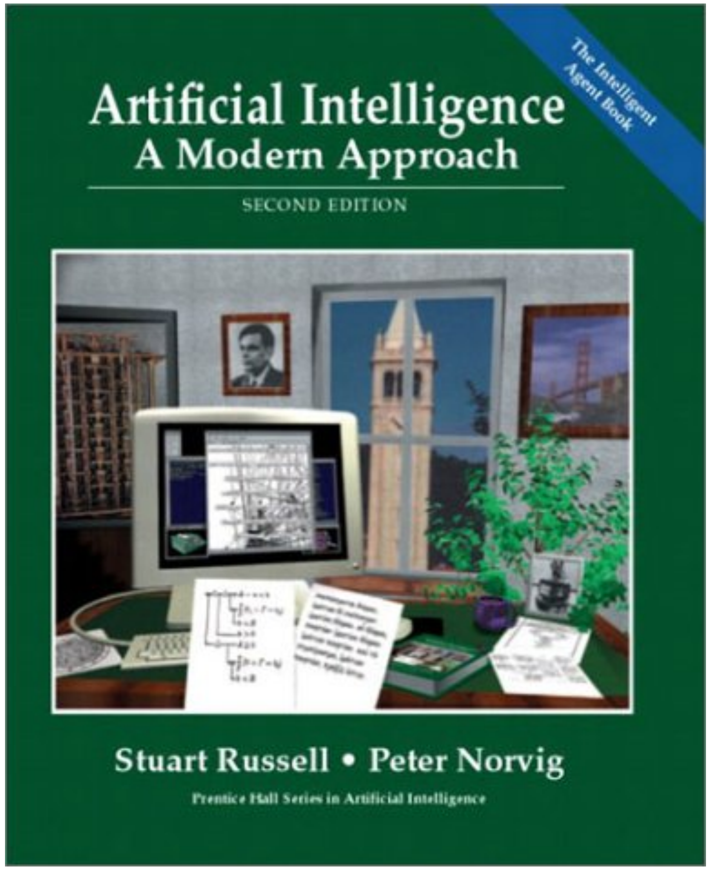
Intelligent Autonomous Agents

Ralf Möller, Rainer Marrone
Hamburg University of Technology

Lab class

- Tutor: Rainer Marrone
- Time: Monday 12:15-13:00
- Locaton: SBS93 A0.13.1/2
  - Starting in Week 3

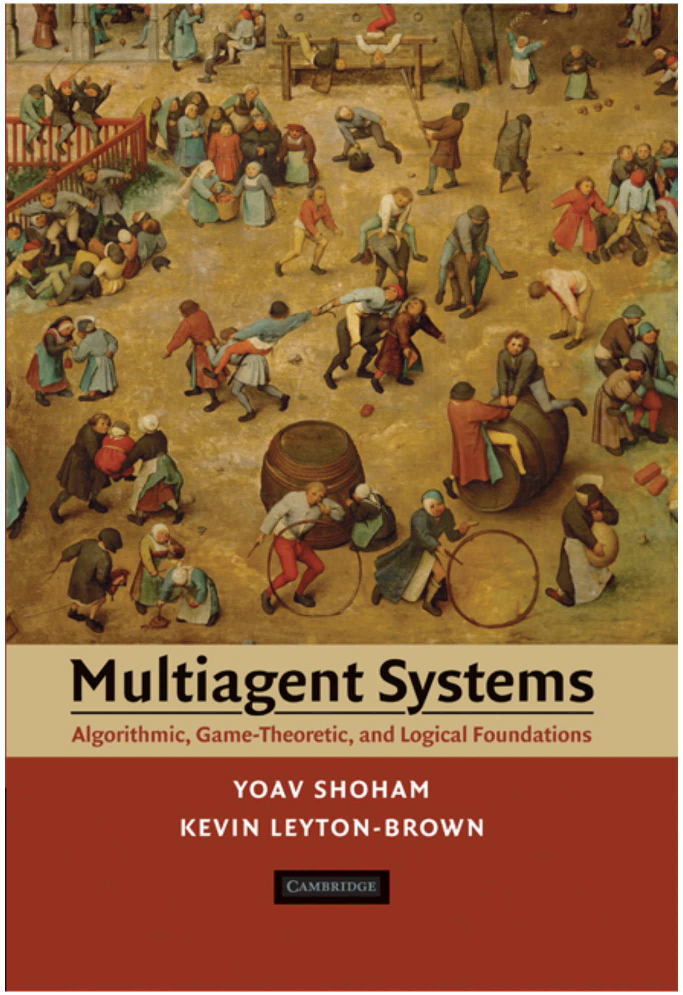
Literature


Artificial Intelligence
A Modern Approach
SECOND EDITION
The Intelligent Agent Book
Stuart Russell • Peter Norvig
Prentice Hall Series in Artificial Intelligence

Chapters 2, 6, 13, 15- 17

http://aima.cs.berkeley.edu

Literature

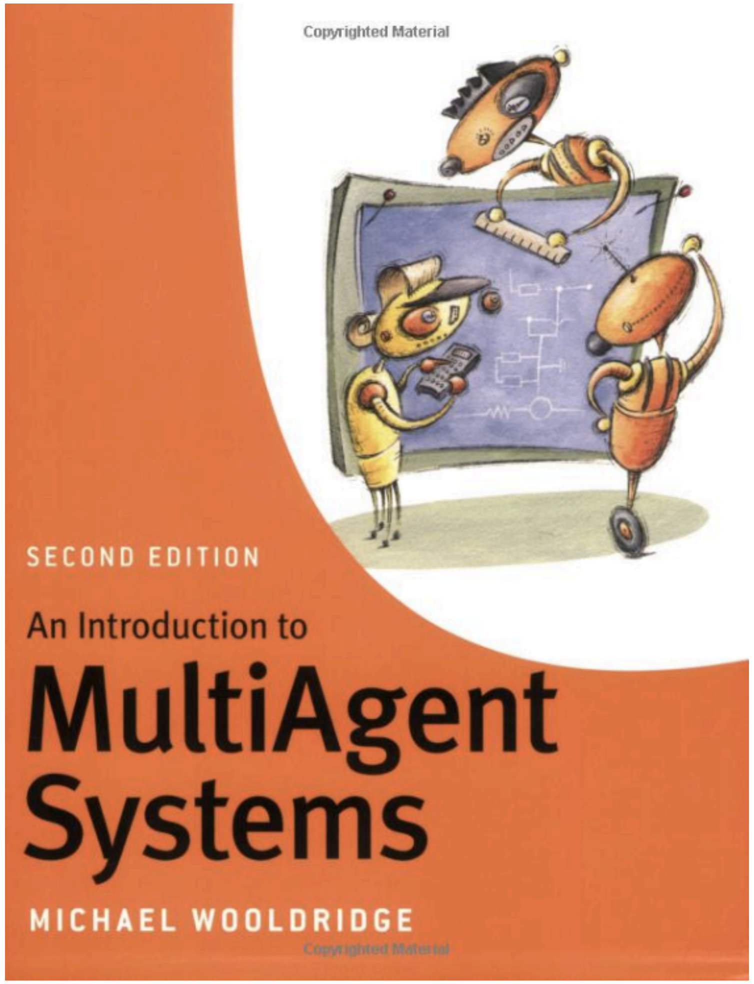

Multiagent Systems

Algorithmic, Game-Theoretic, and Logical Foundations

YOAV SHOHAM

KEVIN LEYTON-BROWN

CAMBRIDGE



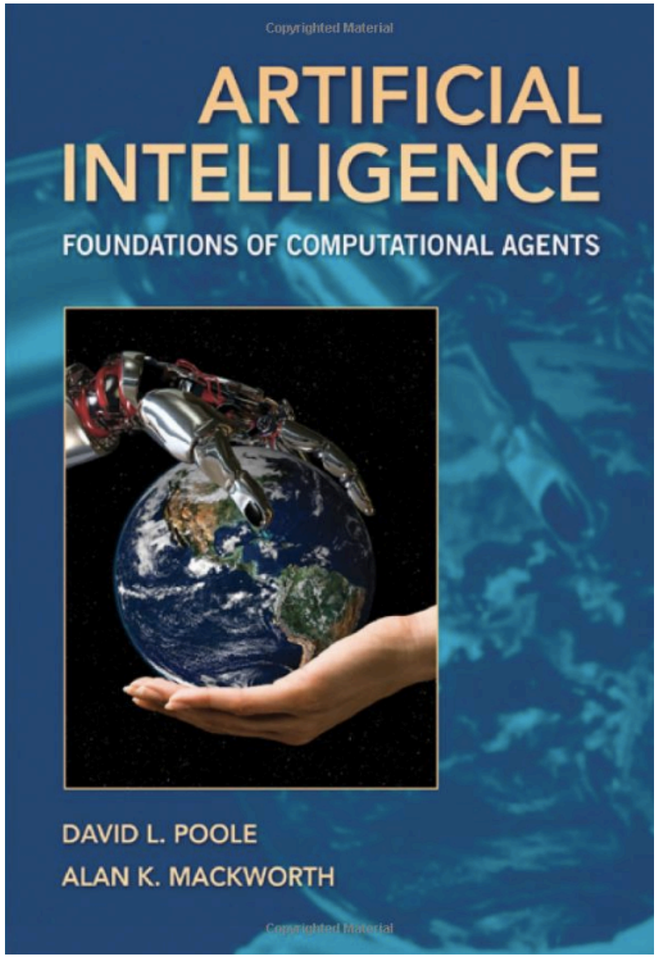
SECOND EDITION

An Introduction to

MultiAgent Systems

MICHAEL WOOLDRIDGE

Literature


Copyrighted Material

ARTIFICIAL INTELLIGENCE

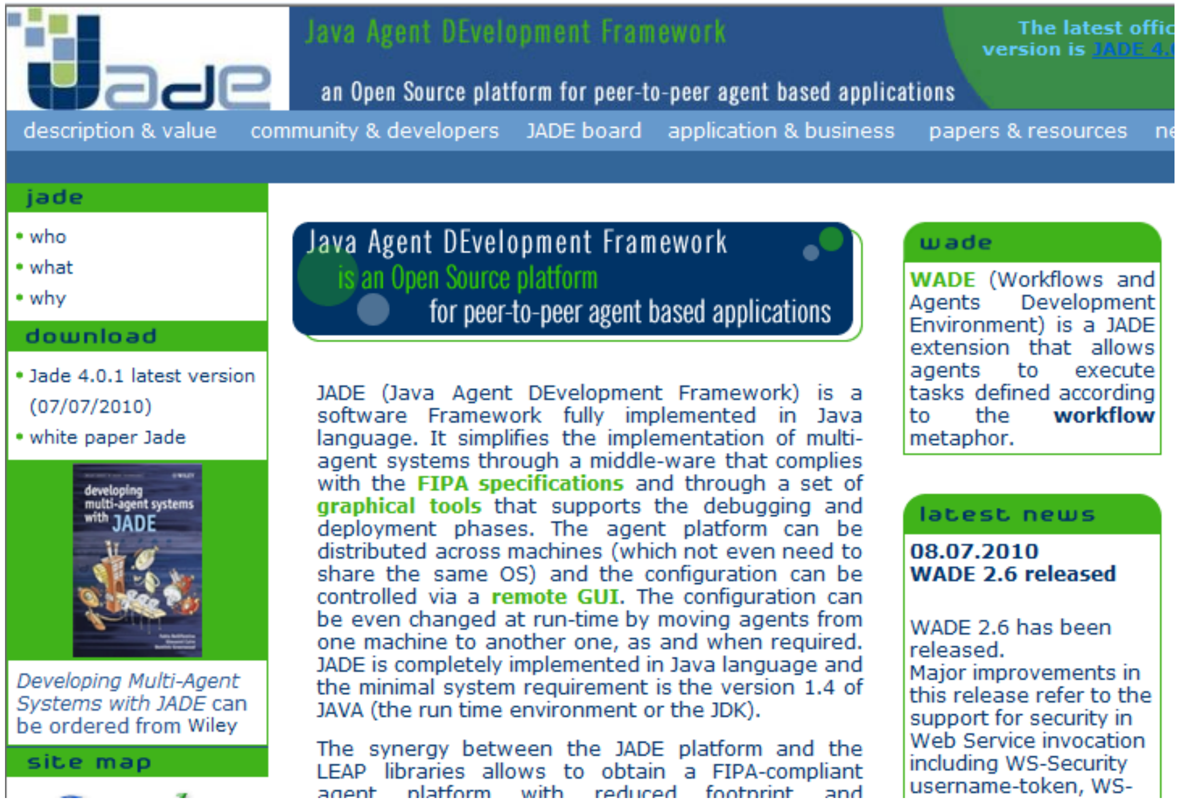
FOUNDATIONS OF COMPUTATIONAL AGENTS

DAVID L. POOLE

ALAN K. MACKWORTH

Copyrighted Material

Java Agent Development Framework



Java Agent DEvelopment Framework

an Open Source platform for peer-to-peer agent based applications

The latest offic
version is JADE 4.

description & value community & developers JADE board application & business papers & resources ne

jade
- who
- what
- why

download
- Jade 4.0.1 latest version (07/07/2010)
- white paper Jade

Developing Multi-Agent Systems with JADE can be ordered from Wiley

site map

Java Agent DEvelopment Framework
is an Open Source platform
for peer-to-peer agent based applications

JADE (Java Agent DEvelopment Framework) is a software Framework fully implemented in Java language. It simplifies the implementation of multi-agent systems through a middle-ware that complies with the **FIPA specifications** and through a set of **graphical tools** that supports the debugging and deployment phases. The agent platform can be distributed across machines (which not even need to share the same OS) and the configuration can be controlled via a **remote GUI**. The configuration can be even changed at run-time by moving agents from one machine to another one, as and when required. JADE is completely implemented in Java language and the minimal system requirement is the version 1.4 of JAVA (the run time environment or the JDK).

The synergy between the JADE platform and the LEAP libraries allows to obtain a FIPA-compliant agent platform with reduced footprint and

wade

WADE (Workflows and Agents Development Environment) is a JADE extension that allows agents to execute tasks defined according to the **workflow** metaphor.

latest news

08.07.2010
WADE 2.6 released

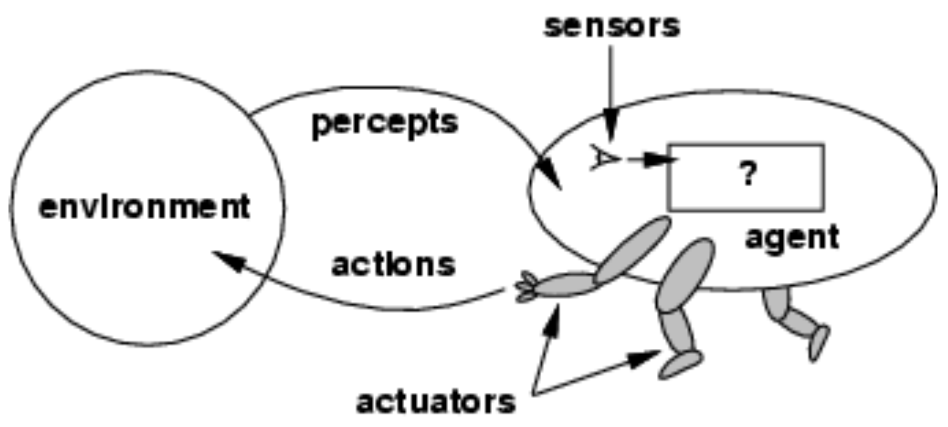
WADE 2.6 has been released.
Major improvements in this release refer to the support for security in Web Service invocation including WS-Security username-token, WS-

http://jade.tilab.com/

What is an Agent?

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

Agents and environments

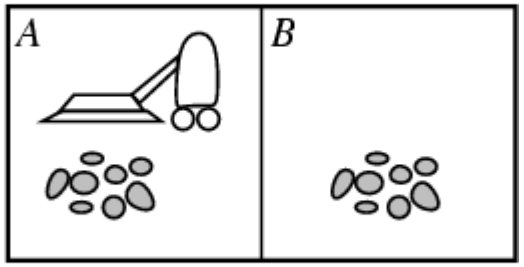


- The agent function maps from percept histories to actions:

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

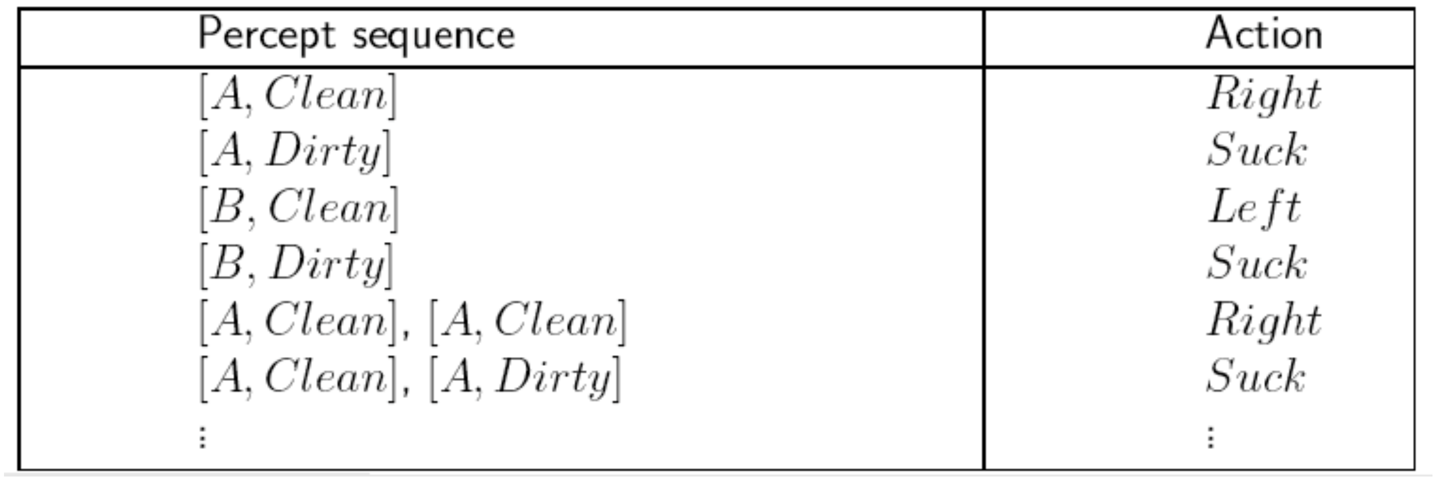
- The agent program runs on the physical architecture to produce *f*
- Agent = architecture + program

Vacuum-Cleaner World



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

A Vacuum–Cleaner Agent



Percept sequence	Action
$[A, Clean]$	$Right$
$[A, Dirty]$	$Suck$
$[B, Clean]$	$Left$
$[B, Dirty]$	$Suck$
$[A, Clean], [A, Clean]$	$Right$
$[A, Clean], [A, Dirty]$	$Suck$
$\vdots$	$\vdots$

Capabilities

- An agent is capable of **flexible** action in some environment

- By flexible, we mean:
  - reactive
  - pro-active
  - social

Reactivity

- If a program's environment is guaranteed to be fixed, the program need never worry about its own success or failure – program just executes blindly
  - Example of fixed environment: compiler

- The real world is not like that: things change, information is incomplete. Many (most?) interesting environments are *dynamic*

- A *reactive* system is one that maintains an ongoing interaction with its environment, and responds to changes that occur in it (in time for the response to be useful)

Proactiveness

- Reacting to an environment is easy (e.g., stimulus → response rules)
- But we generally want agents to *do things for us*
- Hence *goal directed behavior*
- Pro-activeness = generating and attempting to achieve goals
  - Not driven solely by events
  - Taking the initiative
  - Recognizing opportunities

Balancing Reactive and Goal-Oriented Behavior

- We want our agents to be reactive, responding to changing conditions in an appropriate (timely) fashion
- We want our agents to systematically work towards long-term goals
- These two considerations can be at odds with one another
- Designing an agent that can balance the two remains an open research problem

Social Ability

- The real world is a *multi*-agent environment: we cannot go around attempting to achieve goals without taking others into account
- Some goals can only be achieved with the cooperation of others
- *Social ability* in agents is the ability to interact with other agents (and possibly humans) via some kind of *agent-communication language* with the goal to let other agents to make *commitments*

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
- Success to be measured w.r.t. an agent-local perspective
- Performance measure: An objective criterion for success of an agent's behavior
- E.g., performance measure of a vacuum-cleaner agent could be amount of dirt cleaned up, amount of time taken, amount of electricity consumed, amount of noise generated, etc.

Rational Agents

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

Autonomous Agents

- Rationality is distinct from omniscience (all–knowing with infinite knowledge)
- Agents can perform actions in order to modify future percepts so as to obtain useful information (information gathering, exploration)
- An agent is autonomous if its behavior is determined by its own experience (with ability to learn and adapt)

Other Properties

- *Mobility*: the ability of an agent to move around an electronic network, real environment
- *(Veracity*: an agent will not knowingly communicate false information)
- *Benevolence*: agents do not have conflicting goals, and every agent will therefore always try to do what is asked
- *Learning/adaption*: agents improve performance over time

Agents and Objects

- Are agents just objects by another name?
- Object:
  - Encapsulates some state
  - Interacts synchronously with other objects
    - communicates via message passing
    - call methods / generic functions
  - Has methods, corresponding to operations that may be performed on this state

Agents and Objects

- Main differences:
  - *Agents are autonomous:* agents embody stronger notion of autonomy than objects, and in particular, they decide for themselves whether or not to perform an action on request from another agent
  - *Agents are* capable of *flexible* (reactive, pro-active, social) behavior, and the standard object model has nothing to say about such types of behavior
  - *Agents are* inherently *multi-threaded*, in that each agent is assumed to have at least one thread of active control

Main Features

- Performance measure
- Environment
- Actuators
- Sensors

Must first specify the setting for intelligent agent design

PEAS

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

PEAS

- 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

- Agent: Part-picking robot
  - Performance measure: Percentage of parts in correct bins relative to number of parts on the ground
  - Environment: Area with parts, bins
  - Actuators: Jointed arm and hand
  - Sensors: Camera, joint angle sensors

PEAS

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

- 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

- 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

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable	Yes	Yes	No
Deterministic	Strategic	Strategic	No
Episodic	[illegible]	[illegible]	[illegible]
Static			
Discrete			
Single agent			

- The environment type largely determines the agent design
- The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent

Environment Types

	Chess with a clock	Chess without a clock	Taxi driving
Fully observable	Yes	Yes	No
Deterministic	Strategic	Strategic	No
Episodic	No	No	No
Static	[illegible]	[illegible]	[illegible]
Discrete			
Single agent			

- The environment type largely determines the agent design
- The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent

Environment Types

	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			

- The environment type largely determines the agent design
- The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent

Environment Types

	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

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

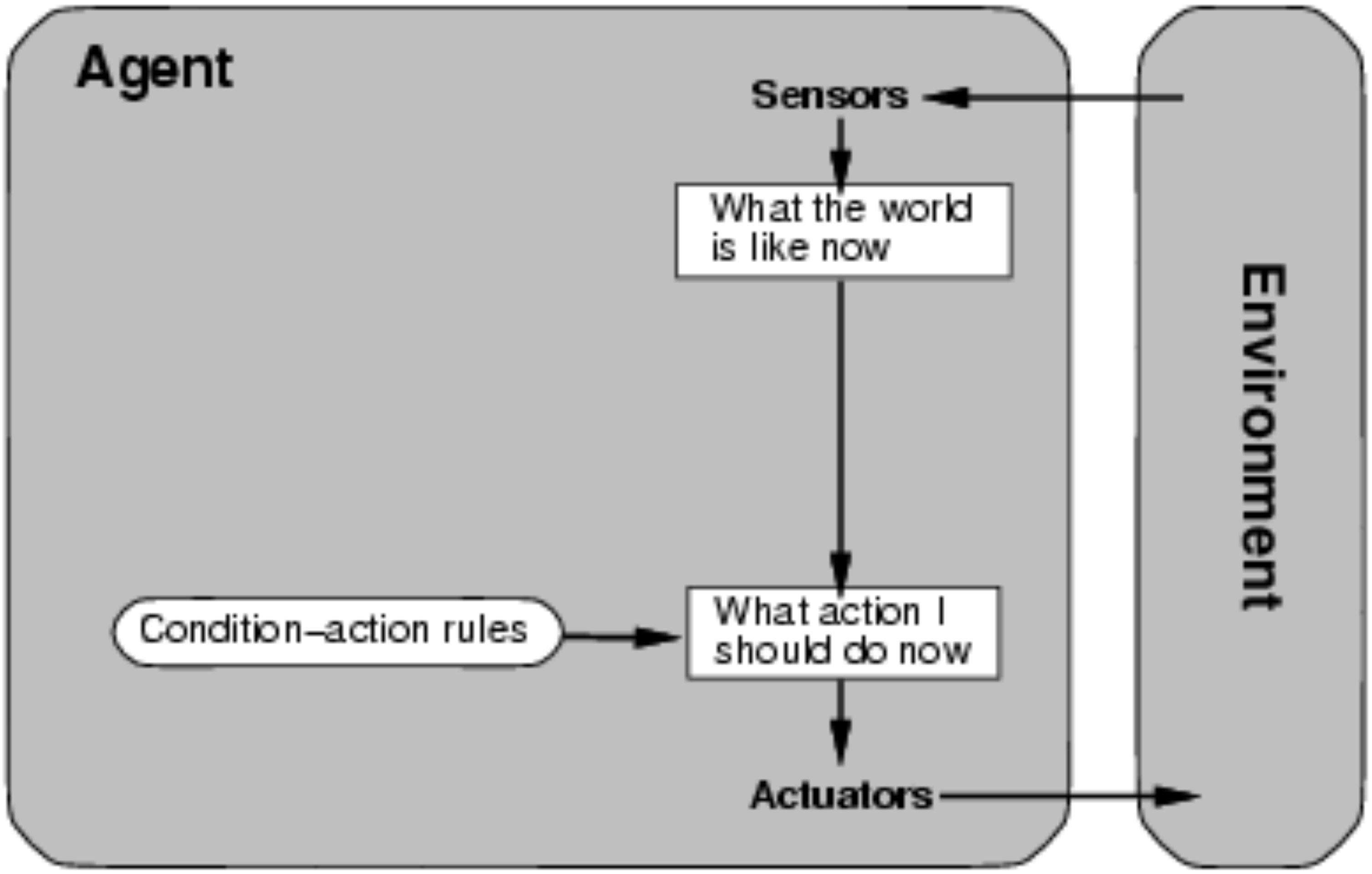
Table-Lookup Agent

- 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

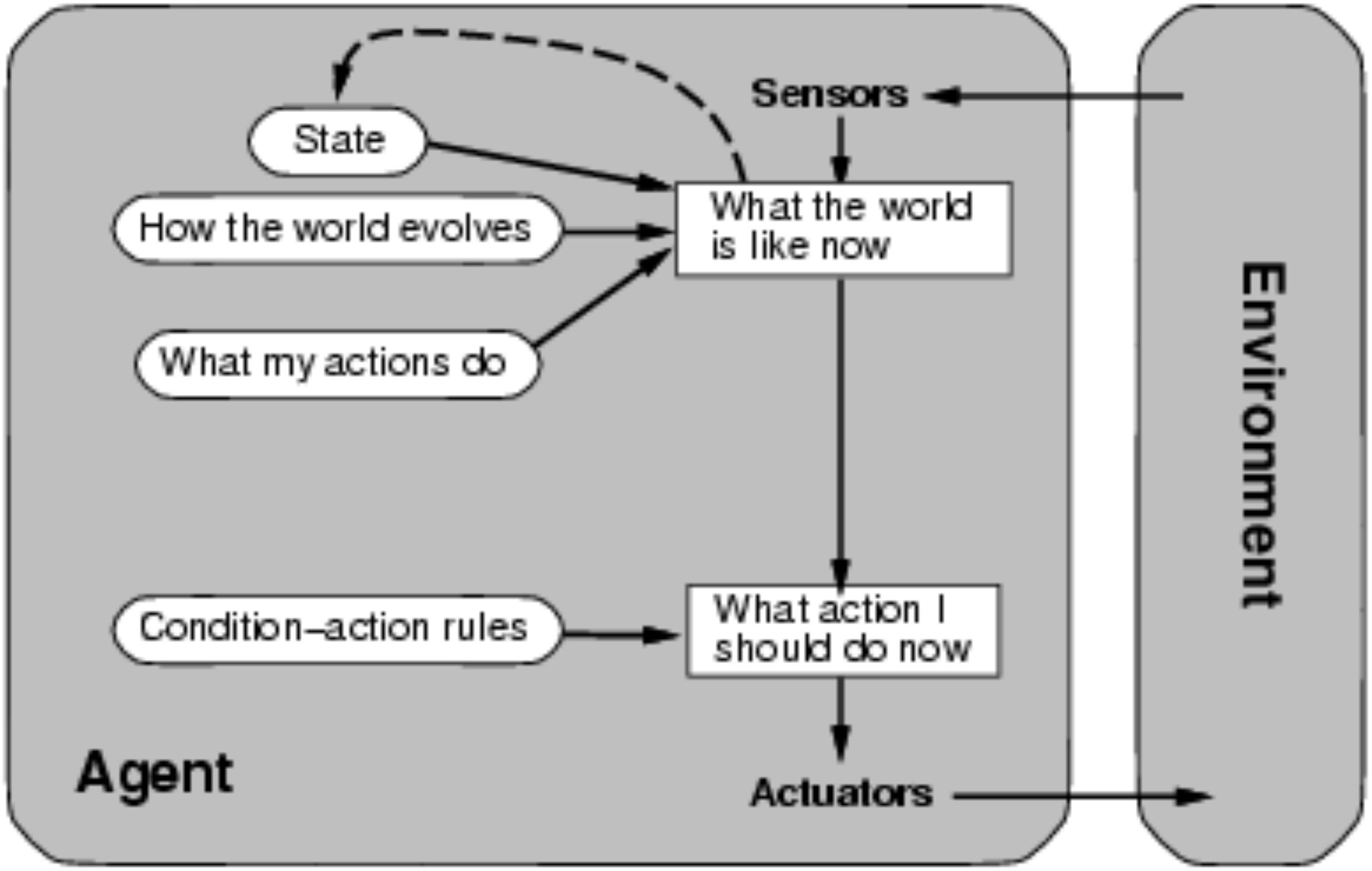
Agent Types

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

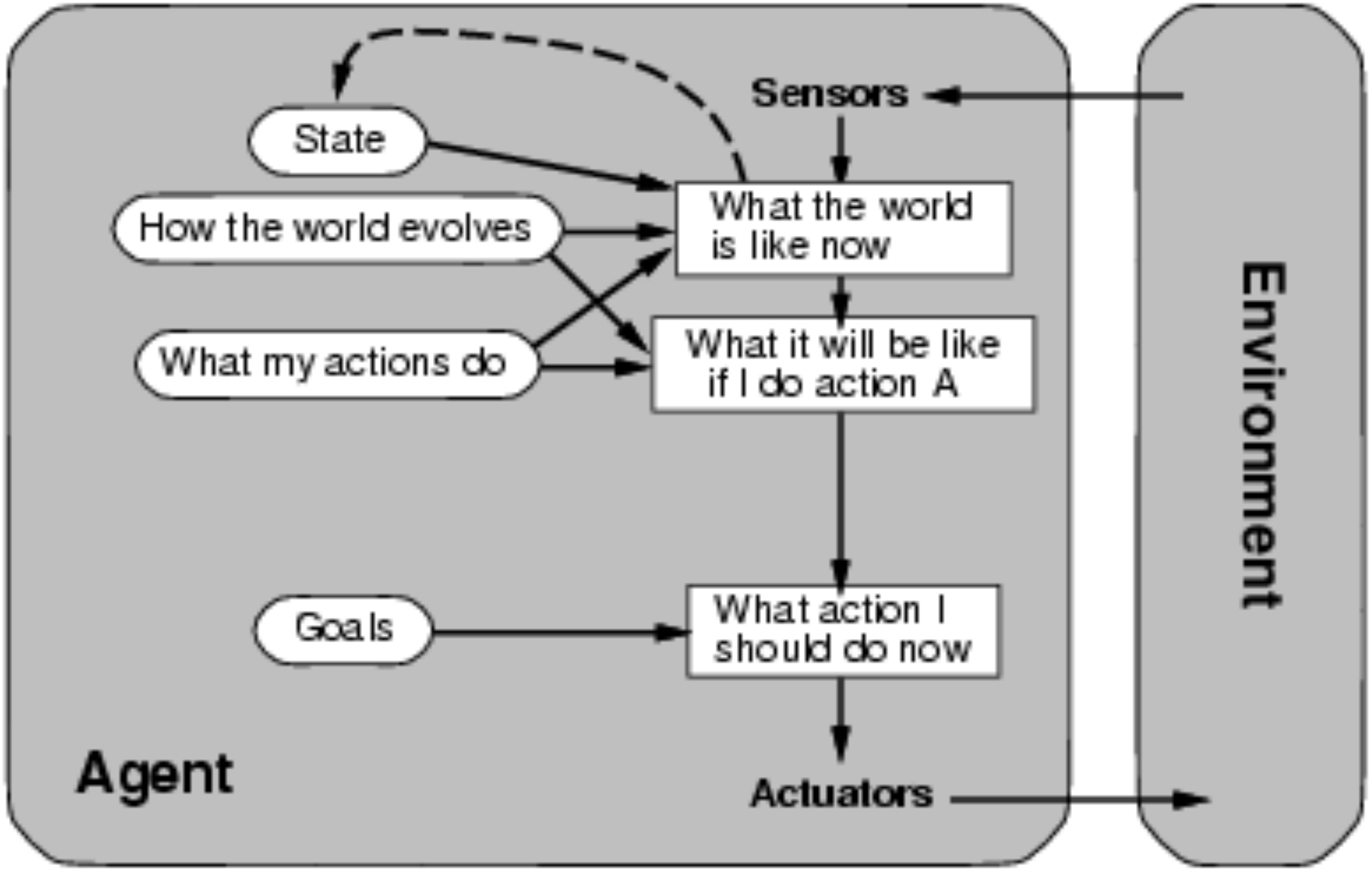
Simple Reflex Agents



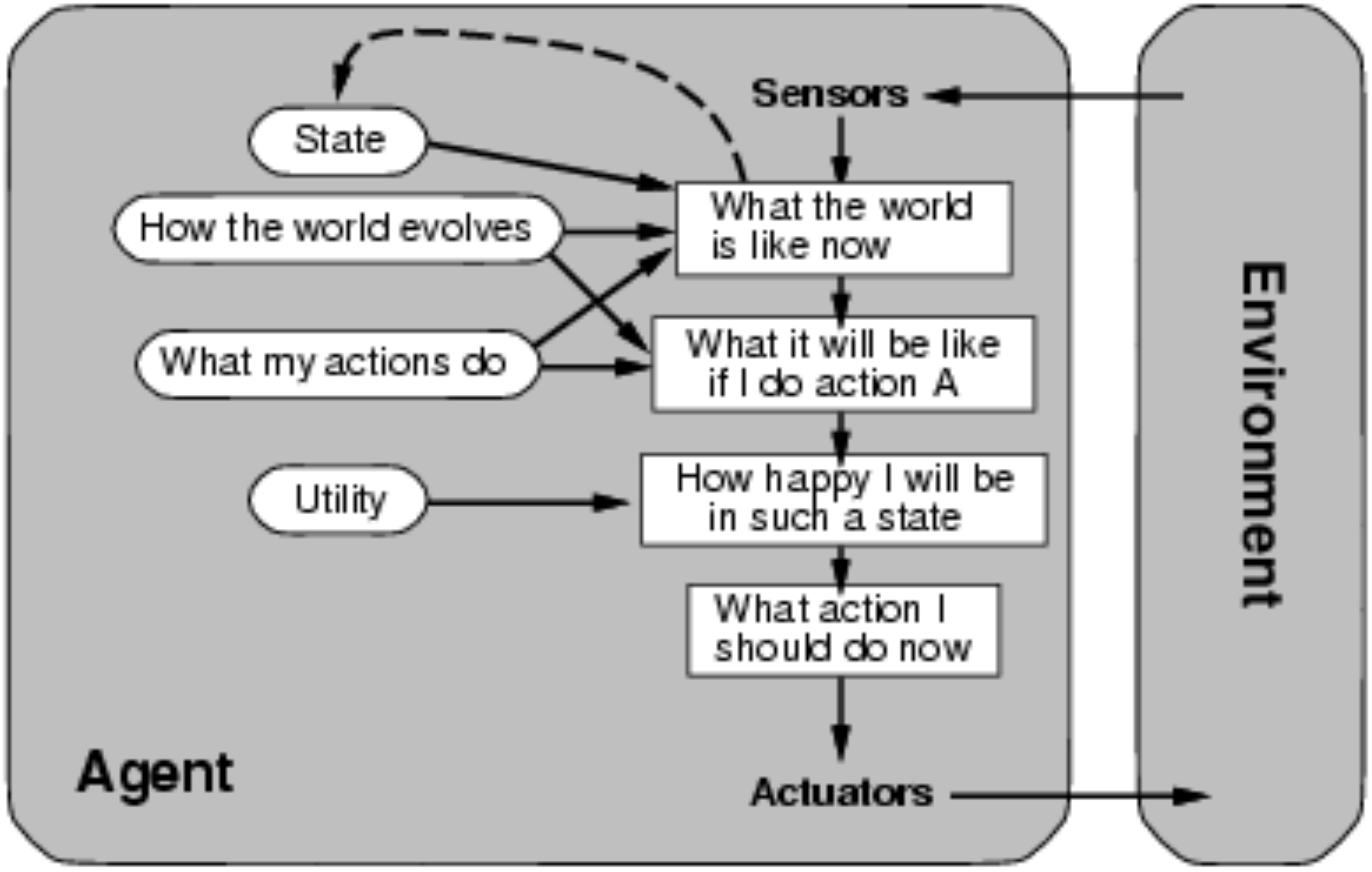
Model-Based Reflex Agents



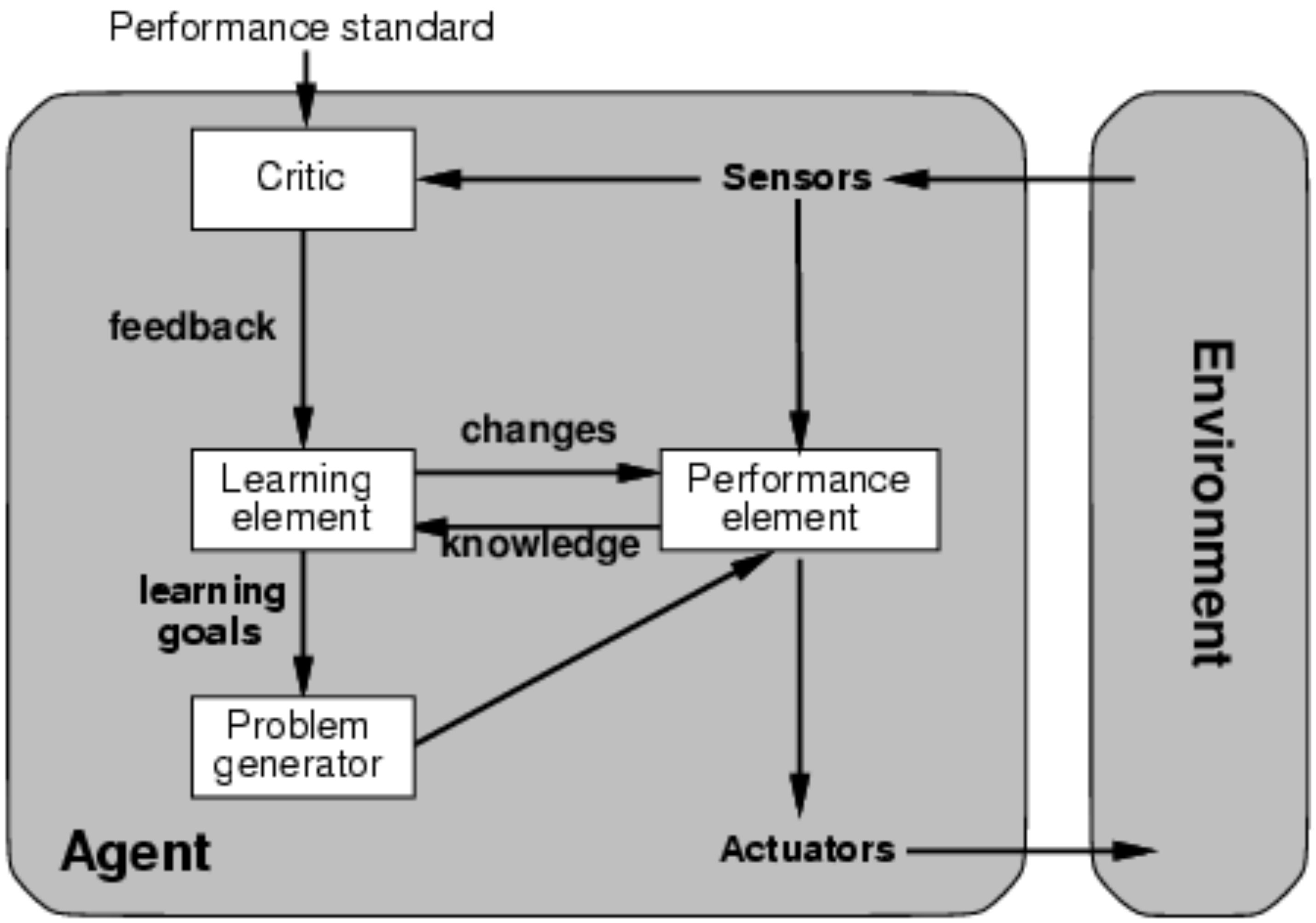
Goal-Based Agents


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
Agent
Actuators
Environment

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
Agent
Actuators
Environment

Learning Agents


Performance standard
Critic
Sensors
feedback
changes
Learning element
Performance element
knowledge
learning goals
Problem generator
Agent
Actuators
Environment