

---

# CmpE 593

# Multiagent Systems

Pınar Yolum  
[pinar.yolum@boun.edu.tr](mailto:pinar.yolum@boun.edu.tr)

Department of  
Computer Engineering  
Boğaziçi University

- Topics
- Work Schedule
- Grading
- Resources
- Academic Integrity

---

# Agents

Based largely on

*Service-Oriented Computing: Semantics, Processes, Agents*

– *Munindar P. Singh and Michael N. Huhns, Wiley, 2004*

*and*

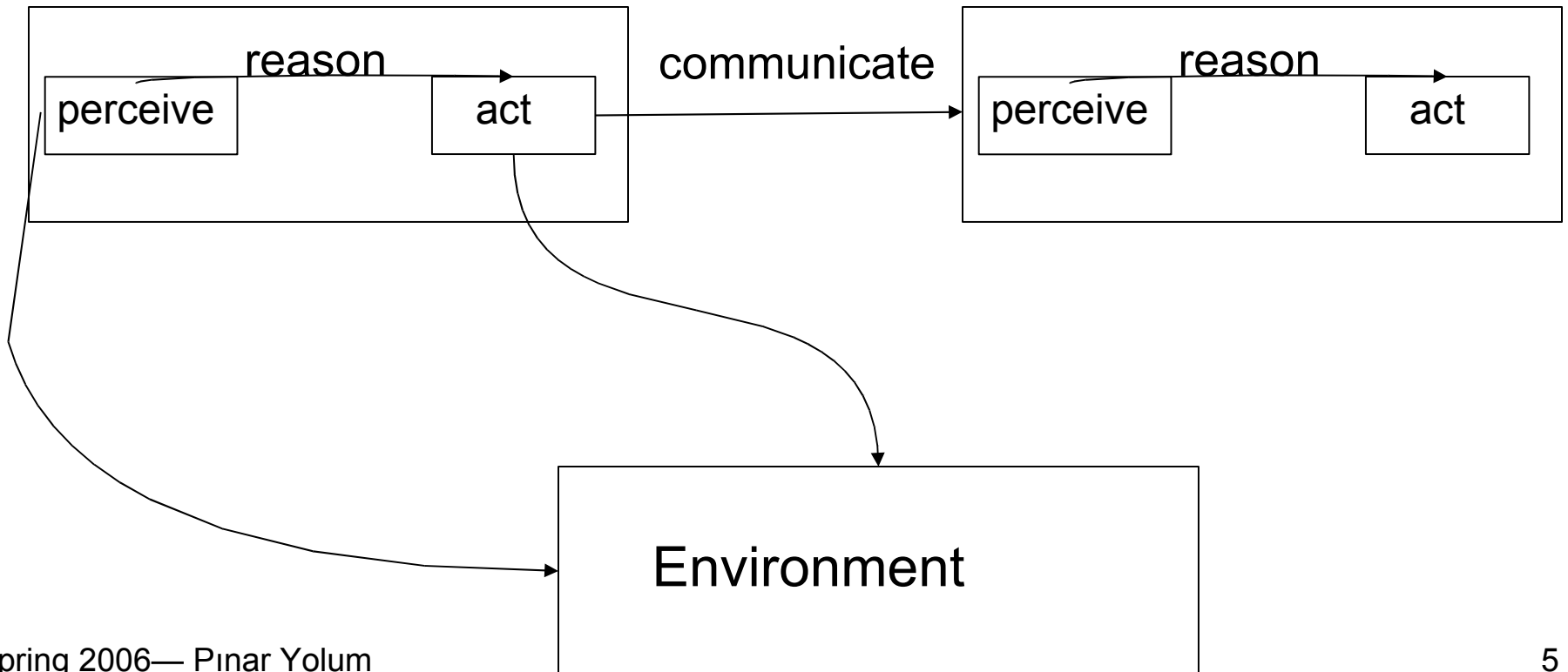
*Artificial Intelligence: A Modern Approach*

*Stuart Russell and Peter Norvig, 1995*

- So, what are agents?
- Agent characteristics
- Agent environments
- Agent types
- Agent architectures

# Agent

A persistent computation that can perceive, reason, act and communicate (Singh&Huhns)



# Agents

- Agents usually represent some principals (people, businesses, etc.)
- Principals (and thus the agents) may have contradicting preferences, goals, etc.
- To cooperate, they will need to communicate, coordinate their actions, negotiate their goals, etc.
- Ex. Your agent talks to Amazon.com agent to settle on a price for a travel book about China
- Different principals, different goals?

# Agent vs. Object

---

- Communicate
  - Request agents to perform an action
  - Call methods of objects
- Reason
  - Agents reason based on the state of the world and perform actions
  - Objects always do what they are told
- Adapt
  - Agents can learn to act differently in different circumstances
  - Objects do not change their behavior over time

# Agent Characteristics (1)

---

- Autonomous (Execution)
  - Freedom to act independently
  - Choose its own actions
  - Decide on other agents that it will interact with
- Heterogeneous (Design)
  - Could be designed and implemented by different parties
  - No need to expose internal properties
  - Might have to comply with some common requirements

# Agent Characteristics (2)

---

- Adaptive
  - Agents can learn what is good for them (and who is useful for them) and change their behavior over time
- Self-interested
  - Agents work for their owners
  - No need to carry out a task because someone asks for it

# Perspectives on Agents

---

- Artificial Intelligence:
  - Cognitive entity, possibly with feelings, emotions, etc.
- Database:
  - Information sources
  - Cooperate to carry out transactions
- Desktop software:
  - Proxy for a computation; mostly an interface
  - SNMP agents; Microsoft agent

# Agent Abstractions/1

---

- The traditional abstractions are from AI and are mentalistic
  - *beliefs*: agent's representation of the world
  - *knowledge*: (usually) true beliefs
  - *desires*: preferred states of the world
  - *goals*: consistent desires
  - *intentions*: goals adopted for action

# Agent Abstractions/2

---

- The agent-specific abstractions are inherently interactional
  - *social*: about collections of agents
  - *organizational*: about teams and groups
  - *ethical*: about right and wrong actions
  - *legal*: about contracts and compliance

# Agent Abstractions/3

---

Agents, when properly understood

- lead naturally to multiagent systems
- provide a means to capture the fundamental abstractions that apply in all major applications and which are otherwise ignored by system builders

# Agents versus AI

	Traditional AI	Agents
Entities	Stand-alone	Social: flexible autonomy, communities, responsibility
Actions (in terms of)	Cause and effect	Ethical concepts of right and wrong
Contracts (in terms of)	Simplistic obligations	Directed relationships capturing rights, duties, powers, and liabilities.

# How to Apply the Abstractions

---

Consider how the components of a large and dynamic software system in a practical situation

- Dynamism => autonomy
- Openness and compliance => ability to enter into and obey contracts
- Trustworthiness => ethical behavior

# Why Do These Abstractions Matter?

---

- Because of modern applications that demand going beyond traditional metaphors and models
  - *Virtual enterprises*: manufacturing supply chains, autonomous logistics,
  - *Electronic commerce*: utility management
  - *Communityware*: social user interfaces
  - *Problem-solving* by teams

# Environment (1)

---

- Agents are situated in an environment
- Different environments (based on Russell and Norvig)
- Accessible environment:
  - If an agent can perceive everything happening around;
  - Agents have complete information.
  - Rarely happens in practice
  - Inaccessible environments: Internet, physical world

# Environment (2)

---

- Deterministic environment:
  - Each action has a guaranteed effect
  - An agent can determine the state of the world by knowing the state before an action happens and knowing the effect of the action
  - Physical world: Deterministic?

# Environment (3)

- History freedom:
  - Episode: Single cycle of an agent perceiving and taking an action
  - Episodic: If the choice depends on the current episode and not on previous episodes
    - Easier to operate
  - Sequential: History matters
    - Needs thinking ahead
    - Chess

# Environment (4)

---

- **Static environment:**
  - The environment only changes by the actions of the agent
  - Ex: Chess
- **Dynamic environment:**
  - Other agents' actions can change the environment
  - The environment itself can change over time
  - Ex: Internet

# Environment (5)

- Discrete environment:
  - Fixed, finite number of actions and percepts
  - Chess: At each time point, the number of actions is finite
- Continuous environment:
  - Not composed of discrete units
  - Either number of actions or percepts infinite
- Possible to convert continuous environments into discrete environments (with loss of precision)

# Environment (6)

---

## Open environment:

### – Autonomous entities

- Can enter and leave frequently
- Don't need to let anyone know
- No control over what others will do

### – Cooperative entities

- No use of sellers if they cannot cooperate with customers
- Need for rules, regulations, contracts

# Generic Agent Architecture (1)

- Assume that the environment can be in one of the following states  $S: \{s_0, s_1, s_2, \dots, s_n\}$
- The set of actions that an agent can do is  $A: \{a_0, a_1, a_2, \dots, a_n\}$
- Deterministic behavior: Takes a sequence of states and determines what actions to take.  
action:  $S^* \rightarrow A$ 
  - Actions include communicative actions
- Nondeterministic effect: Takes a state and an action and determines the possible set of states  
environment:  $S \times A \rightarrow \varphi\{S\}$

# Generic Agent Architecture (2)

- History: Sequence of environment states and agent actions

$$h: s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \dots s_n$$

- Short-lived computations:  $n$  is fixed; agent eventually terminates
- Long-lived (persistent) computations: Loop forever
- Full history: If the agent remembers all the states it has been in as well as all its actions

# Reactive Agents (1)

- Stateless reactive agents: action:  $S \rightarrow A$ 
  - Don't care about the history
  - Act based on the current state now
- Famous thermostat example!
  - Perception allows the agent to get the temperature
  - The agent compares the temperature to a range of values to decide if it is OK

$$action(S) = \begin{cases} \text{heater off if temperature} = \text{OK} \\ \text{heater on otherwise} \end{cases}$$

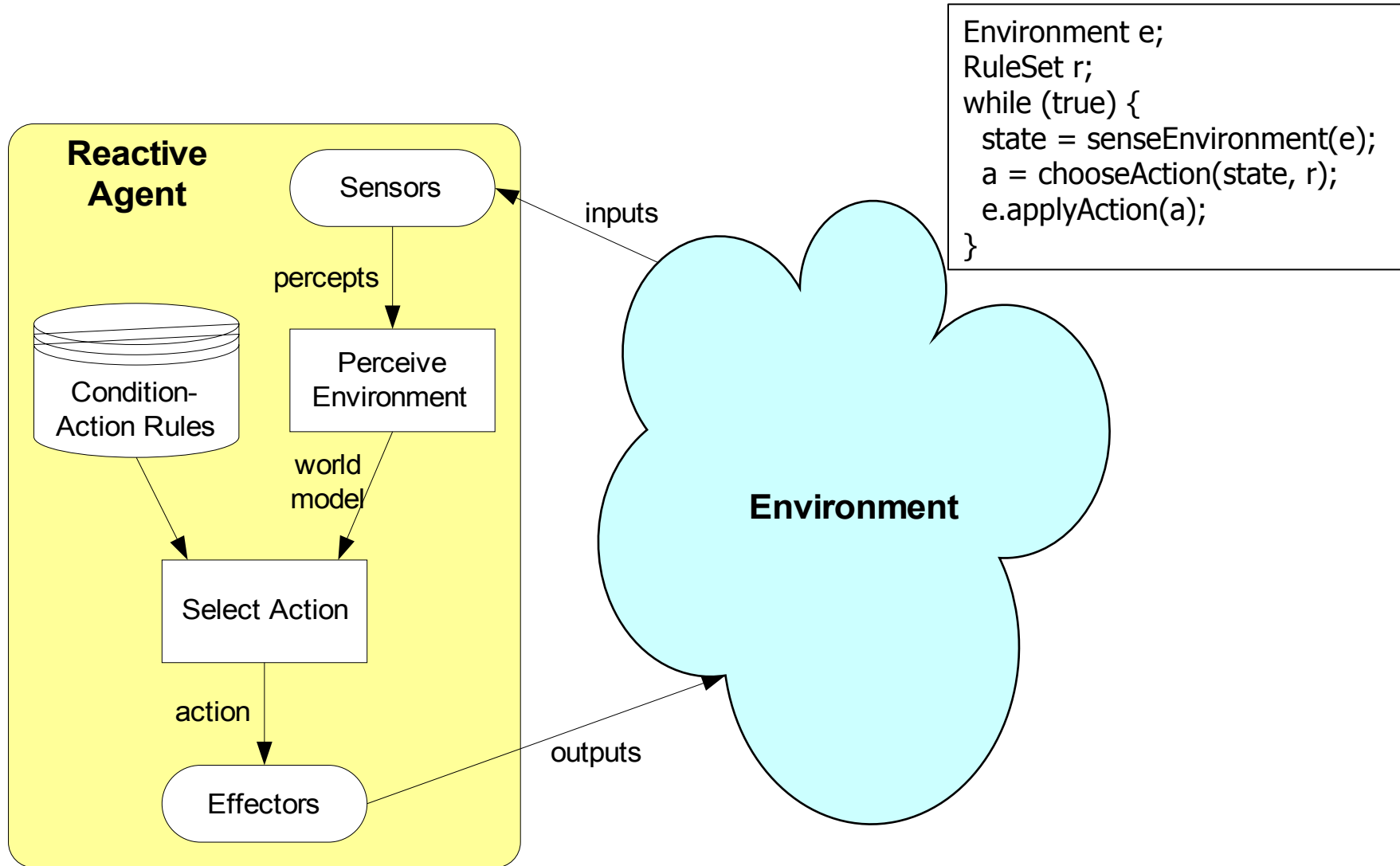
- Simple to implement; a set of condition-action rules
- Cannot be applied if the history is important

# Reactive Agents (2)

---

- Add an internal state
  - Start with an initial internal state
  - Perceive the environment and update the internal state accordingly
  - Act based on the internal state
- Still simple to implement; need more memory for the state and a set of condition-action rules

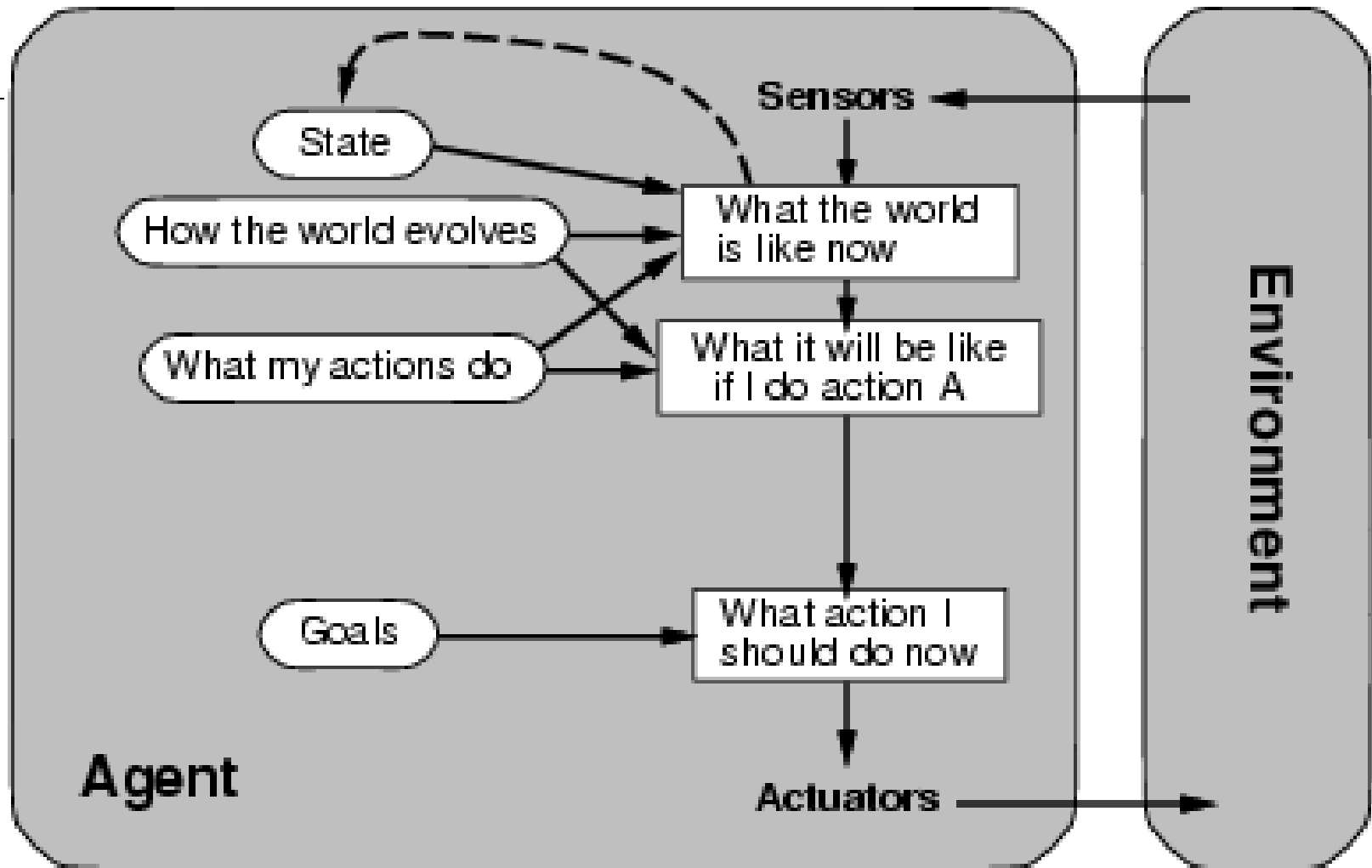
# A Reactive Agent in an Environment



# Goal-Oriented Agents

- Add a goal
  - What is the agent trying to do?
  - Drive to Taksim; buy a book online
  - Perform actions that will realize the goal
- A set of condition-action rules is not enough
- Think about a goal state where the desired goal holds
- *Plan or search* a sequence of actions such that applying those actions will transform the current state into the goal state

# Goal-based agents (R&N)



# Utility-Based Agents (1)

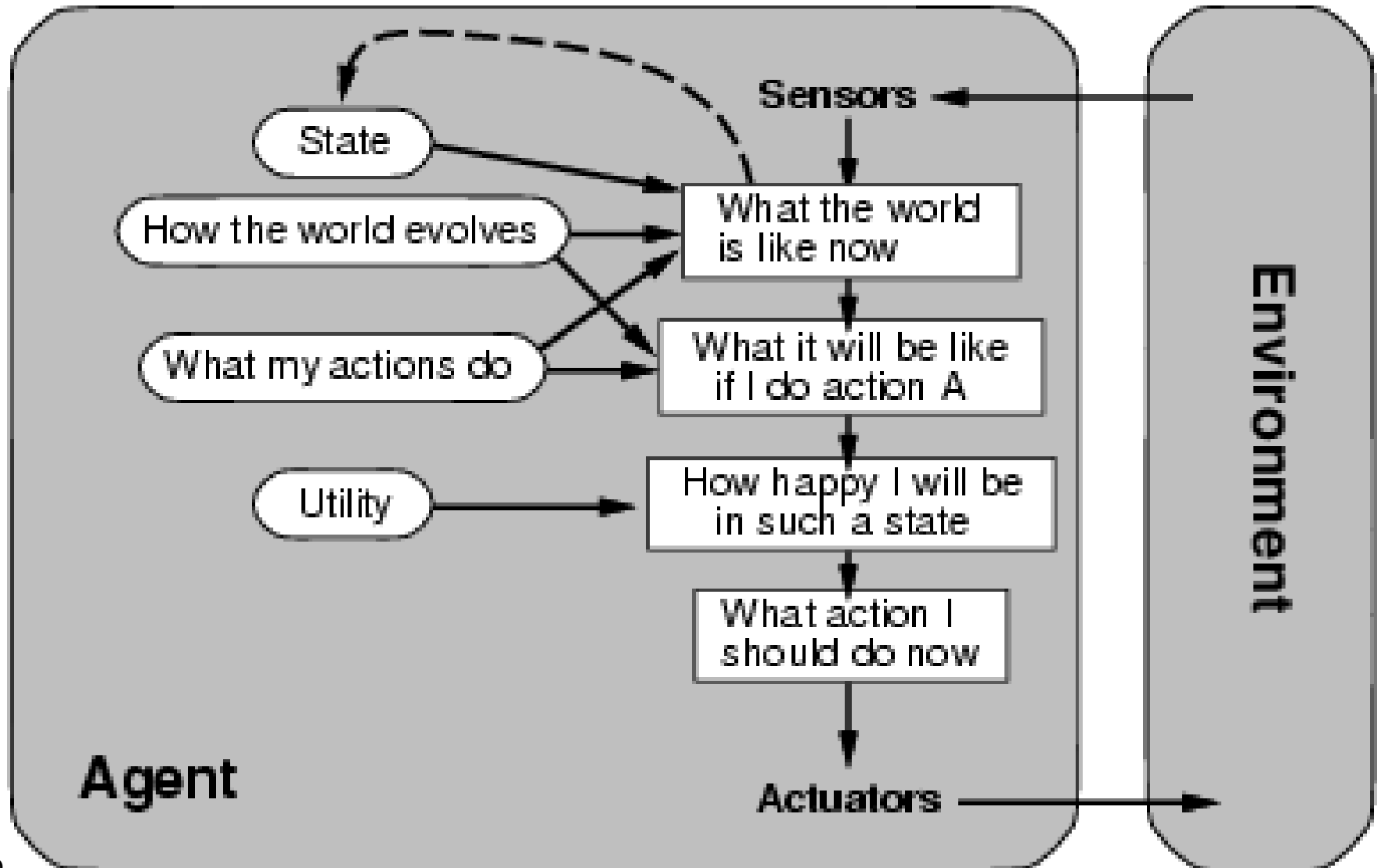
- A goal-oriented agent may achieve its goals in a number of ways.
- Nothing is said about whether one way is better than another
- However, one path can be preferred over another because it's shorter, cheaper, etc.
- Utility:
  - Quantitative measure of a chosen path
  - A function from states to real numbers

# Utility-Based Agents (2)

---

- Agents may have conflicting goals\*
  - Buy a book with a low cost and buy it fast
  - Which goal should have precedence over the other?
- Assign a utility to each goal
- Maximize expected utility

# Utility-based agents (R&N)



# Agent Architectures

---

- Logic-Based
- Reactive
- Belief-Desire-Intention (BDI)
- Layered Architecture

- Consists of:
  - Formal system described by
    - Syntax: How to make sentences
    - Semantics: How sentences relate to facts
  - Proof theory
    - A set of rules for entailment
- Kinds
  - Propositional logic
  - First-order logic
  - Temporal logic
  - Fuzzy logic

# Logic Based Agents

---

- Decision making is realized through logical deduction
- View the agents as particular type of knowledge based system
  - Contains an explicitly represented symbolic model of the world
  - Takes decisions via symbolic reasoning
- Problems:
  - Translating the real world into an accurate adequate symbolic description, in real-time
  - How to represent information symbolically about complex real-world entities

# A Rational Agent

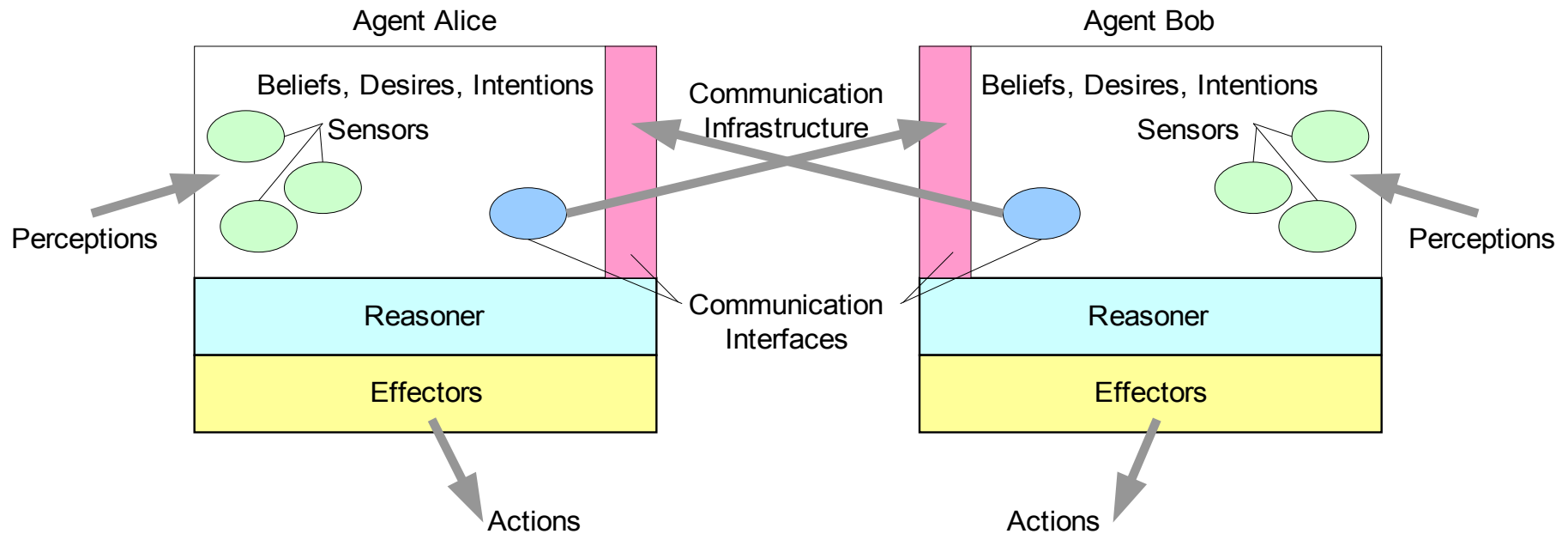
---

Rationality depends on...

- The performance measure for success
- What the agent has perceived so far
- What the agent knows about the environment
- The actions the agent can perform

An **ideal rational agent**: *for each possible percept sequence, it acts to maximize its expected utility, on the basis of its knowledge and the evidence from the percept sequence*

# Cognitive Architecture for an Agent



# Reactive Architecture

---

- Does not rely on symbol manipulation, e.g., Rodney Brooks' subsumption architecture
- Intelligent behavior can be generated without explicit representations proposed by symbolic AI
- Intelligent behavior can be generated without explicit abstract reasoning
- Intelligence is an emergent property of certain complex systems

# Subsumption Architecture

---

- A hierarchy of task-accomplishing behaviors
- Each behavior is a rather simple rule-like structure
- Each behavior competes with others to exercise control over the agent
- Lower layers present more primitive kinds of behavior
- In terms of computation, the resulting systems are extremely simple

# BDI Agents

- Agents are represented with beliefs, desires, and intentions.
- Beliefs:
  - What the agent thinks is true in the environment
  - Need not be correct
- Desires:
  - The environment that the agent prefers to exist in
  - Can be inconsistent

- Intentions:
  - The environment that the agent is trying to achieve
  - Subset of agent's desires
  - Should be consistent with each other
  - Agent's actions are related to its intentions
  - Agent carries out a plan to realize an intention

# Means-end Reasoning

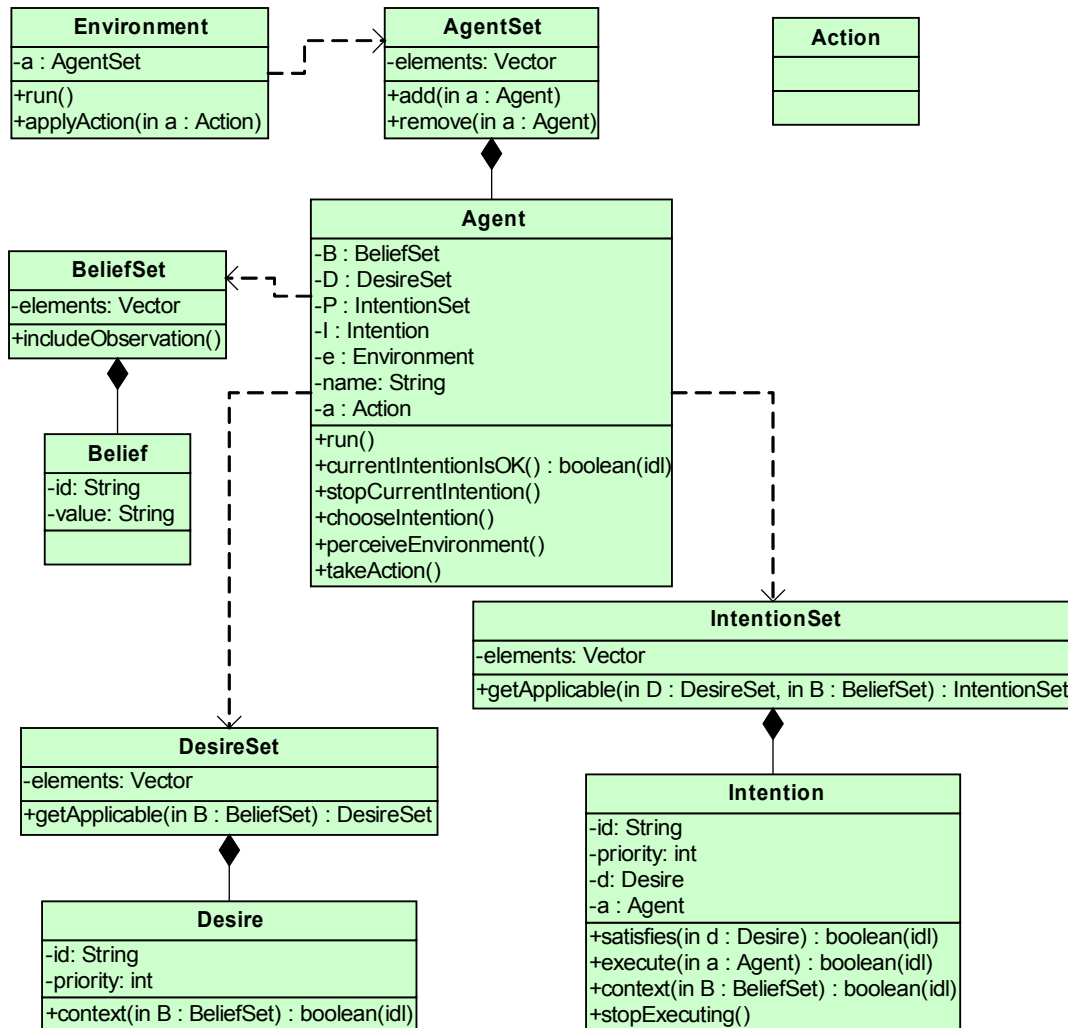
---

- Decide what intentions to adopt or revise
- What actions to perform
- The agent does multiple things concurrently
  - An agent thread constantly checks the environment
  - An agent thread uses the environment input to revise its internal working
  - Possibly more threads for communication, acting, etc.

# Revision

- Action revision
  - Check if the state of the environment is compatible with its actions
  - If they are not compatible, revise actions
- Example:
  - Intention to be at the university at 10:00 am
  - Perceive from the environment that it is already 10:15
- Realizing an intention
  - Carrying out a plan of actions
  - Modified intention means modified plan
- Incompatible environment and intention
  - Drop intention (Too late to go)
  - Revise intention (Try to be there at 10:30)

# Architecture of BDI-Based Agent



## Execution Cycle:

2. New information arrives that updates beliefs and goals
3. Actions are triggered by new beliefs or goals
4. A triggered action is intended
5. An intended action is selected
6. That intention is activated
7. An action is performed
8. New beliefs or goals are stored
9. Intentions are updated

# Revision

- Belief revision
  - Perceive a new belief that contradicts old belief
- Example:
  - Belief: Course starts at 10:00
  - Talk to a friend and learn that the course starts at 11:00
- Need to resolve it
  - Ask other people; look up on the Web
  - Rules for belief revision
    - New belief overrides an inconsistent old belief

# Knowledge (1)

- Knowledge=“True” belief?
- Knowledge representation for reasoning
  - Expressed in machine-understandable form
  - Natural Language?
- Two aspects:
  - Syntax: Describe what a sentence is
    - Allowed sequences of characters, words, etc.
  - Semantic: Determines the facts that the sentence corresponds to
- Example:  $x \leq y$
- Need precise syntax and semantics to reason

# Knowledge (2)

- Facts in real world vs. representation in agents
- Reasoning derives new knowledge
- Reasoning on facts and representation should yield the same knowledge
- Entailment: Deriving new sentences based on old ones
- $KB \models \alpha$
- Inference Procedure
  - Generate all sentences entailed by KB
  - Check if alpha is entailed by KB

# Inference Procedure

---

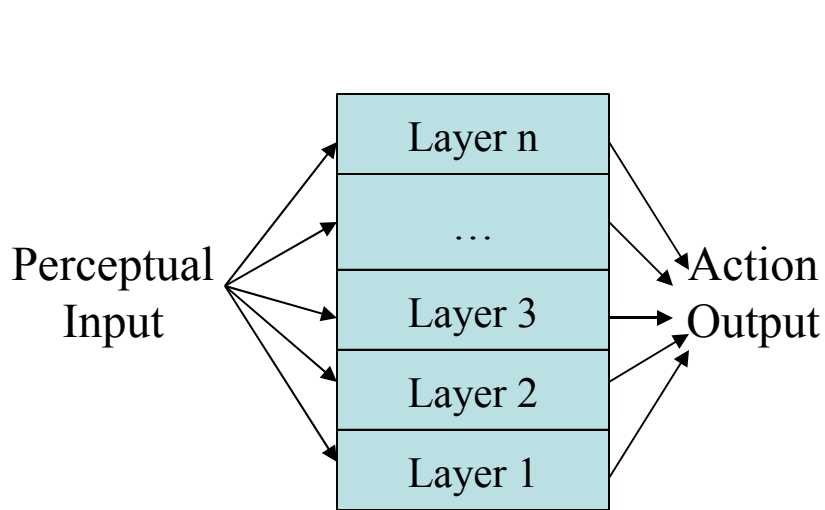
- Inference Procedure
  - Generate all sentences entailed by KB
  - Check if alpha is entailed by KB
- Sound
  - If the procedure generates entailed sentences only
- Complete
  - If the procedure has a proof for every entailed sentence
  - Difficult to achieve

# Layered Architecture

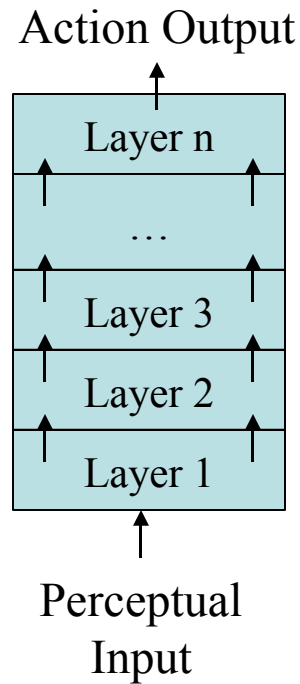
---

- Decompose the system into different layers to deal with different types of behaviors
- Typically at least two layers – to deal with reactive and proactive behaviors
- Broadly, two types of control flow within layered architectures
- Horizontal layering
- Vertical layering

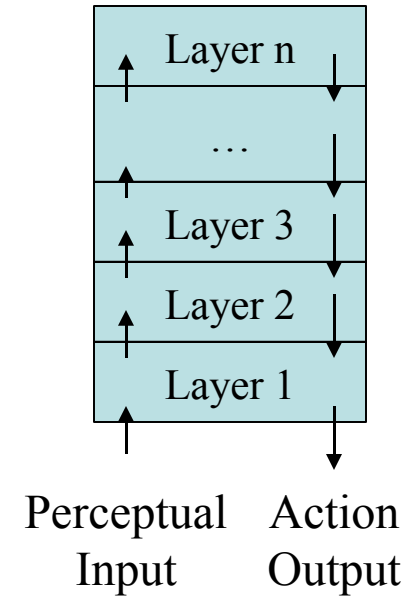
# Layered Architecture



Horizontal Layering



Vertical Layering  
(one pass control)



Vertical Layering  
(two pass control)

# Developing Agents (1)

- Depends on what is expected of the agent
- An agent that interacts with other agents over the Web
  - JAVA!
  - Better support for Web standards
  - Easy integration with other programming environments
- An agent that processes logical formulae (logic-based agents)
  - Prolog
  - Java+Prolog

# Developing Agents (2)

- Agent0, GOLOG, 3APL
  - Agent programming languages
  - Constructs for beliefs, goals, etc.
  - Specify rules to reason on these (update, delete)
  - Use an interpreter which will track which rules are fired and make modifications accordingly

# Developing Agents (3)

---

- Java Agent DEvelopment Framework (JADE)
  - FIPA-compliant
  - Available libraries for agent templates
- MadKit
  - Agents programmed in Java
  - Enforce an organization model so that each agent plays a role in the MAS
- OAA, JATLite, IBM Able