Software Agents

Tony White
arpwhite@scs.carleton.ca

Overview

- Motivation and definitions
  - Why do we need agents?
  - What is an agent?
- Agent architectures
  - technologies, issues, advantages, disadvantages
- Collaboration
  - blackboard, KQML, etc.
- Examples
  - e-commerce, network management
  - enabling technologies

Motivations

- Why do we need agents?
  - Increasingly networked, temporary connectivity increasing (wireless).
  - Data overload (e-mail, web pages, fax, …).
  - Greater exchange of digital information
  - Increasingly dependent upon electronic sources of information.
  - Desire to be ‘better informed’.

Tools

- Inadequacy of current tools
  - Browsers are user driven, Pull technology *marginally* better.
  - ‘Friendly’ software becoming more difficult to use (e.g. MS Word!)
  - WWW too polluted for casual browsing, intelligent search tools required; even search engines beginning to fail us!
    - Coverage, web pages exploiting indexing algorithms of engines, broken links.

Solution!

- Need software solution (agents) that can act in our place:
  - can interact with (say) Internet data sources
  - can process e-mail, voice, fax and other electronic message sources
  - can communicate with other agents
  - can accurately represent our needs and preferences in the networked information environment
  - can negotiate

And the solution is…*Agents*

- So, what is a software agent? *No generally agreed definition.* Has characteristics:
  - Something that acts on behalf of another
  - Is sociable, capable of meaningful interaction with other agents (and humans)
  - Can make decisions on our behalf
  - Is capable of adapting to changing environments and learning from user interaction
  - Is mobile

A Basic Definition

"Intelligent software agents are defined as being a software program that can perform specific tasks for a user and possessing a degree of intelligence that permits it to performs parts of its tasks autonomously and to interact with its environment in a useful manner."

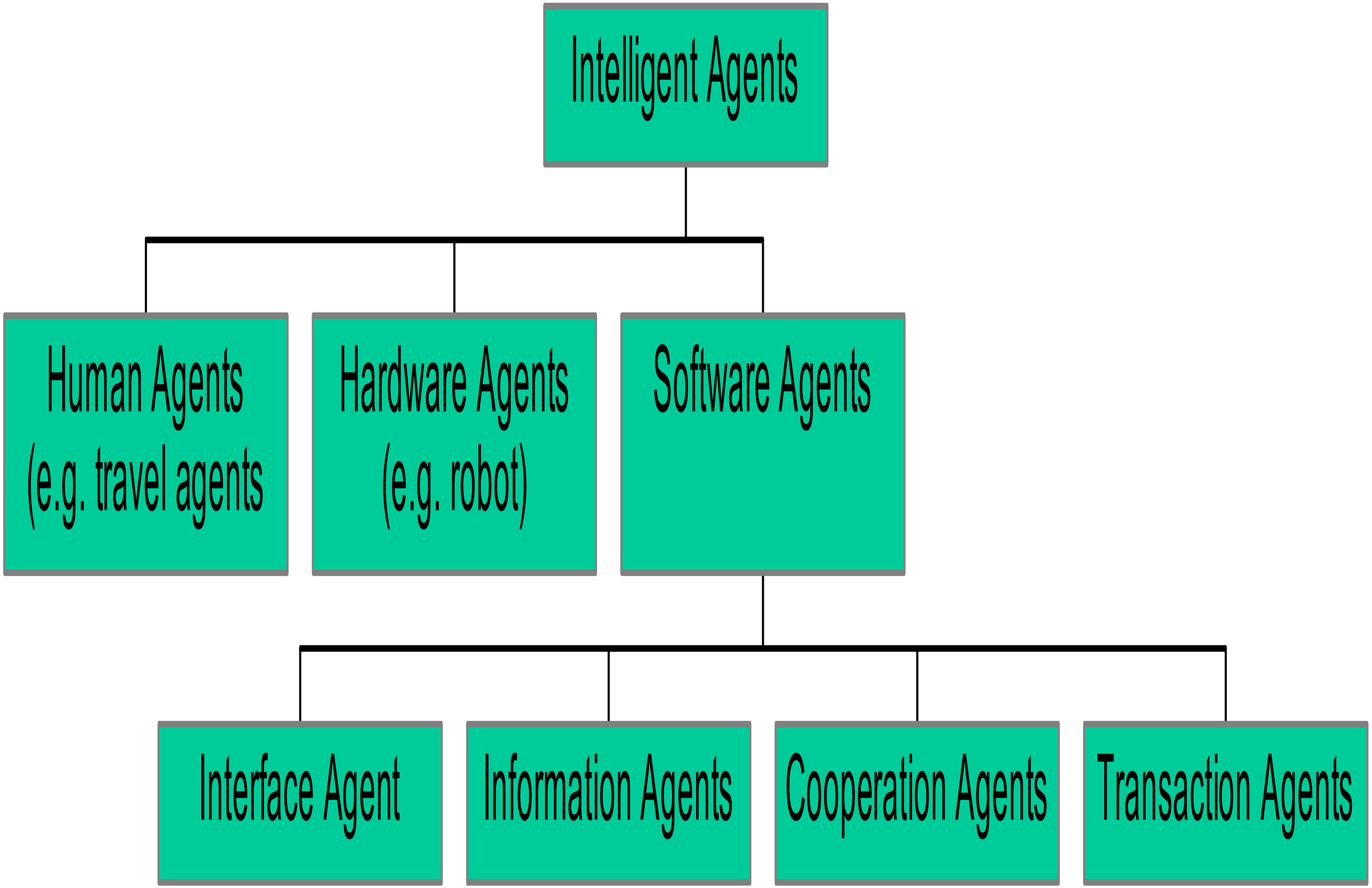
From Intelligent Software Agents

Brenner, Zarnekow and Wittig.

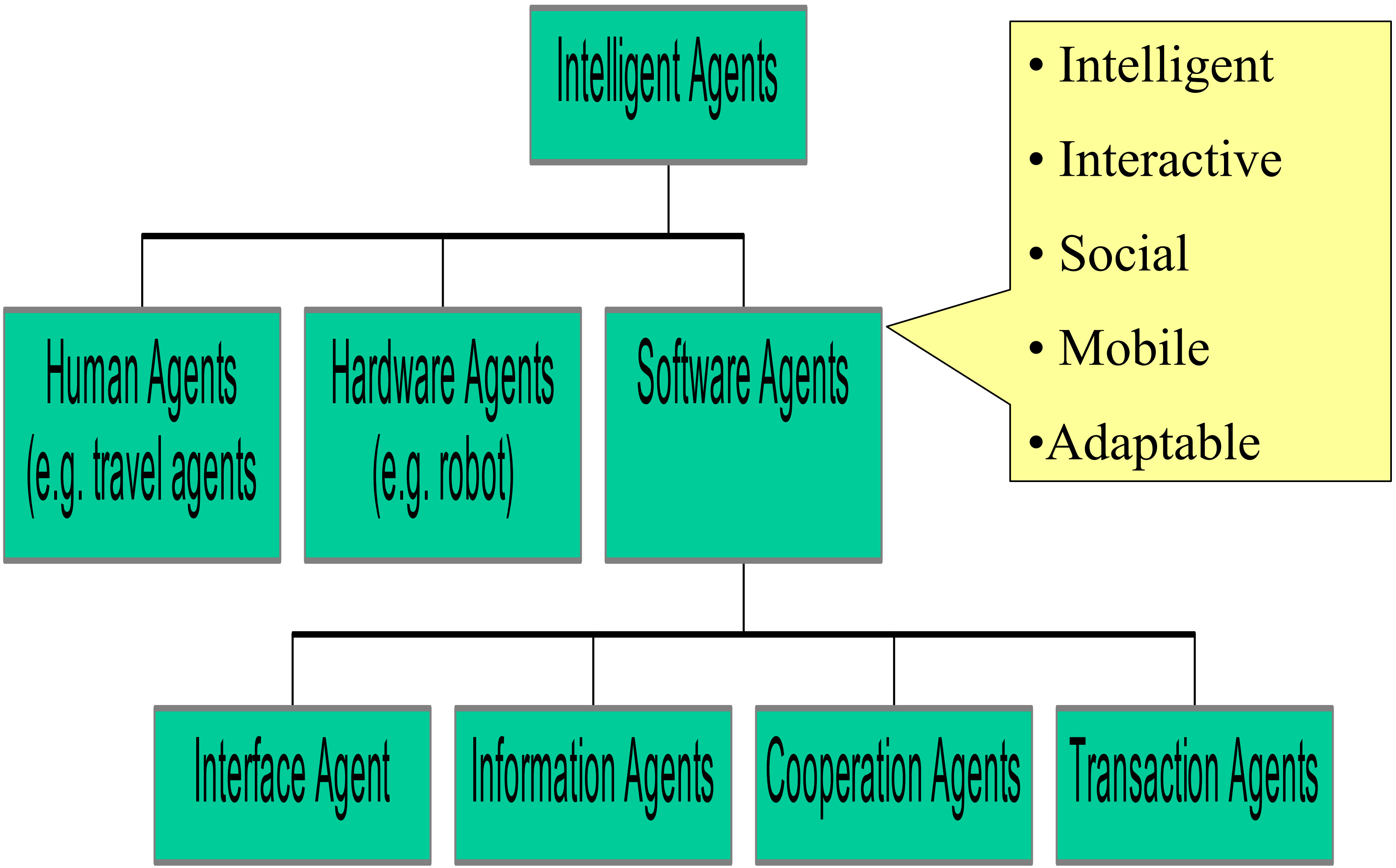
Potential agent rewards

- In the Internet:
  - **efficiency**: agent is given goal and returns the result;
  - **effectiveness**: agent can terminate search when acceptable solution found. Has a higher degree of multi-threading;
  - **transparency and optimization**: correlation between multiple data sources possible => higher quality results.

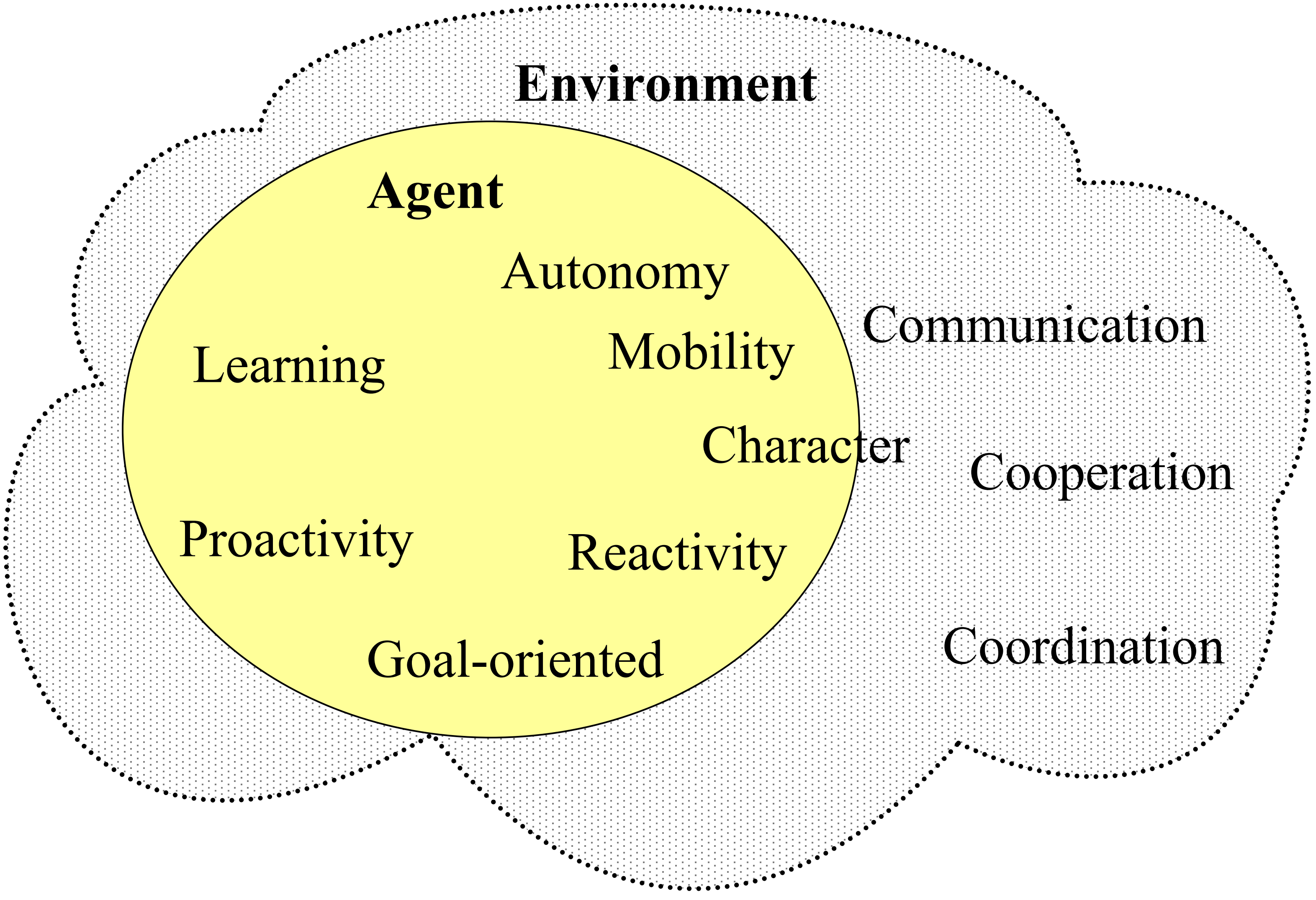
Taxonomy of Agents



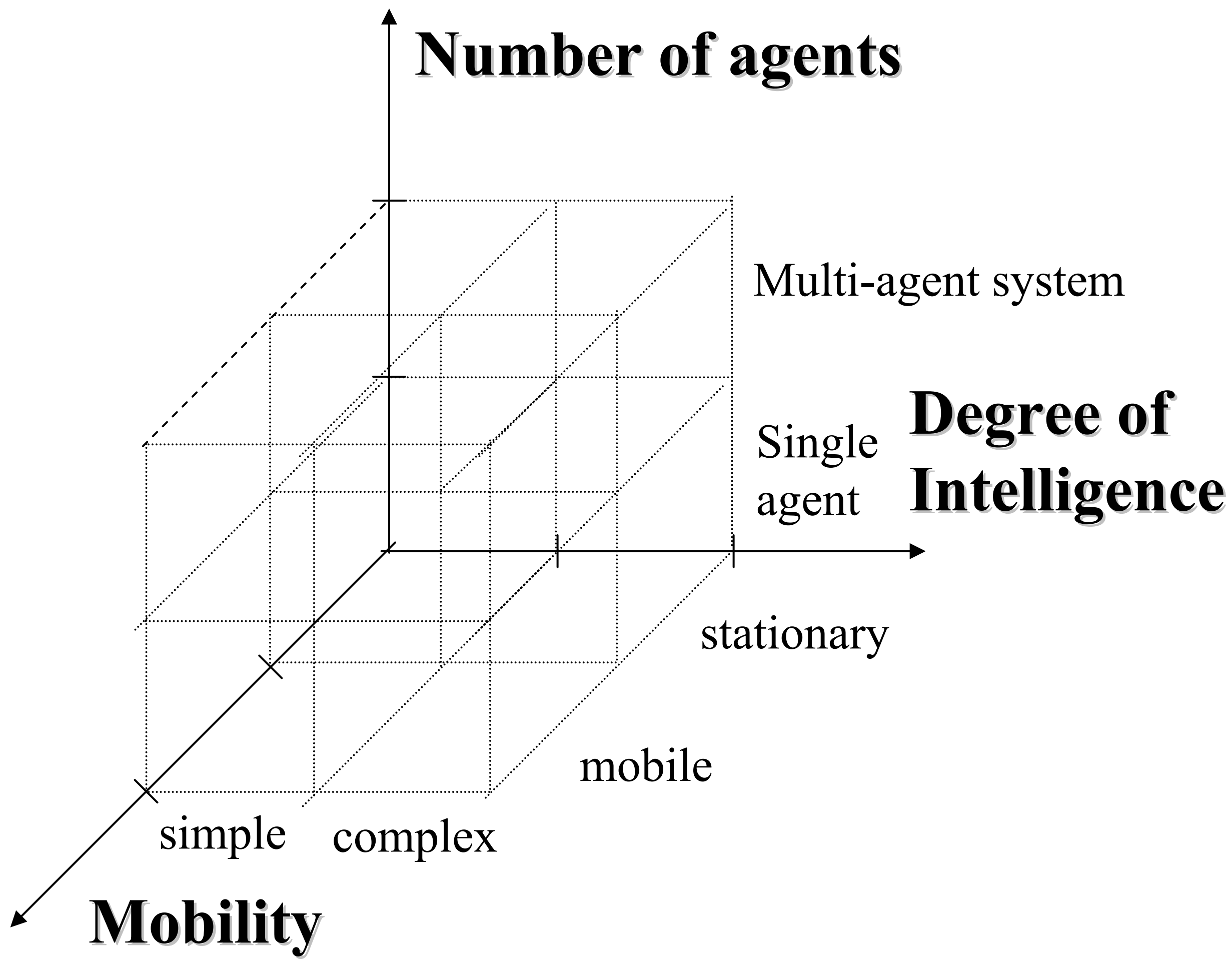
Taxonomy of Agents



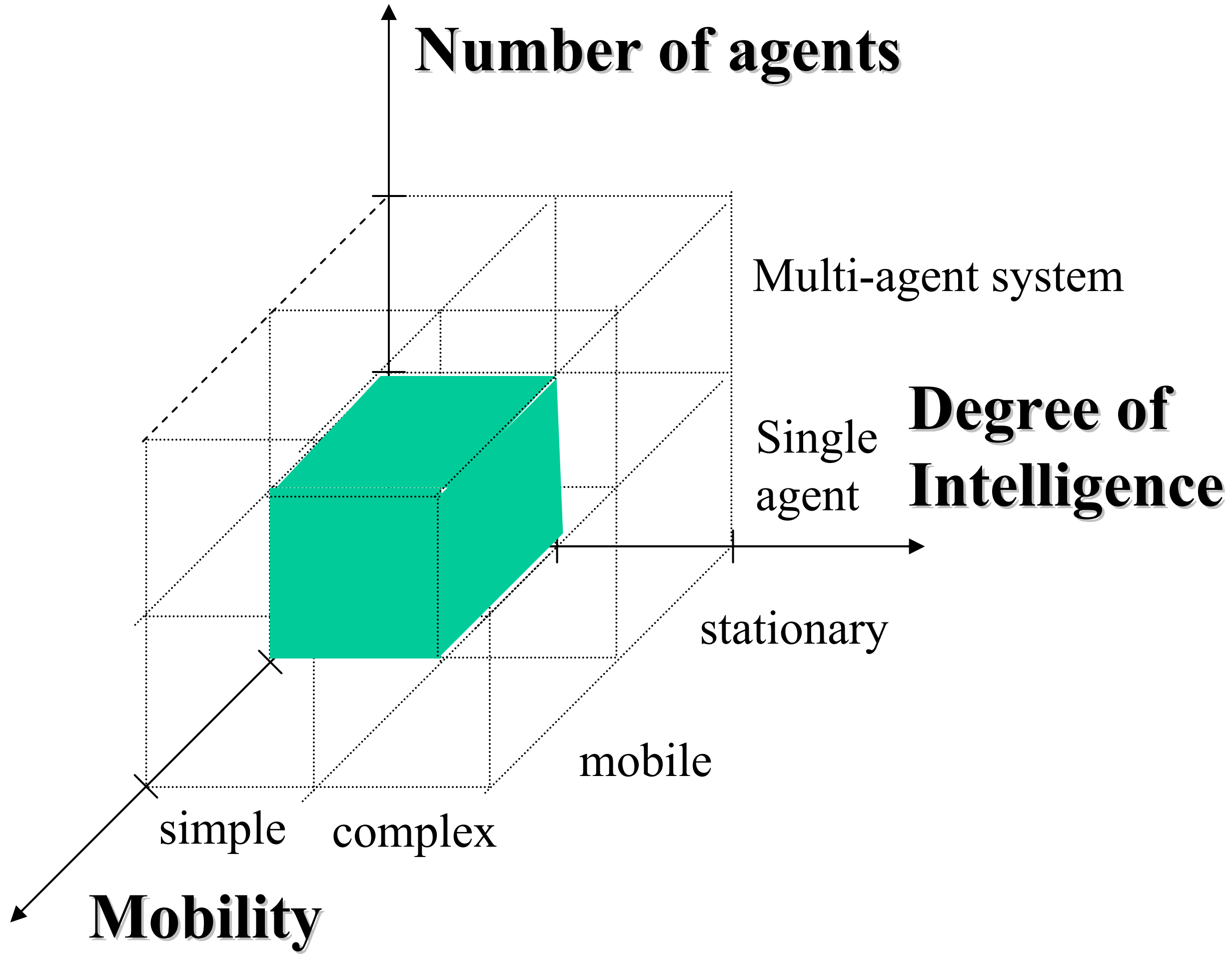
Intelligent Agents' Characteristics



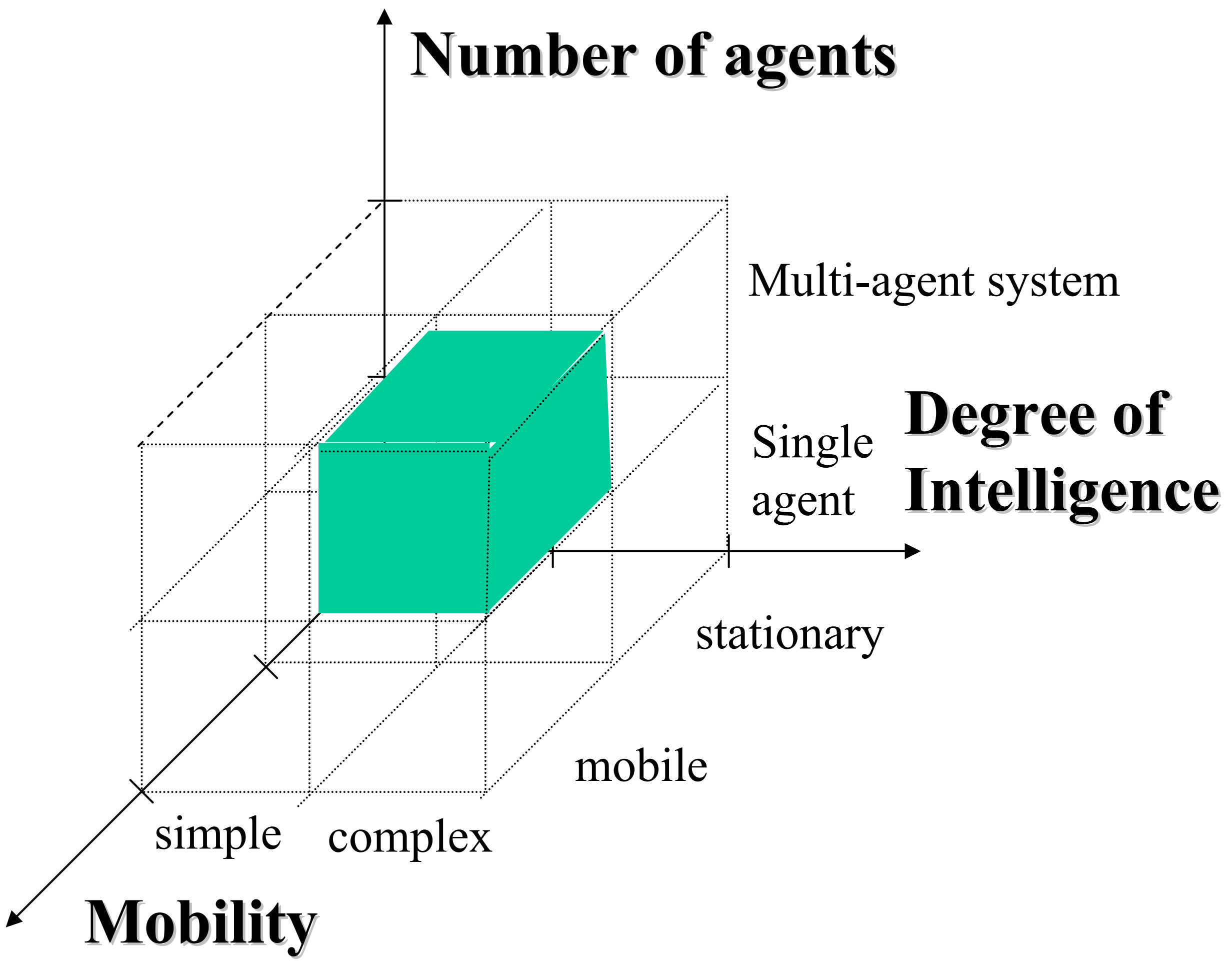
Classification Matrix



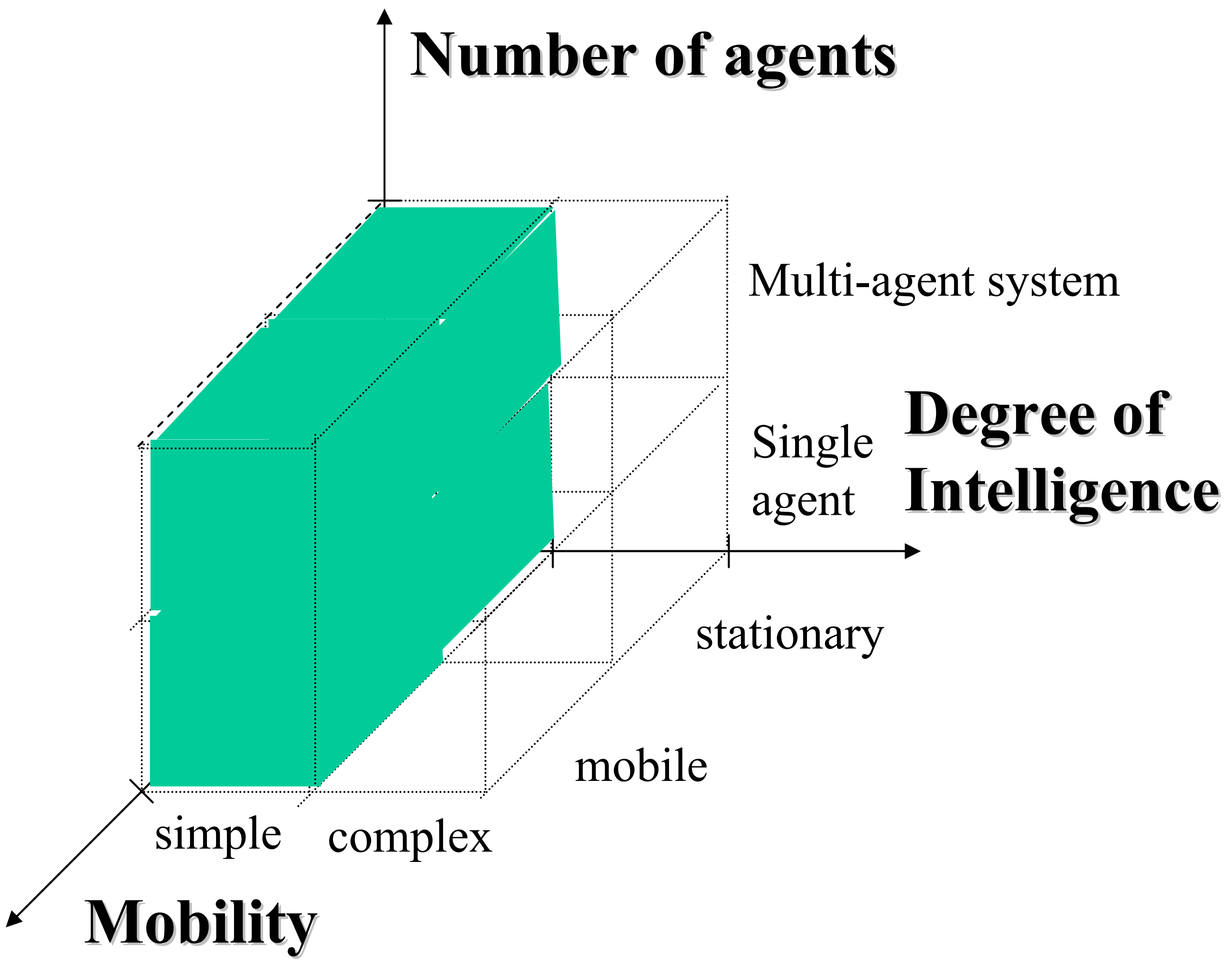
Information Agent



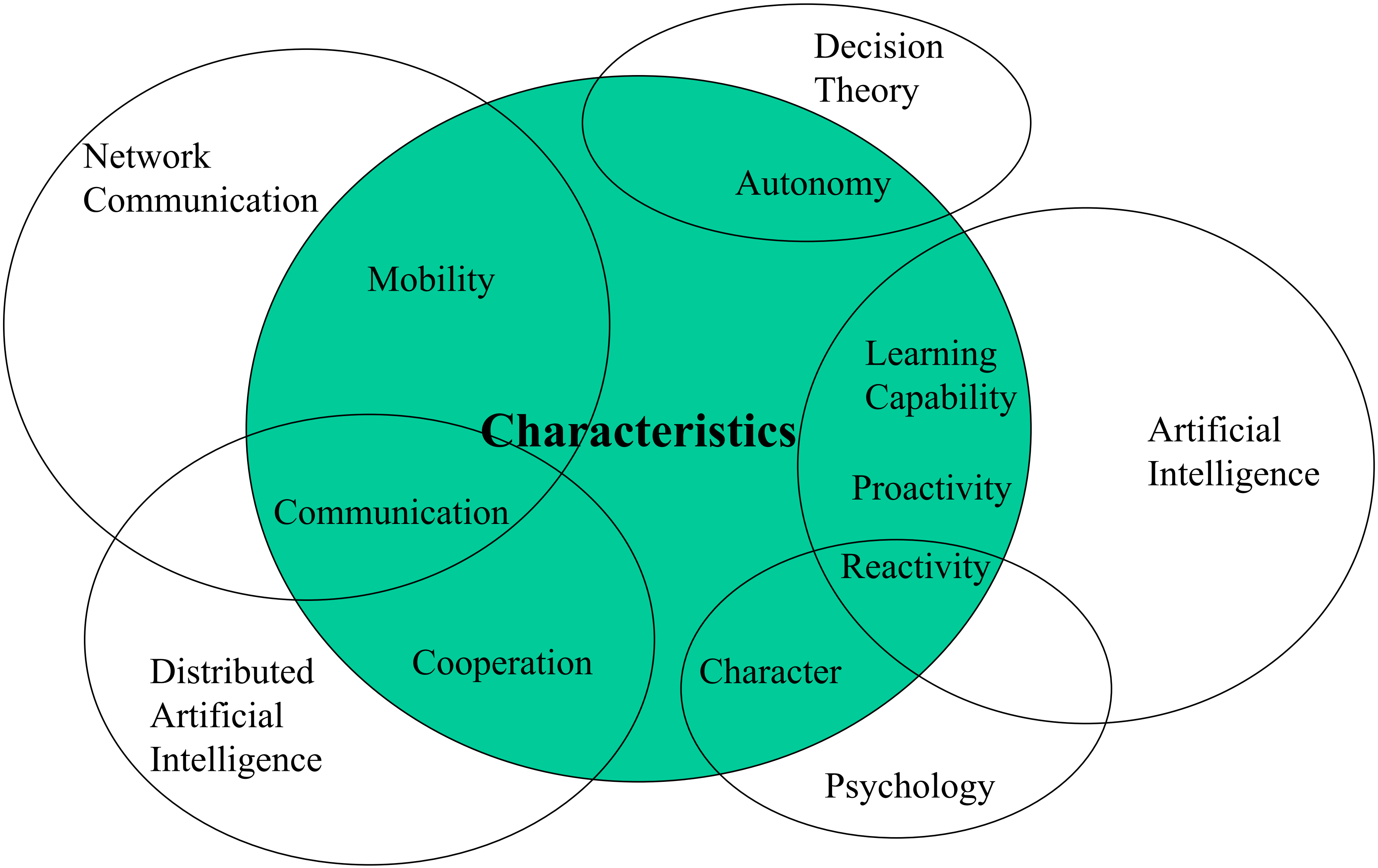
Cooperation Agent



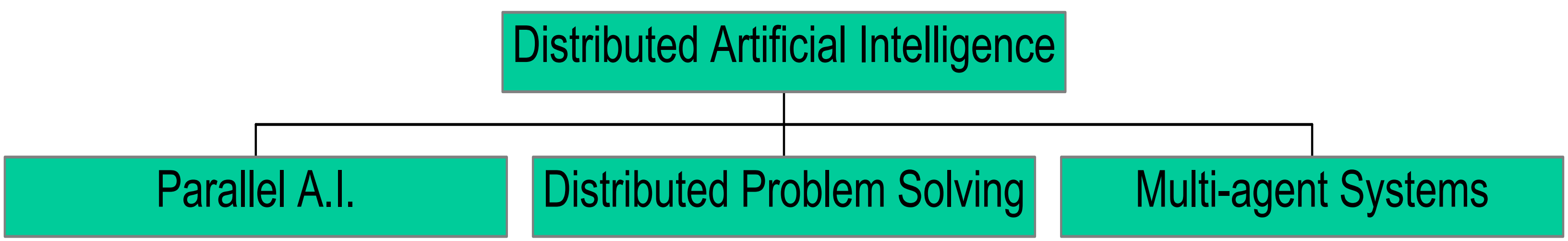
Transaction Agent



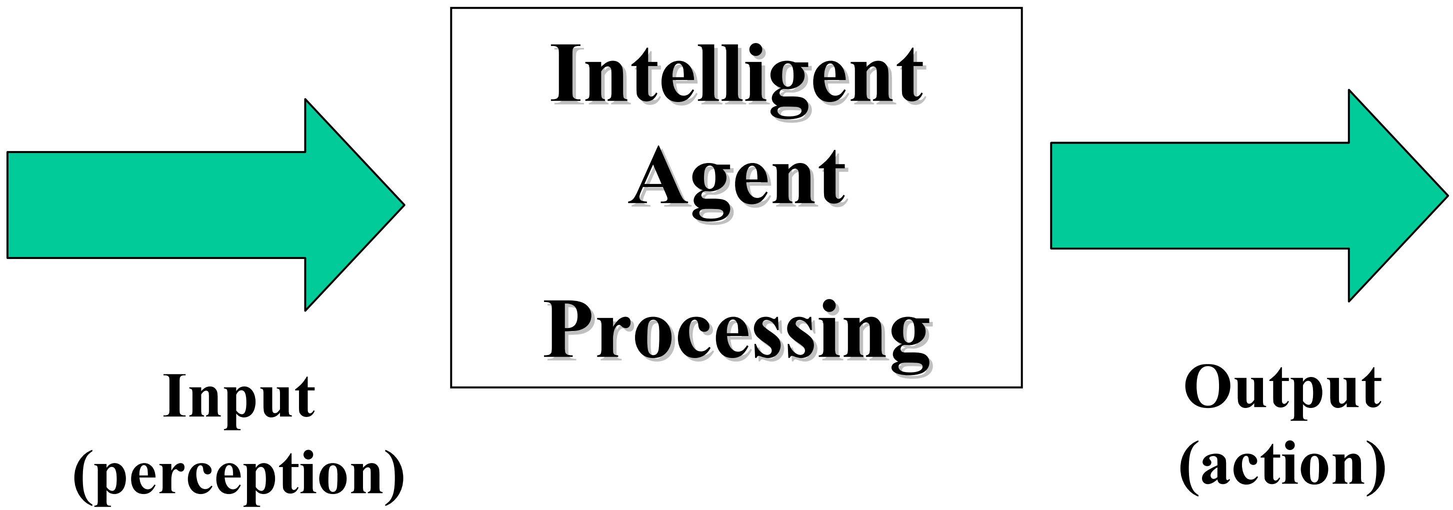
Areas of Influence



Subareas of D.A.I.

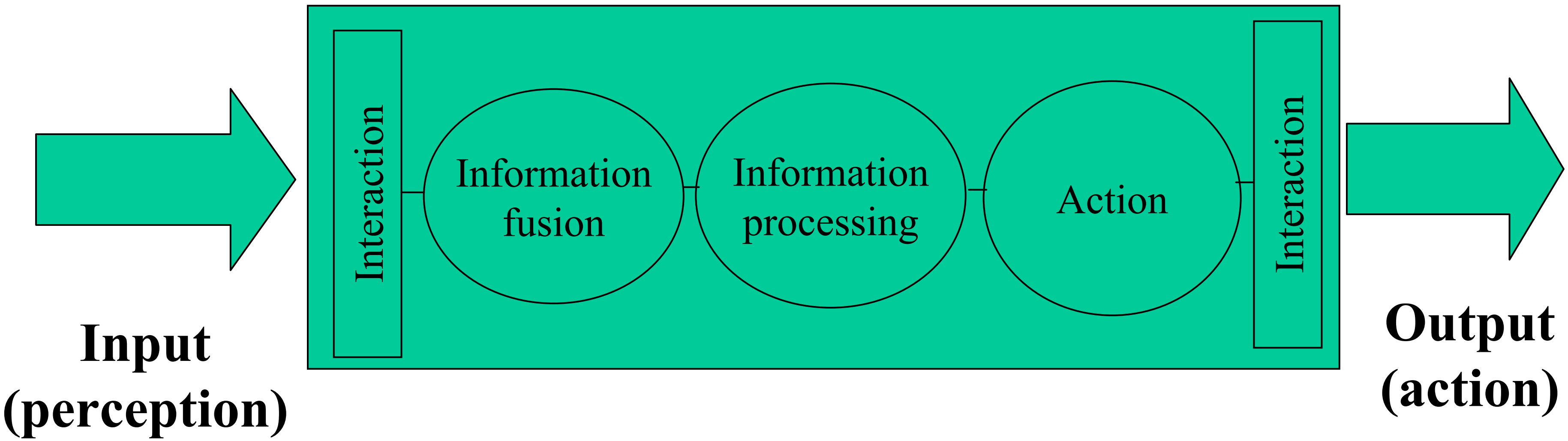


Agent as a black box

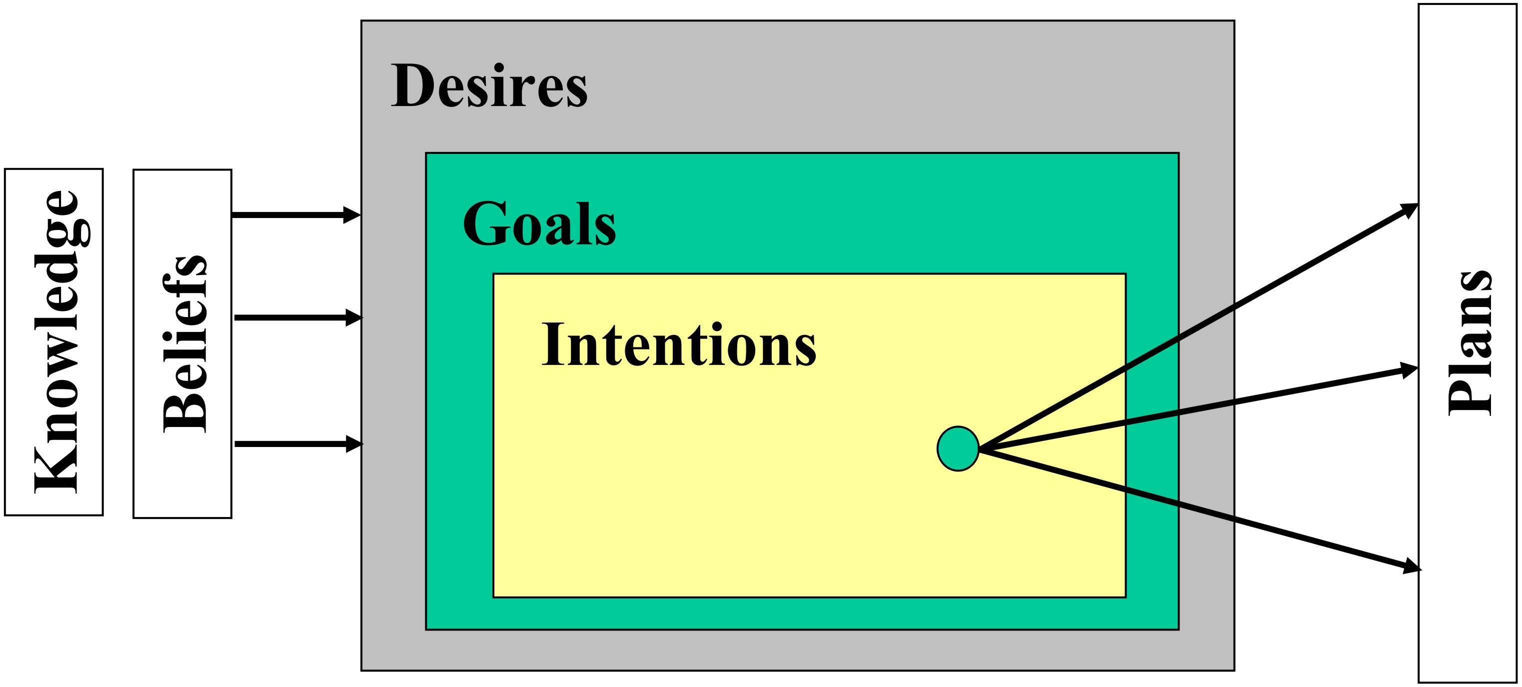


Reactive vs Deliberative

The work of an Intelligent Agent

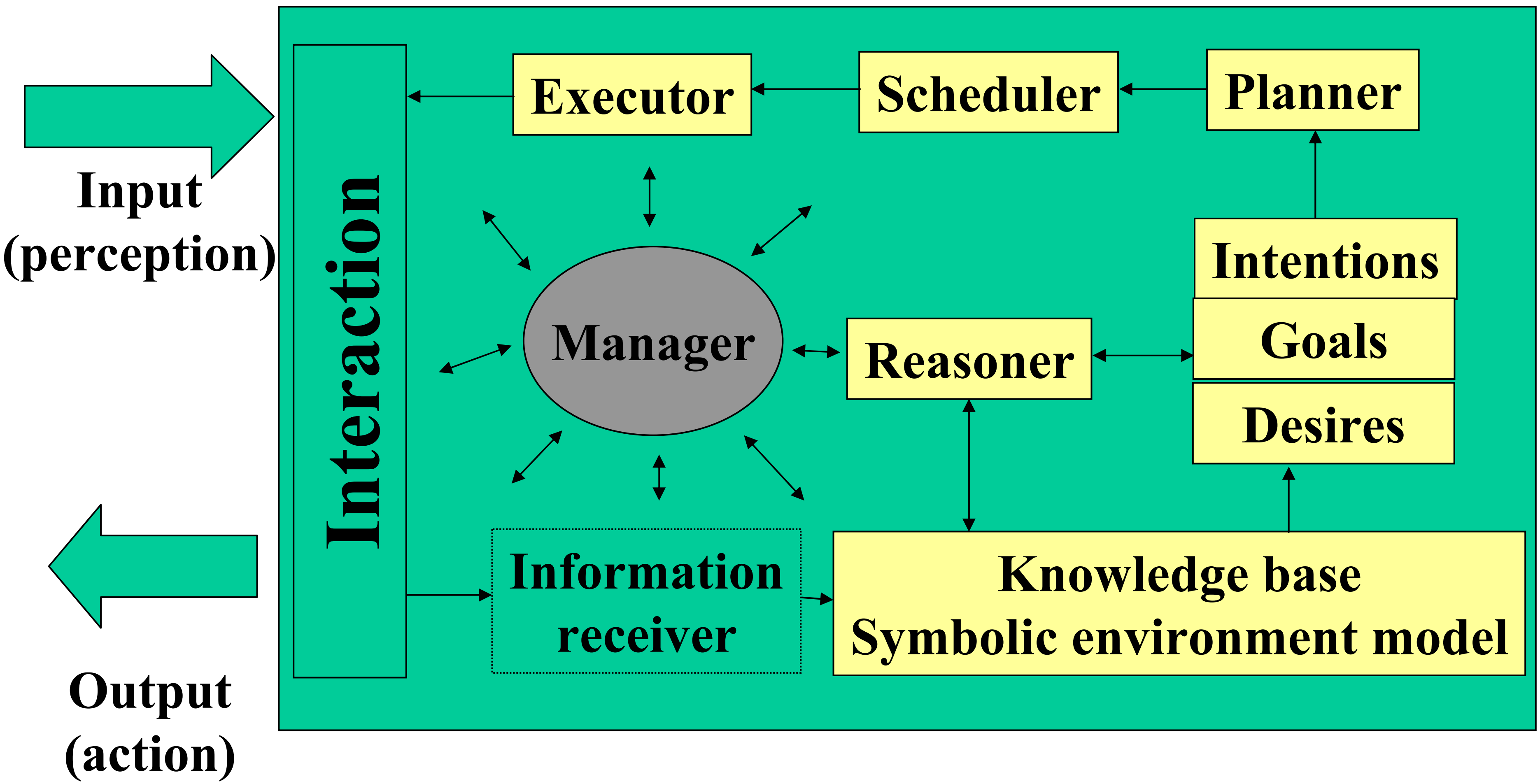


BDI Architecture

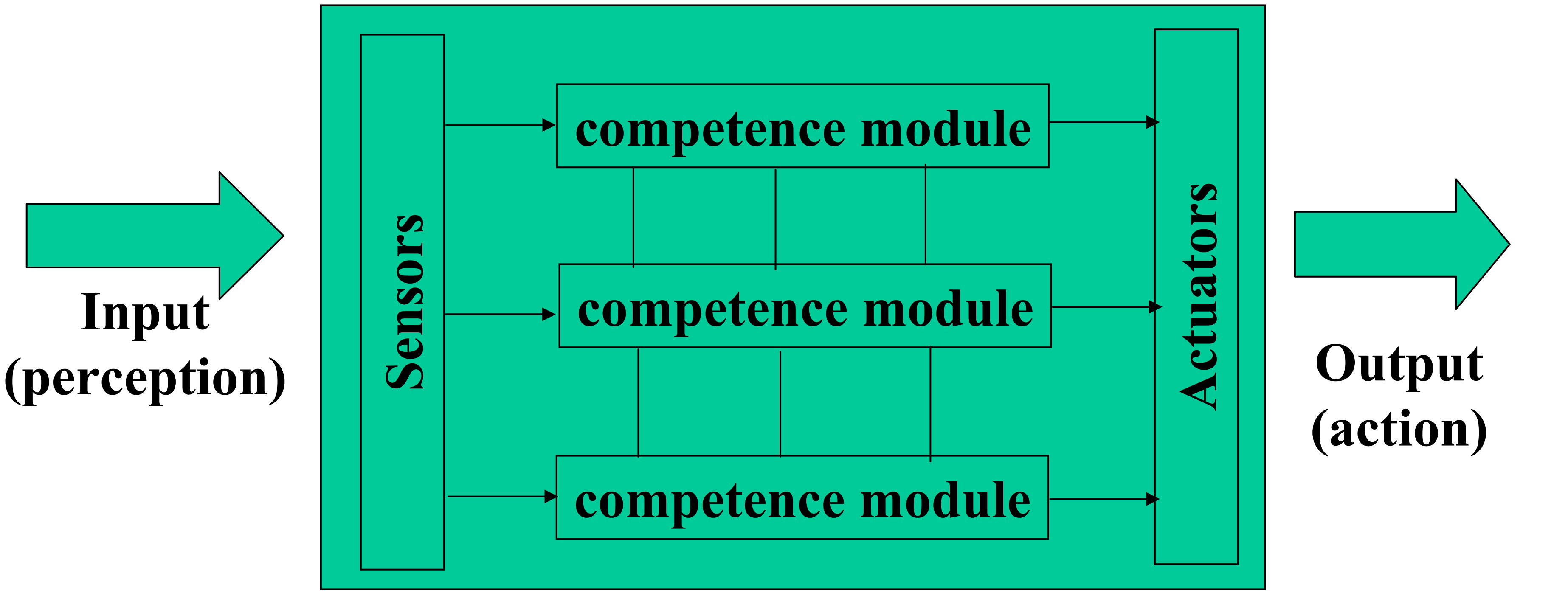


Rao/Georgeff ‘95

Architecture of deliberative agents



Architecture of reactive agents



Brooks '86

Existing Agent Architectures

	Existing System Architecture
Deliberative Agents	GRATE (Jennings), BDI (Rao, Georgeff), MECCA (Steiner et al)
Reactive Agents	**Subsumption** (Brooks), Pengi (Agre, Chapman), Dynamic Action Section (Maes), SynthECA (White)
Hybrid Agents	RAP (Firby), **Interrap** (Muller), AIS (Hayes-Roth), TouringMachine (Ferguson)

BDI

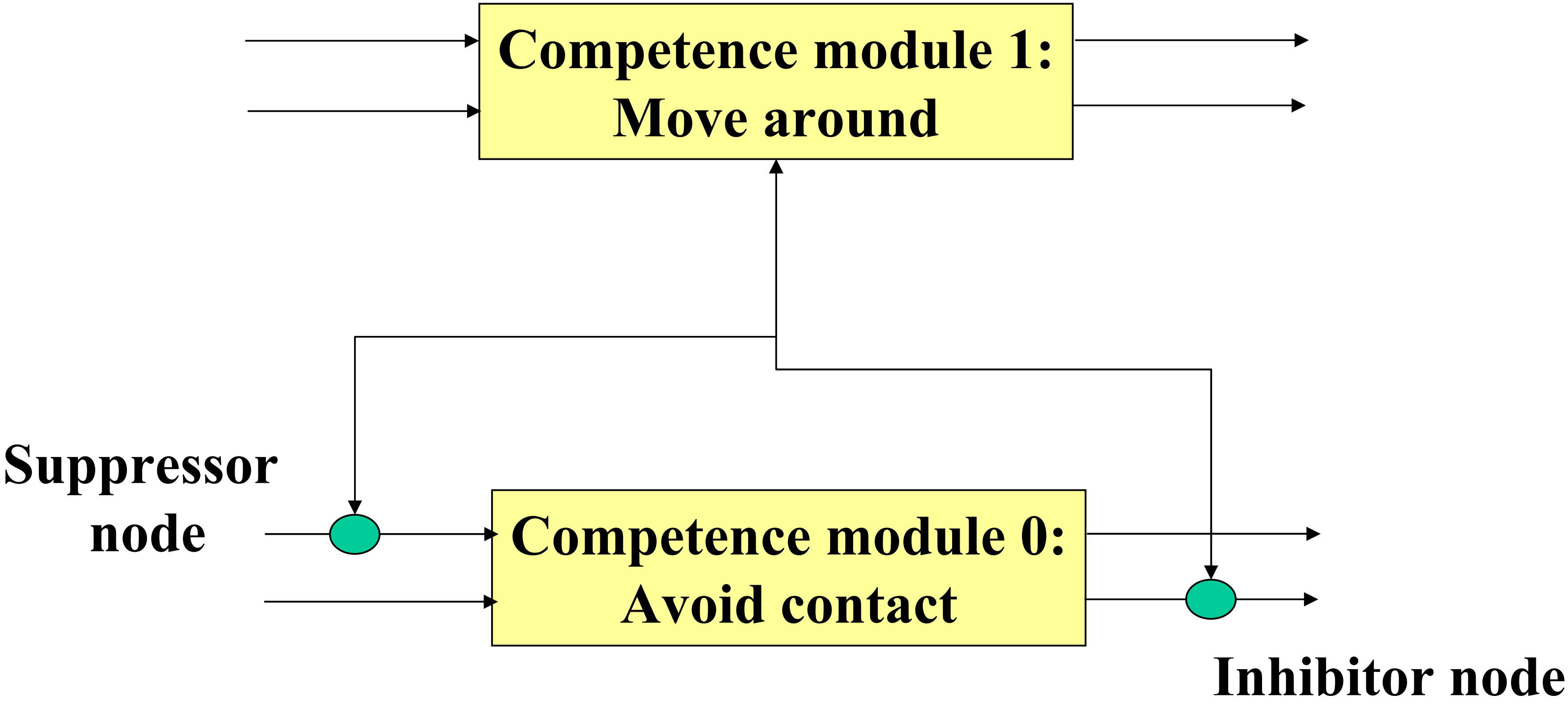
```
Initialize-state();
repeat
        options=option-generator(event-queue);
        selected-options=deliberate(options);
        update-intentions(selected-options);
        execute();
        get-new-external-events();
        drop-successful-intentions();
        drop-impossible-intentions();
end repeat
```

BDI has formal logic, partially implemented in algorithm, dMars, PRS also BDI implementations.

Subsumption

- Brooks ‘86, Hayzelden ‘98, White ‘98
- No explicit knowledge (“connectionist”)
- Distributed behaviour architecture
- Intelligence is “emergent”
- No reasoner, planner or centralized “manager”
- pure activity-oriented task division rather than functional decomposition.

Suppressor and Inhibitor Nodes



[Subsumption]()

Suppressor node: modifies input signal for period of time
Inhibitor node: inhibit output for period of time

Spreading Activation Model

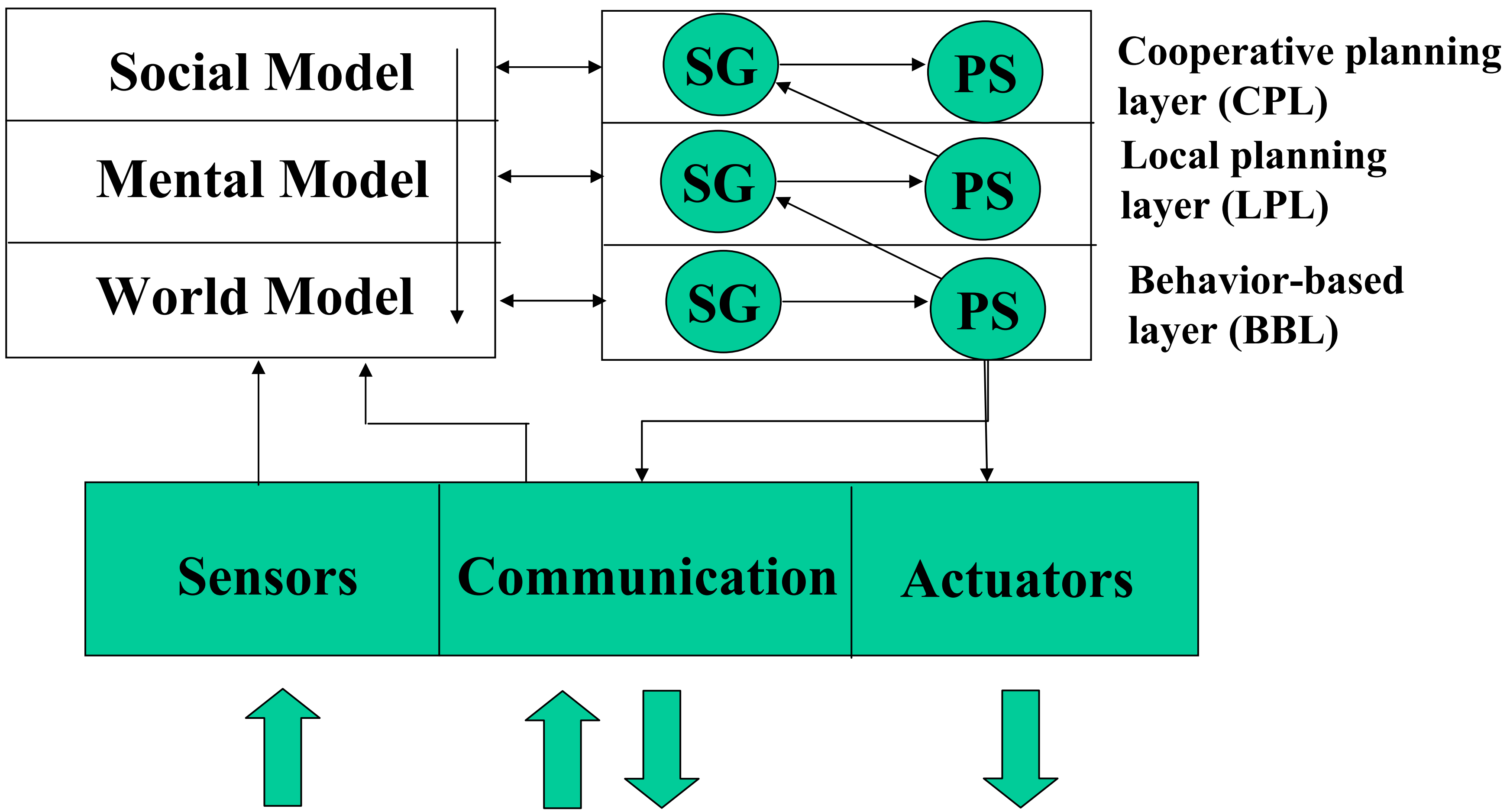
Mathematical Model

Reactive Systems

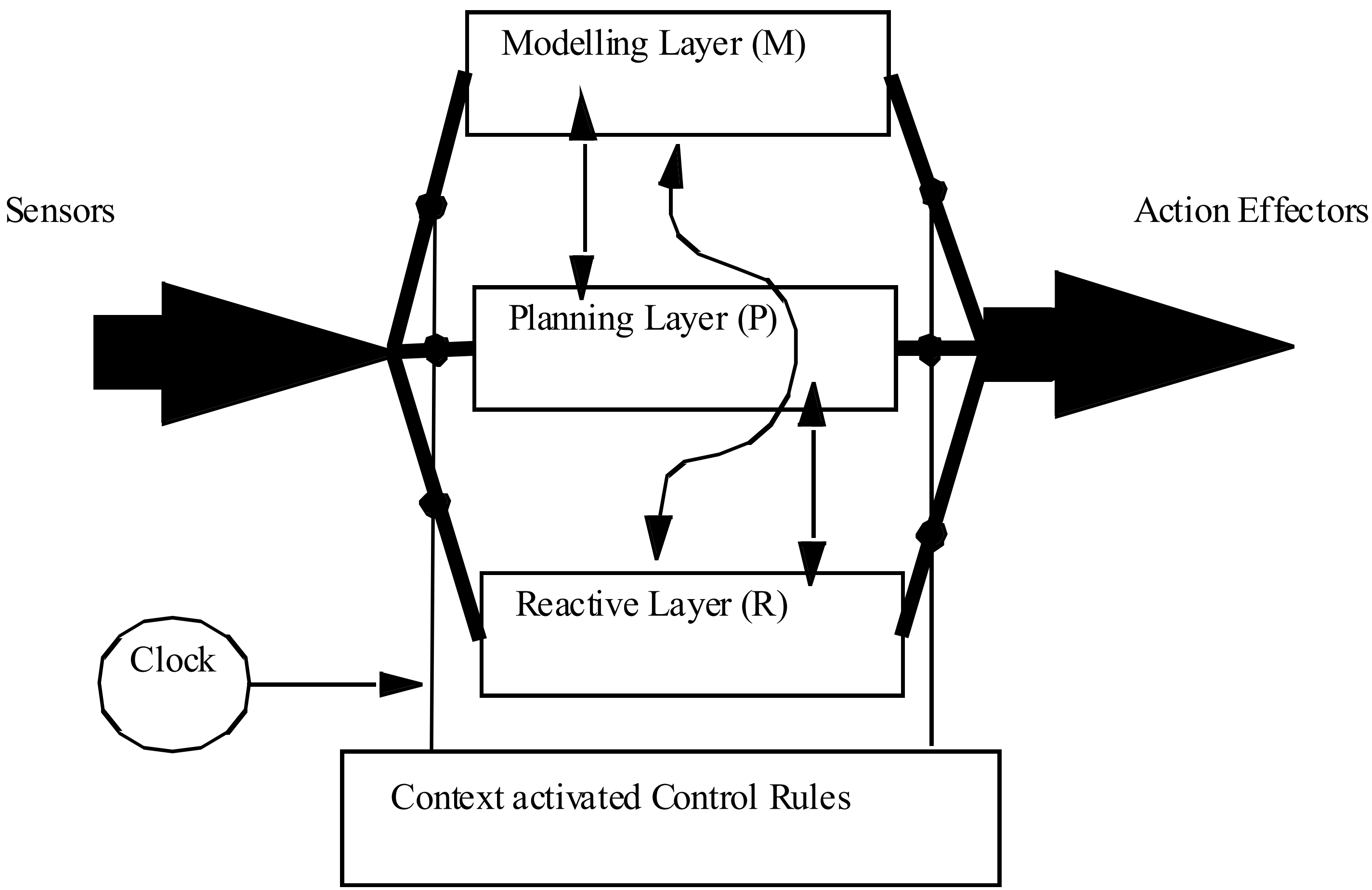
Pengi explained...

More Pengi

Interrap Hybrid Architecture

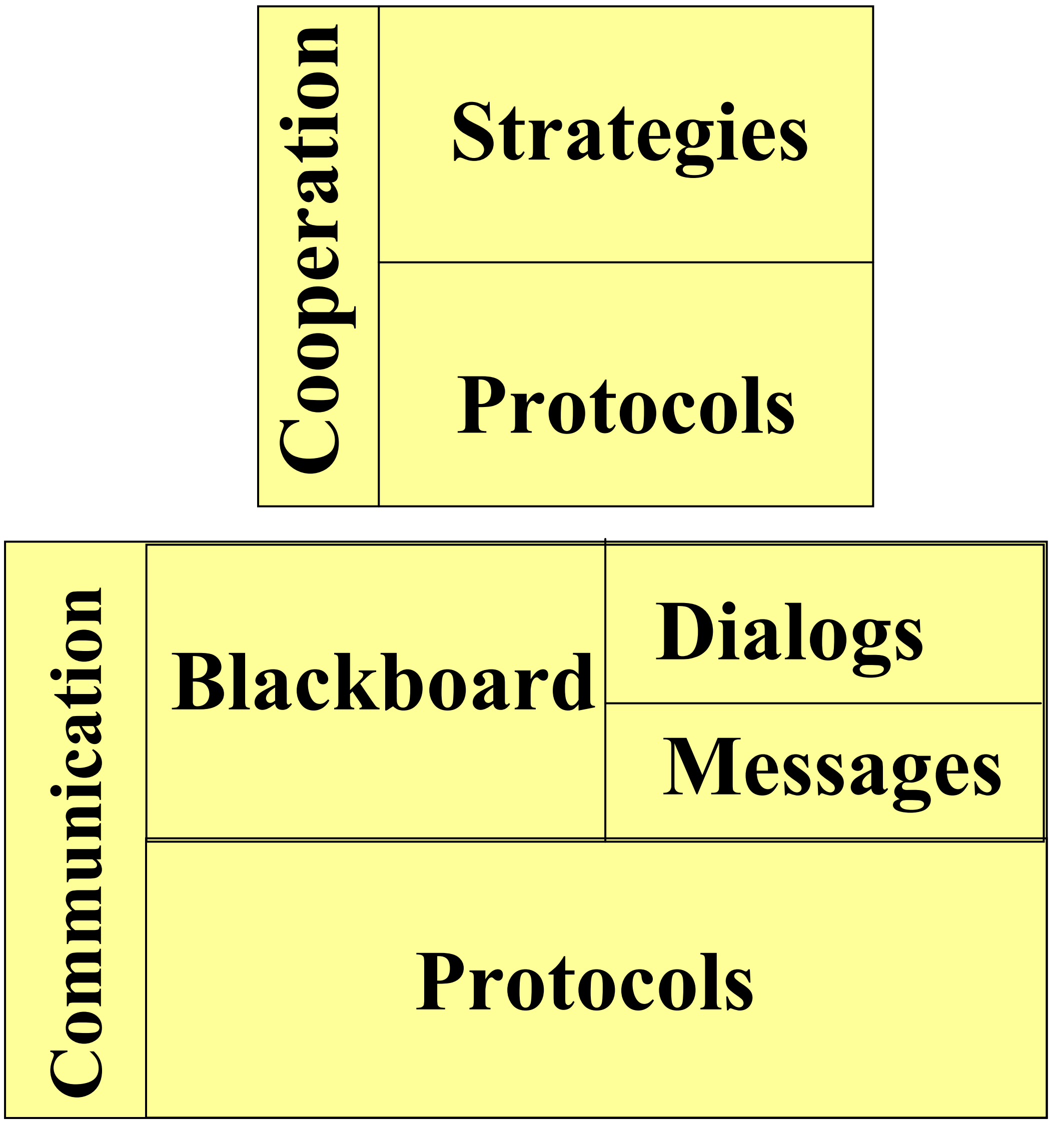


Touring Architecture

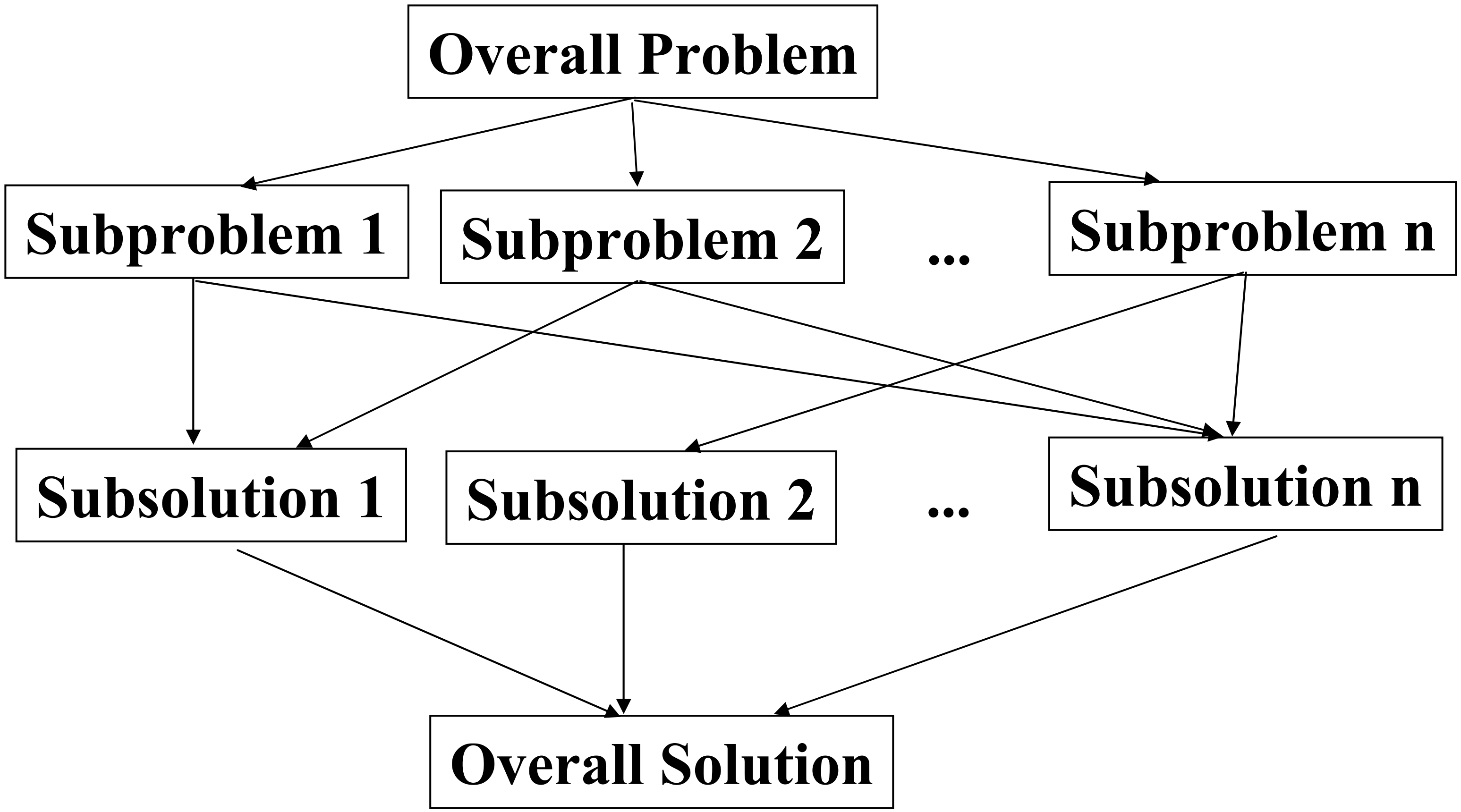


Layer connectivity in TouringMachines

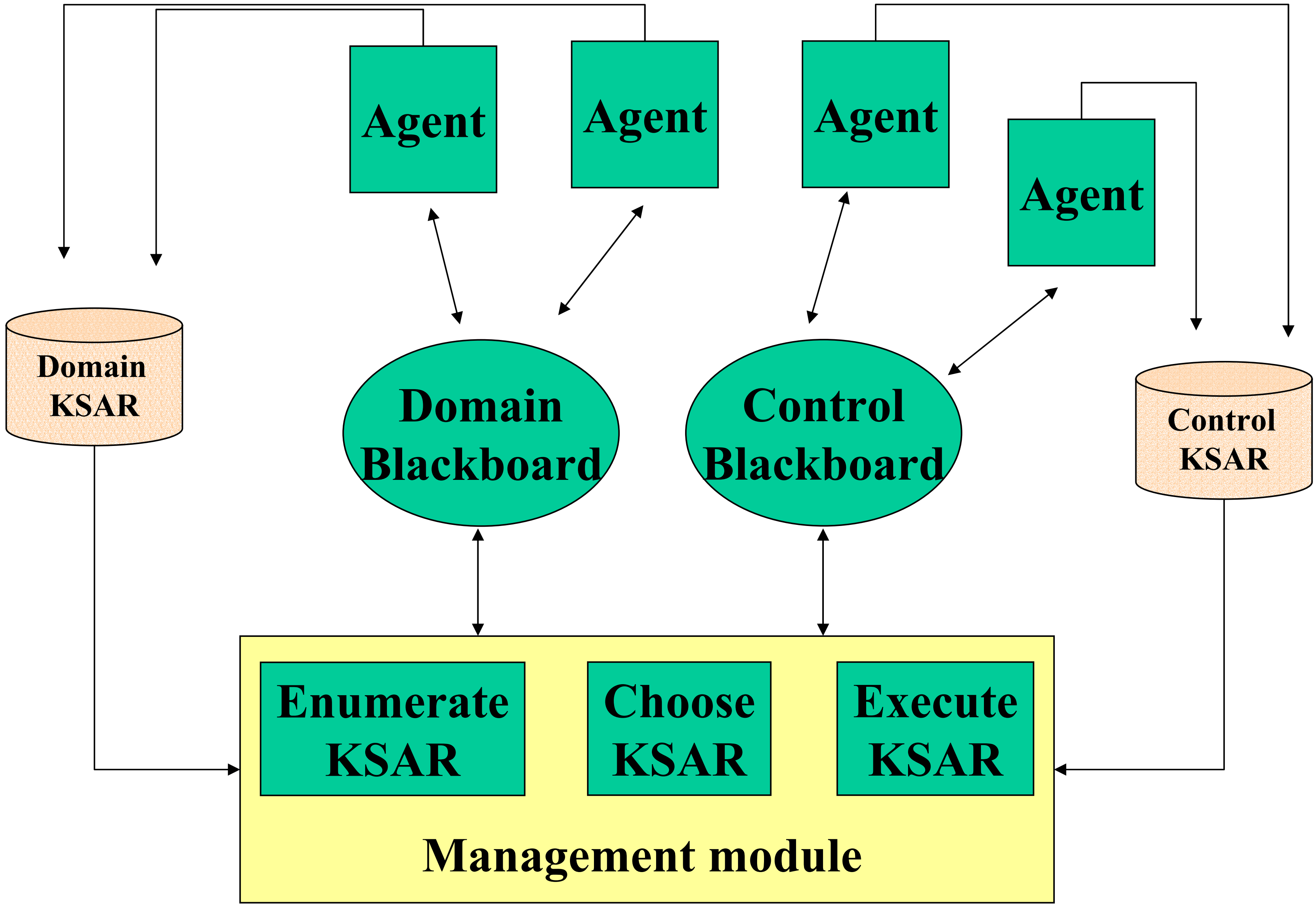
Communication and Cooperation



Distributed Problem Solving



Blackboard



Agent
Agent
Agent
Agent
Domain KSAR
Domain Blackboard
Control Blackboard
Control KSAR
Enumerate KSAR
Choose KSAR
Execute KSAR
Management module

Messages

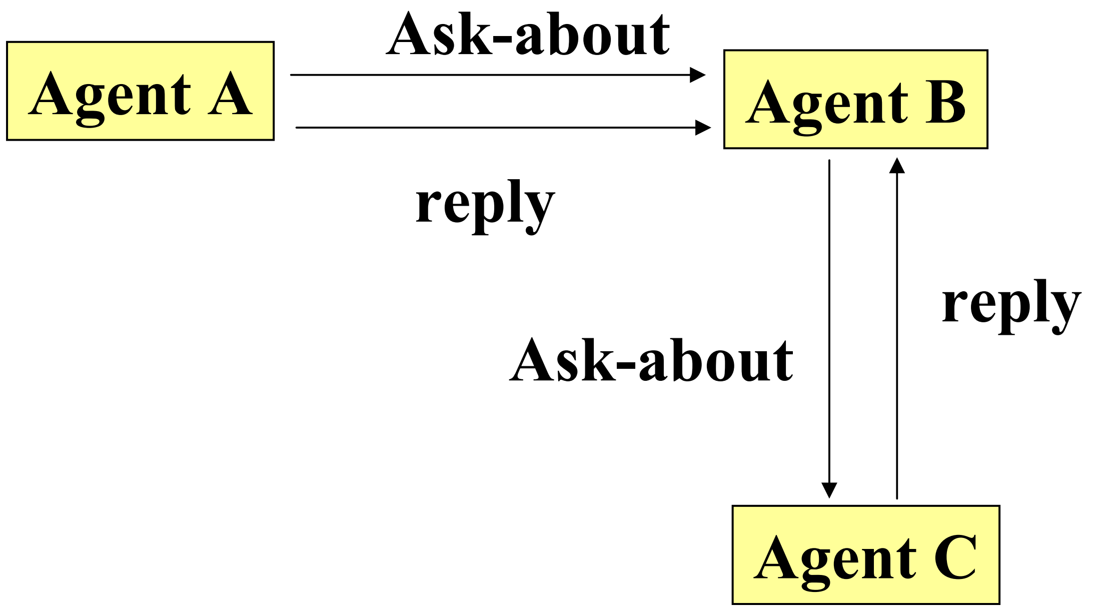
- Messages based upon ‘speech acts’ [Austin, 62]
- A speech act designates a message that contains not only a true/false statement but also exercises a direct influence on the environment by causing changes within the environment.

Can you give me certain information?

Knowledge Query and Manipulation Language

- KQML based upon speech act theory
  - result of American Knowledge Sharing Effort (KSE) [Finin ‘93].
- KQML differentiates between three layers: communication, messages and content
  - communication: protocol
  - messages: speech acts
  - content: content or meaning of message
- KQML deals with speech acts.

Dialog



Dialog: a sequence of agent message interactions with some common thread.

KQML format

(<Performative>
:content <statement/speechact>
:sender <name>
:receive <name>
:language <text>
:ontology <text>
)

Performative corresponds to speech act types.

Important KQML speech act types

Speech act type	Meaning
achieve	S wants E to make true some statement in his environment
advertise	S is particularly suitable to perform some particular speech act type
ask-all	S wants all answers in E's knowledge base
ask-one	S wants an answer in E's knowledge base
broker-one	S wants E to find help for answering of his speech act
deny	The speech act no longer applies for S
delete	S wants E to remove specific facts from his knowledge base.

Important KQML speech act types

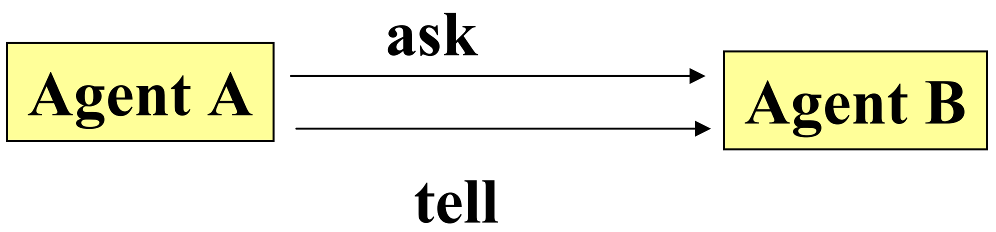
Speech act type	Meaning
recommend-one	S wants the name of an agent that can answer a speech act
recruit-on	S wants E to request an agent to perform a speech act
sorry	S does not possess the required knowledge or information
subscribe	S wants continuously information of E's answers for a speech act
tell	S transfers an information item.

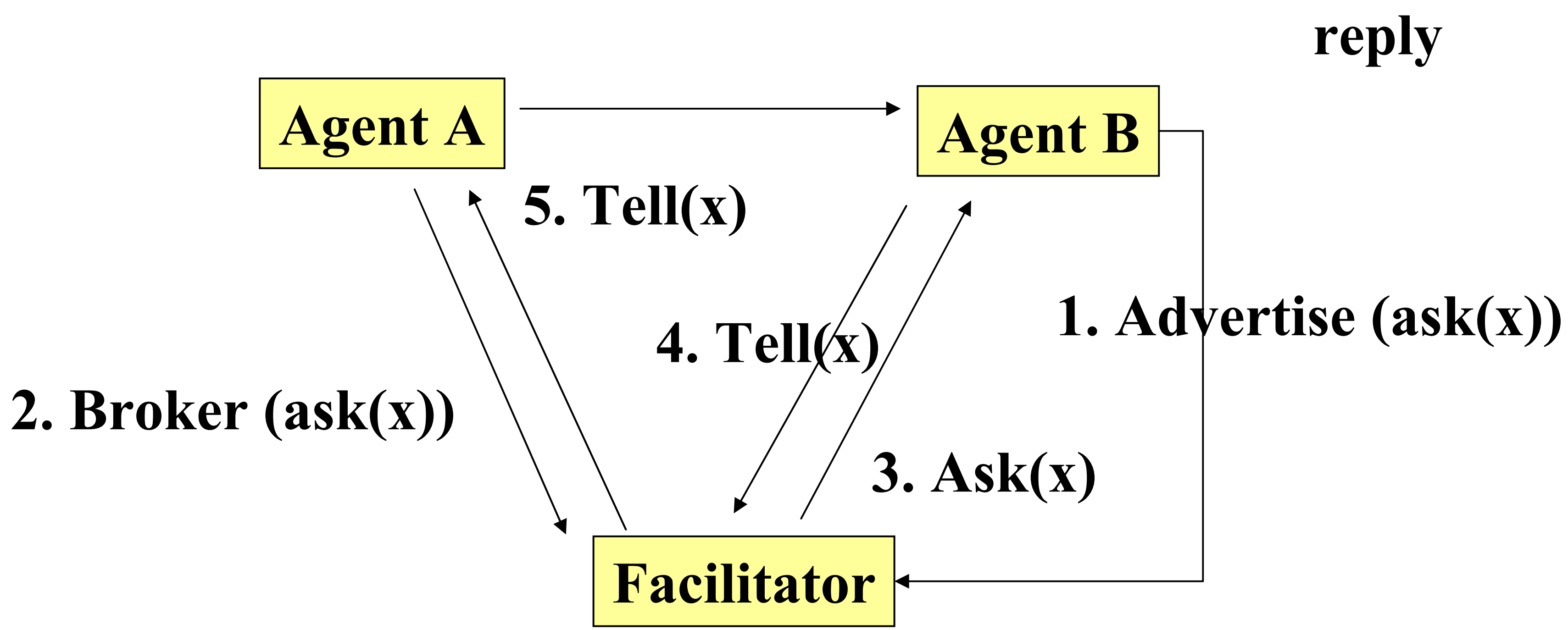
Example

```
(ask-one
:content (PRICE IBM ?price)
:receiver stock-server
:language LPROLOG
:ontology NYSE-TICKS
)
```

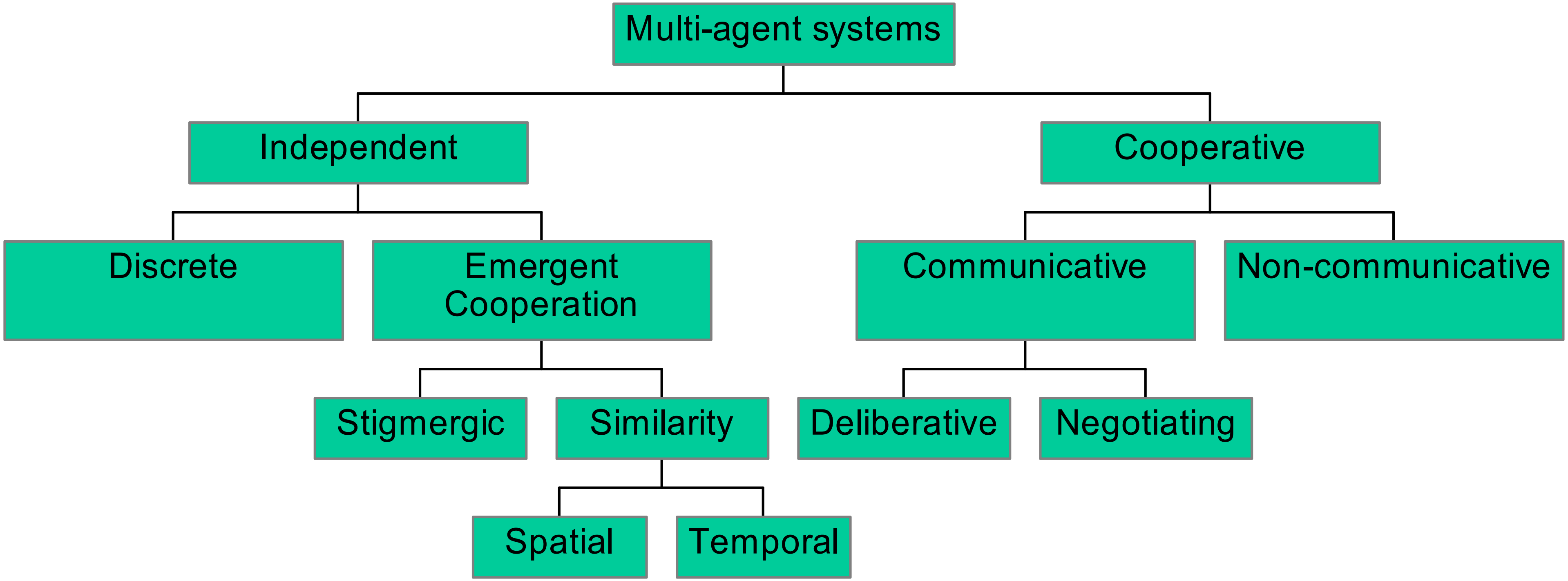
Query formulated using LPROLOG.
Ontology is ‘computer systems’.

Using a Faciltator





Cooperation typology



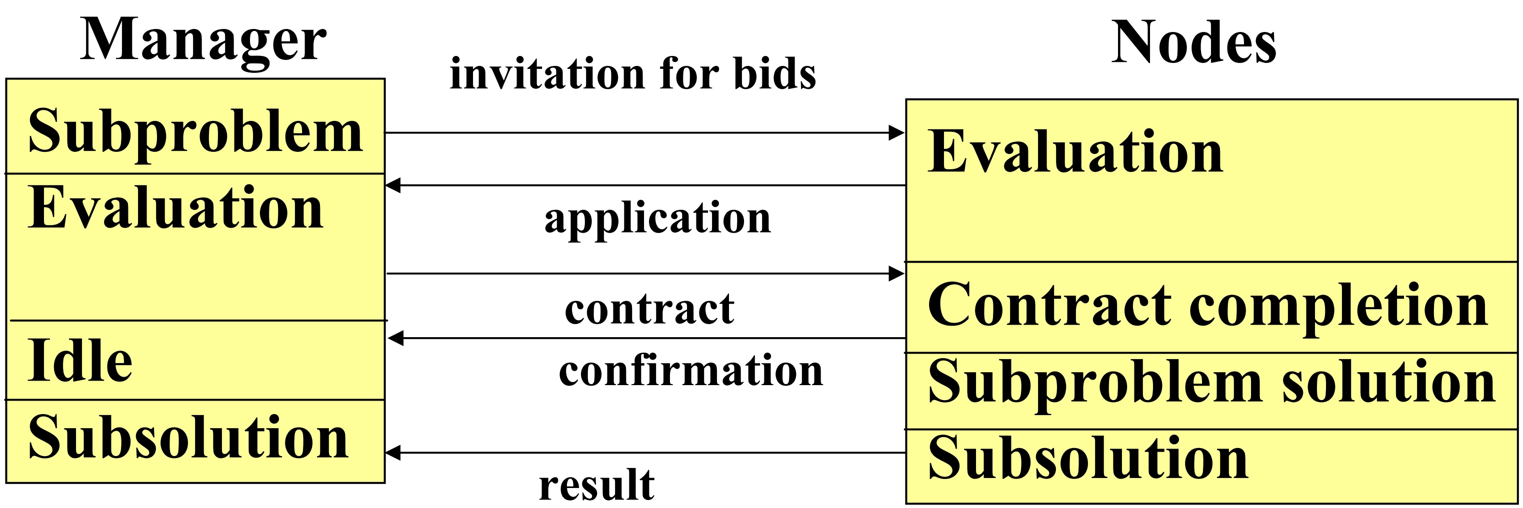
Doran et al '97

Contract-Net Protocol

- Desire for efficient coordination in multi-agent systems.
- Modelled after ‘free market economy’.
  - Subtasks are openly offered as bids
  - nodes reply, if interested
- Requires a commonly understood inter-node language [Smith, 80].
  - Common message format.

Contract Net Systems

- Contract net system engaged after problem division phase.
- Manager node undertakes the assignment of subproblems via the contract net protocol.



Example Protocol

1

TO: all nodes

FROM: manager

TYPE: task bid announcement

ContractID: xx-yy-zz

Task Abstraction:

<subproblem description>

Eligability Specification:

<list of minimum requirements>

Bid specification: <description of required application information>

Expiration time: <latest possible application time>

2

TO: manager

FROM: node X

TYPE: application

ContractID: xx-yy-zz

Node Abstraction: <description of the node's capabilities>

3

TO: node X

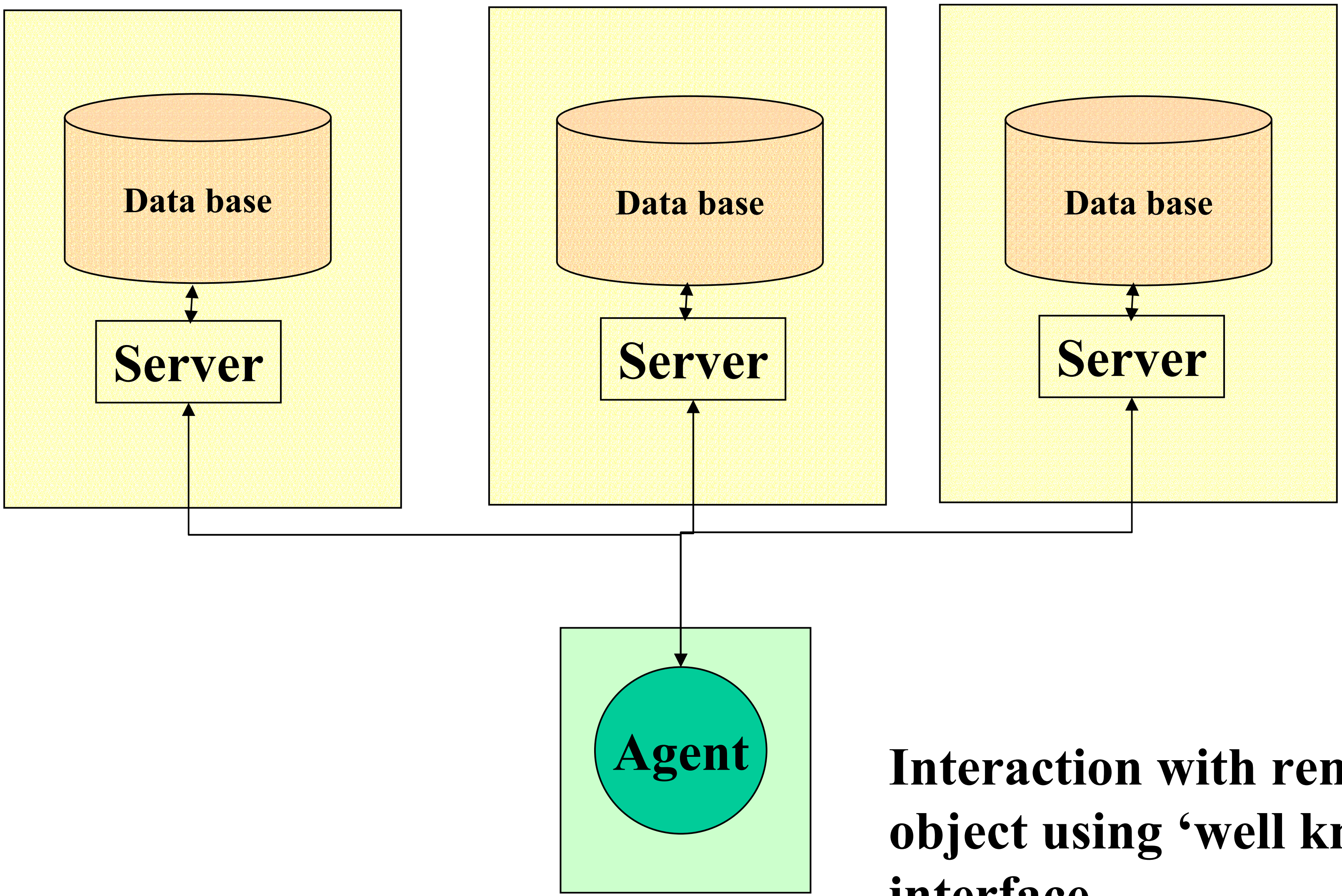
FROM: manager

TYPE: contract

ContractID: xx-yy-zz

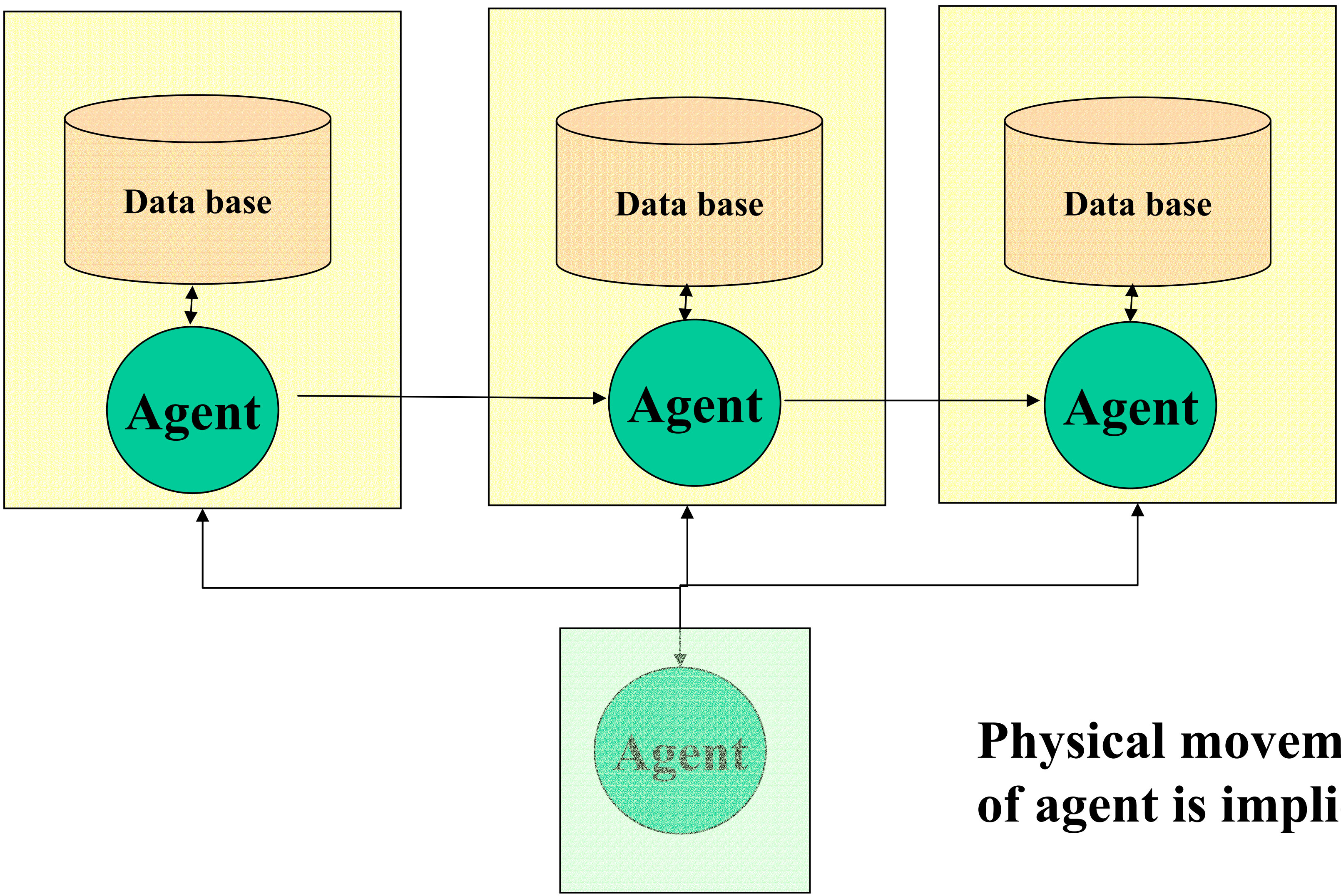
Task Specification: <description of the subproblem>

Mobility: Remote Procedure Call



Interaction with remote object using 'well known' interface

Mobility: Remote Programming



Physical movement of agent is implied

RP versus RPC

Communication	Properties	Agents
Remote Programming	High intelligence, flexible	mobile
Remote procedure call	low intelligence, proprietary	stationary

Collaboration

- Division of work amongst many agents of the same type in achieving goal