Introduction Agents and Multi-Agent Systems

Autonomous Software Agents

A.A. 2022-2023

Prof. Paolo Giorgini

Dr. Marco Robol

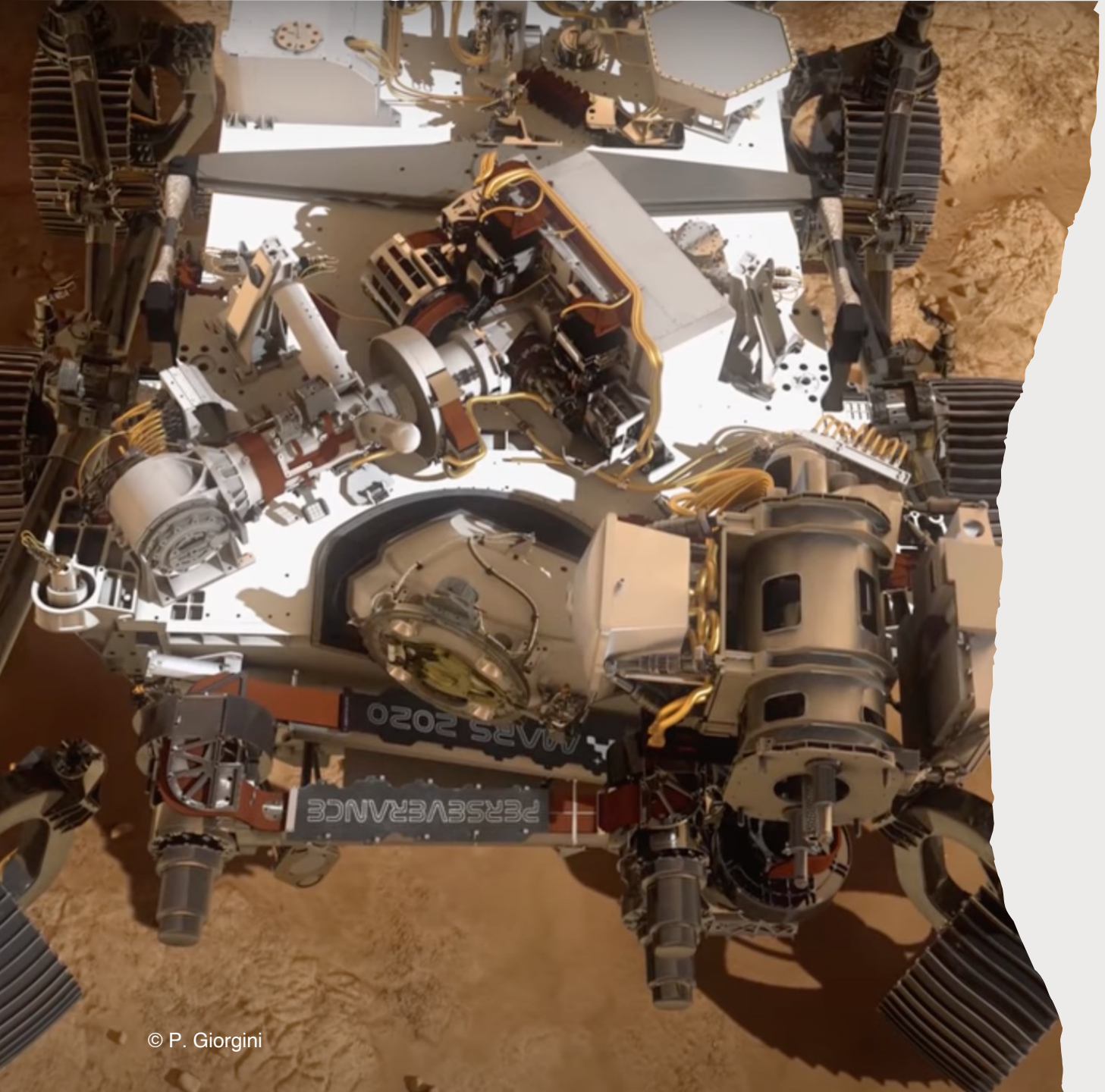
UNIVERSITY OF TRENTO - Italy

Department of Information and Communication Technology

What is a (software) agent?

New perspective

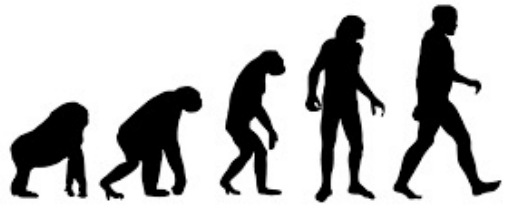
- Computers are not very good at knowing what to do
  - Every action a computer performs must be explicitly anticipated, planned for, and coded by a programmer
- They fail when they encounter a situation that their designers did not anticipate
  - they crash and may cause system crash and loss of life at worst
- Growing interest in developing software that may decide for themselves what they need to do in order to achieve their objectives
  - able to change their behavior in rapidly changing, unpredictable, or open environments



Examples

- Space probe making its long flight to outer planets
  - Ground crew is required to continually track its progress and decide to deal with unexpected eventualities
  - Very costly and if decisions are required quickly, it is simply not practicable
  - NASA and ESA are interested in the possibility of making probes more autonomous (richer onboard decision-making capabilities and responsibilities)

Human Orientation and delegation



- Software systems as human models
  - Programmers tend to conceptualize and implement software in terms of higher-level – more human-oriented – abstractions
    - The object abstraction is not enough to model a piece of software
- Delegation and control
  - Unpredictable behavior
  - Emergent and collective behavior
- Socio-technical systems
  - Technical components and human actors are part of the actual system and its design – no conceptual distinction between them
  - Strong interdependency between human actors and software components

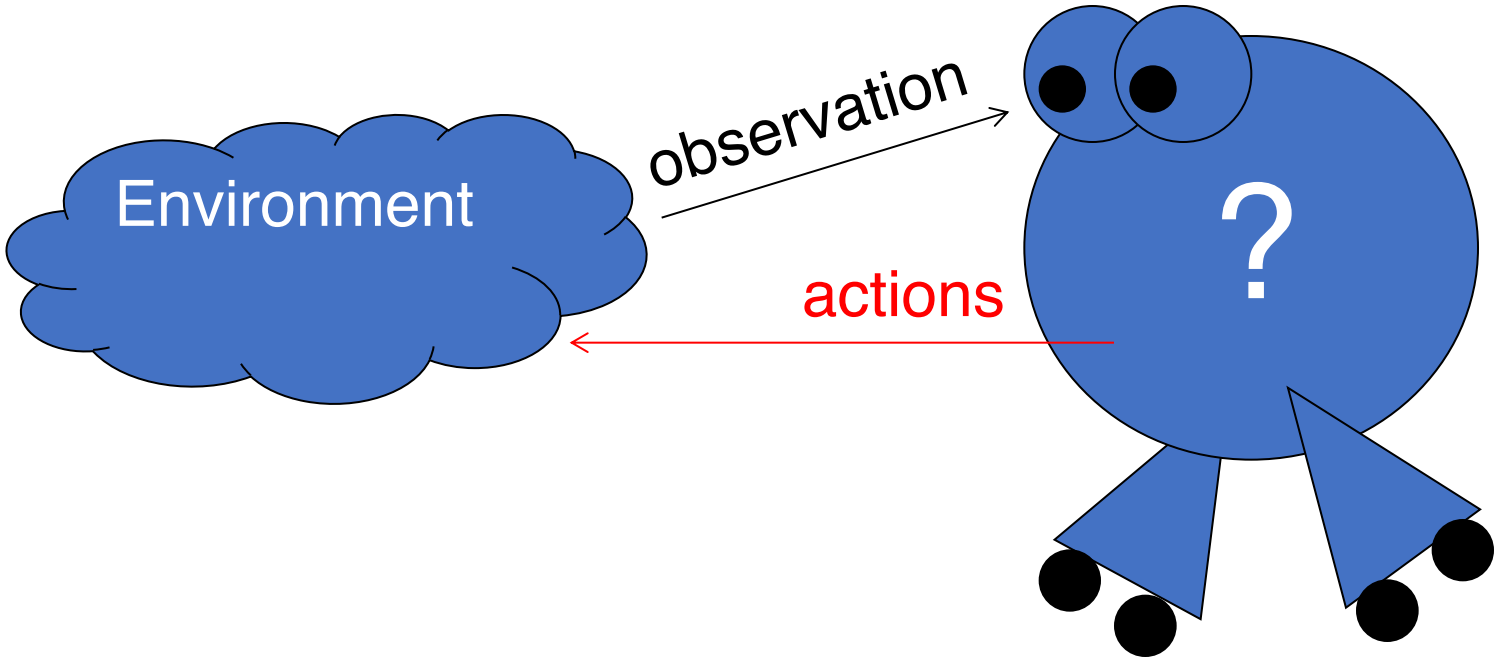
Development is getting harder

- Very ambitious requirements
  - E.g., Intelligent and human-like software
- Less certain requirements
  - Greater scope for change
  - More interdependent requirements
  - Socio and Organizational constraints
- More challenging environments
  - Greater dynamism
  - Greater openness
- Lack of appropriate design models

Agent? What's?

- Very controversial definitions for the agent concept
  - *"…encapsulated computer system, situated in some environment, and capable of flexible autonomous action in that environment in order to meet its design objectives"* [Wooldridge]
    - autonomous: control over internal state and over own behaviour
    - situated: experiences environment through sensors and acts through effectors
    - flexible
      - reactive: respond in timely fashion to environmental change
      - proactive: act in anticipation of future goals

An agent its environment



An example of agent

(neither autonomous or intelligent)

- Any control system can be viewed as an agent
  - Thermostat has a sensor for detecting the room temperature
  - Temperature is too low or OK (two possible values)
  - The thermostat can perform `heating_on` or `heating_off`
  - Of course, `heating_on` action cannot guarantee to increase the temperature (partial control)

```
too cold → heating_on
temperature OK → heating_off
```

- What about if windows are open?
  - I should know in advance that there are windows in the room (or doors) that if open can decrease the temperature

```
if(temperature==too_cold){
    heating_on()
} else {                    // temperature_OK
    heating_off()
}
```

  - But, what if we don't know about that in adance? Can the software learn it?

```
if(temperature==too_cold && windows_closed){
    heating_on()
} else {
    heating_off()
}
```

??? →

```
if(cond){
    heating_on()
} else {
    heating_off()
}
```

Making it a bit more complex

- Thermostat + Automated Windows

Too cold → heating_on
Temperature OK → heating_off

Stale air → open_windows
Air ok → close_windows

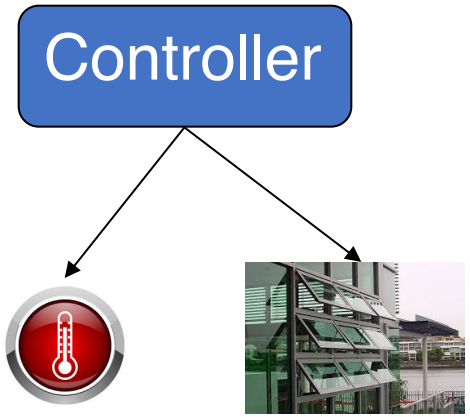
```
// Thermostat agent
if(temperature==too_cold){
    heating_on()
} else {
    heating_off()
}

// Windows agent
if(Stale_air) {
    open_windows()
} else {
    close_windows()
}
```

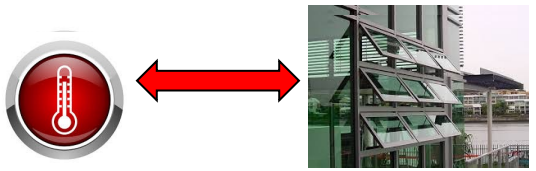
- Does it make sense to keep the heater on when windows are open?
  - Thermostat and Windows should be coordinated

Solution?

```
too_cold ^ windows_closed → heating_on
Temperature_OK → heating_off
```

How does the software will look like?





Consider also this

Can you close the window?

why she is asking me so?

window_closed → temp_ok
window_closed → noise_ok

may she want to increase the temperature or there is too much noise in the room?

Abduction

she wants to make true temp_ok (this means NOT temp_ok) or she wants to make true noise_ok (this means NOT noise_ok)

Consider also this

Can you close the window?

why is she asking me so?

window_closed → temp_ok
window_closed → noise_ok

may she want to increase the temperature or there is too much noise in the room?

NOT temp_OK → feel_cold
NOT noise_ok → NOT follow_the_lecture
dress_coat → NOT fell_cold

She is dressing the coat (NOT fell_cold) so she is asking because she wants follow_the_lecture

In another form

- Receive a request for an action a
- Find the reason of why of such a request
  - if I Do(a) -> b && c will be true
  - this means !b || !c
  - !b -> d
  - !c -> !e
  - !g -> !d
  - If I know g is true -> she wants to make true e
- So what?
  - If she wants to follow the lecture (e), is there any “better” way to make this true?

a: close the window

b: temperature is ok

c: noise is ok

d: feel cold

e: follow the lecture

g: dress coat

How to develop an agent like that?

- We should provide knowledge to the agent
  - Do(a) -> b && c // expexted consequences of an action
  - If we want b && c it means !b || !c // ----------
  - !b -> d // knowledge abut the world
  - !c -> !e //
  - !g -> !d // ----------
  - g // the current state of the world
- And mechanisms to reasoning about that
  - -> e // producing other knowledge
- Use already known knowledge and produced knowledge to
  - Explore alternative
  - Take a decision

Flexibility

```
if(obstacle) {
    avoid(obstacle)
} else {
    g = revise_goals()
    p = plan(g)
}
```

An agent can act reactivly as well as proactively

- Reactive means that the agent reacts in reasonable time and in an appropriate way to changes in its environment and to changes in the requirements placed on it
- Proactive means that the agent acts with prediction, planning and goal orientation

Flexibility, consisting of reactivity and proactivity, is thus the capability to handle possibly unexpected events and simultaneously to act with planning and goal orientation

Interactivly

- An agent can interact with its environment – especially with human actors and with other agents
- Such interaction can be on a very high level (i.e., they can be markedly communication and knowledge intensive) and they serve the purpose of coordination with third parties, i.e., the coordination of activities and the handling of mutual dependencies
  - Coordination: cooperation / competition
    - E.g., negotiation and conflict resolution in the realm of cooperative planning activities and competitive sales processes
- Interactivity requires a precise interface that normally overshadows all the internals of the agent. Thus in general interactivity designates all the (higher) social (communicative, cooperative and competitive) capabilities of an agent

Autonomy

- In the realm of its task processing, an agent can decide largely autonomously and without consultation or coordination with third parties (human users or other agents) which activities to execute
- This frequently requires or implicitly assumes that the decisions to be made by the agents are non-trivial, i.e., that they might require extensive knowledge processing or that the effects are significant
- An agent has a certain scope of decision-making authorization and freedom of action and so is subject to control by third parties only to a restricted degree
- At the bottom line, autonomy implies the ability of an agent to independently handle its own complexity and that of its application and thus especially to relieve its users while protecting their interests

Further agent attributes

- The most prominent of further attributes include:
  - Situatedness/embeddedness. An agent is connected to its environment via close sensory and/or actuatory coupling. Thus it acts and interacts directly in a concrete and socio-technical environment and not only in an abstract model of this environment
  - Learning capability/adaptivity. An agent independently optimizes its functionality with respect to the tasks that are assigned to it, which might change over time
  - Evolution, change of requirements

Learning

- What can an agent learn?
  - About the environment
    - Eg. At 5pm all shops are closed
    - Eg. When she asking me to close the door, she wants to tell me something private
  - About itself
    - E.g. I always fail to close the door
    - E.g. it is always better to do one thing at a time
    - E.g., if it is going to rain the goal of organizing a picnic will fail
- How?
  - ML techniques -> Models about
  - By design – dedicated components

Other attributes

- Temporal continuity and Persistency*:* an agent does not simply implement a one-time computation, but acts over a longer period of time
- Rationality*:* an agent acts in the realm of its capability and knowledge as well as possible with respect to fulfilling its task and goals; i.e., it maximizes its chances of success
- Self-containment*:* an agent is a functionally complete and executable entity
- Inferential capability*: an agent* acts on abstract task spec., using models of itself, situation, and/or other agents
- Mobility*:* an agent can migrate from one host to another in a self-directed way

Role/Goal vs. Task

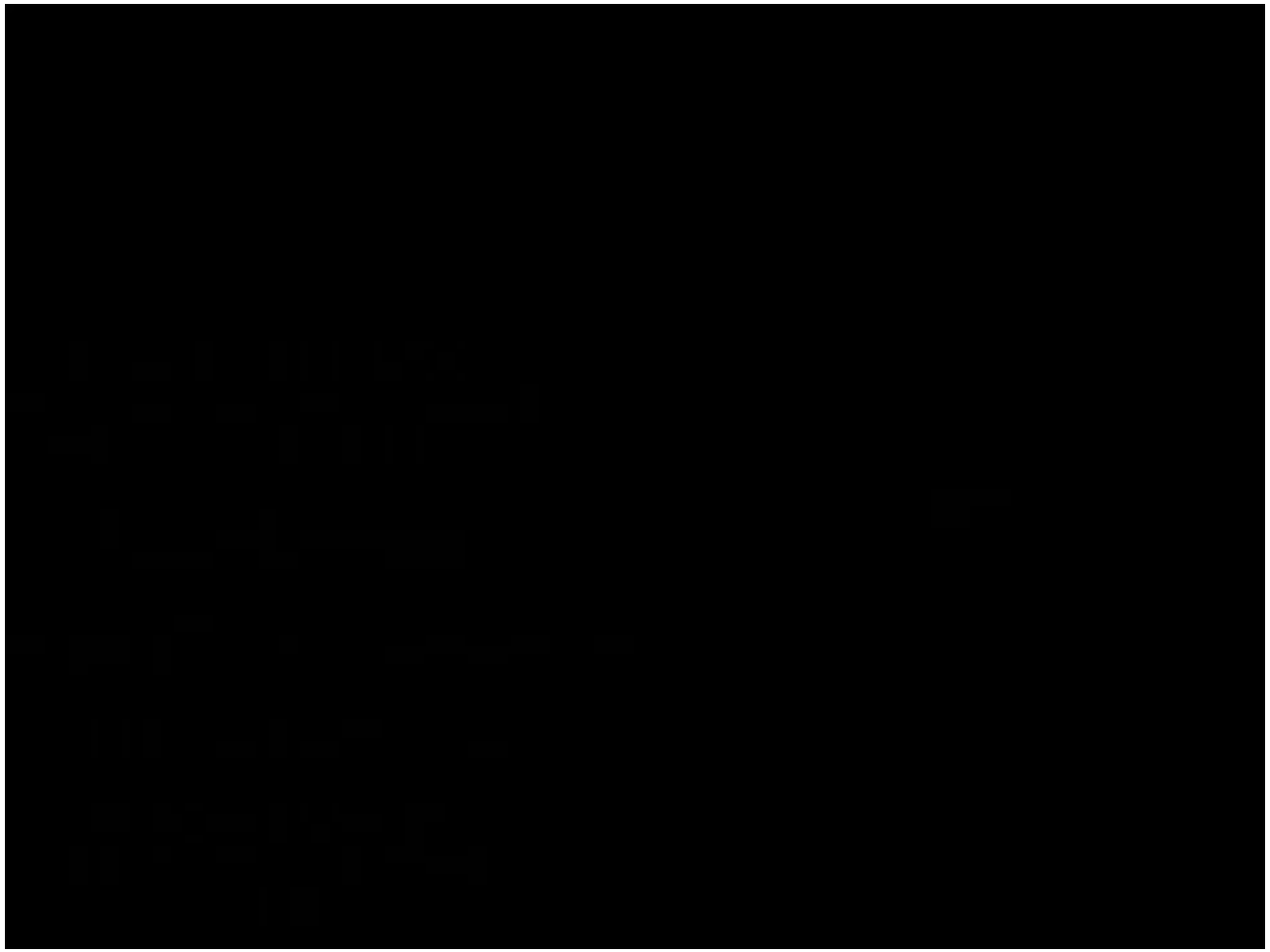
- We assign “tasks” (or services) to software
  - Example: “on-line registration software”
  - “What” and “how” are specified in advance
  - Changes in requirements are not tolerable
- We assign “roles/goals” to agents
  - Example: registration agent
  - “What” is specified in advance. “How” is determined dynamically
  - Changes in requirements can be tolerated
- An agent selects and executes tasks at runtime
  - Goal --> Tasks
- Higher level of conceptualization

Multi-agent systems

More then a single agent

- In most cases, a single agent is not enough
  - No such thing as a single agent system (!?)
  - Multiple agents are the norm to represent
    - Decentralization
    - Multiple loci of control
    - Multiple perspectives
    - Competing interests
- A Multi-agent system definition
  - A multi-agent system is one that consists of a number of agents, which interact with one-another
  - In the most general case, agents will be acting on behalf of users with different goals and motivations
  - To successfully interact, they will require the ability to cooperate, coordinate, and negotiate with each other, much as people do

A coordination example



Another definition of MAS

- *"… If a problem domain is particularly complex, large, or unpredictable, then the only way it can reasonably be addressed is to develop a number of functionally specific and (nearly) modular components (agents) that are specialized at solving a particular problem aspect. When interdependent problems arise, the agents in the system must coordinate with one another to ensure that interdependencies are properly managed"* [Katia Sycara]

- Basics of a MAS:
  - Interaction
    - Cooperation, Coordination, Competition
    - Communication
  - Local and organizational interests
    - Agent objectives vs. organizational (social) objectives
  - Organizational structure
    - Overall structure, agents' roles, norms, rules, etc.

Agent Interactions

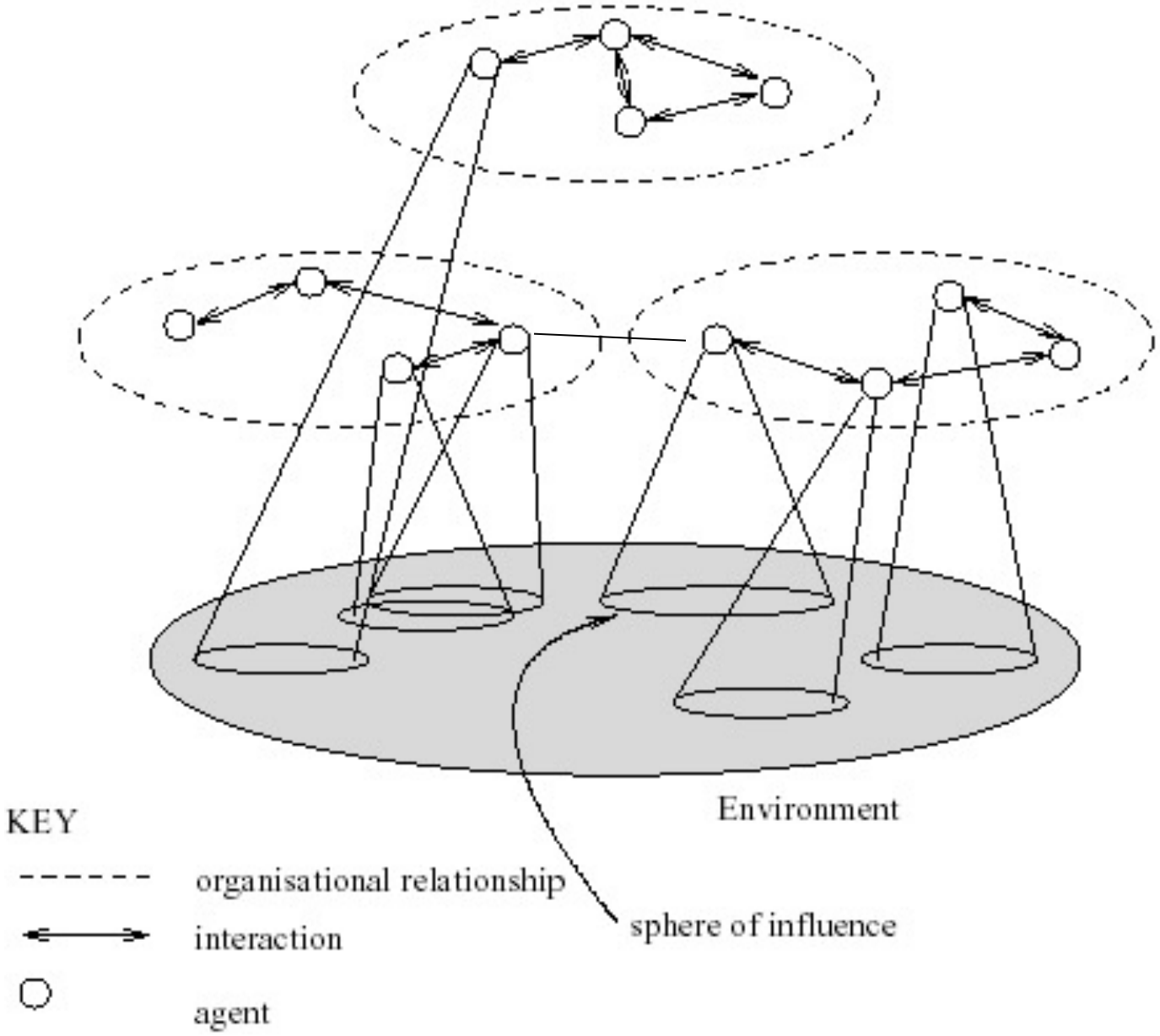
- Interaction between agents is inevitable
  - to achieve individual objectives
  - to manage inter-dependencies
- Conceptualised as taking place at knowledge-level
  - which goals, at what time, by whom, what for
- Agents act to achieve objectives
  - on behalf of individuals/companies
  - part of a wider problem solving initiative
- Agents interact in some organisational setting
  - organisational relationship between agents

Organisations

- The organisational context
  - Influences agents' behaviour
  - Relationships and rules need to be made explicit
    - Peers, teams, coalitions, institutional norms
- Agents act and interact in organisational institutions
  - They perform particular roles (which may change)
  - They obey particular norms and regulations (which are explicitly stated)
  - Different levels of an organisation
    - E.g., Strategic level, management level, operational level
  - Organisational processes

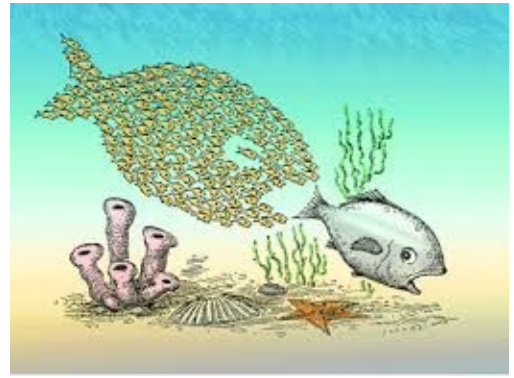
A canonical view

- A multi-agent system contains a number of agents…
  - …which interact through communication…
  - …are able to act in an environment…
  - …have different "spheres of influence" (which may coincide)…
  - …will be linked by other (organizational) relationships



Emergent behaviour

- Multi-agent system behaviour is not fully predefined
  - Behavior of a system that is not explicitly described by the behavior of the components of the system and is therefore unexpected to a designer or observer
- Result of dynamic interaction among the participants
  - flocking of birds cannot be described by the behavior of individual birds
  - market crashes cannot be explained by "summing up" the behavior of individual investors
- Local objectives generate emergent behaviours
  - Depending on the social/organizational context
  - Depending on the dynamics
  - Openness of the system



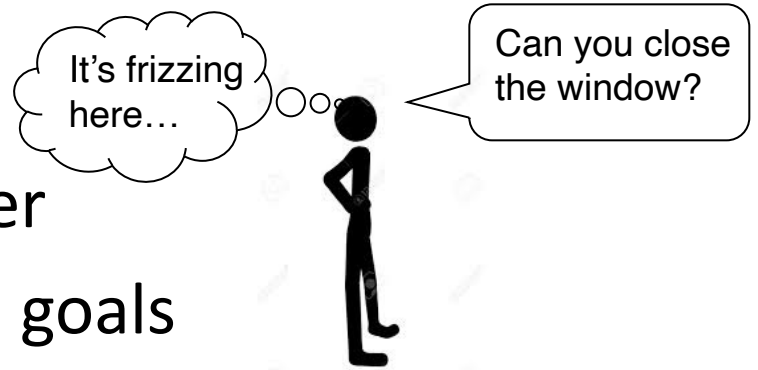
Agent communication (1)

- Basis of interaction and social organization in MAS
- Not just sending messages
  - Communication is carried out by signals between agents
- Significance
  - To be significant there must
    - be an agent that can perceive the message
    - be an interpretive system that can transform the message into meaning

Agent communication (2)

- communication consists of
  - Sending of information from a sender to a receiver
  - Encoding/decoding of information in a language
  - Transmission of information on channel or medium
  - A context in which the interlocutors (sender and receiver) are place

Agent communication (3)



- Characterize the attitude of the sender
- Indicate agent's state, intentions, and goals

- Allow agents to coordinate and make common belief
- Allow order/request & acceptance/refusal
- Send data related to the state of the world
- Verify channel function via acknowledge functions, etc.
- Request for clarification/more detail
- Ensure messages are properly understood

SE challenges

- Two different problems:
  - How do we build agents capable of independent, autonomous actions, so that they can successfully carry out goals and tasks we delegate to them?
  - How do we build agents that are capable of interacting (cooperating, coordinating, negotiating) with other agents in order to successfully carry out those delegated tasks, especially when the other agents cannot be assumed to share the same interests/goals?
- The first problem is *agent design*, the second is *society/organisational design*
  - micro vs. macro

Q/A