Un' introduzione a sistemi multi-agenti basati su logica computazionale

Paola Mello

DEIS, Università di Bologna

e-mail: paola.mello@unibo.it

Scopo del tutorial

- **Impossibile in 3 ore**
  - **essere esaustivi**
  - **fornire una panoramica completa**
- **Possibile in 3 ore (obiettivi)**
  - **Introdurre i sistemi ad agenti intelligenti e la logica (computazionale): *cosa* e *perché***
  - **Fornire *chiavi di accesso* al settore (con particolare riferimento alla parte dei protocolli e della comunicazione).**
  - **Presentare, con un esempio di attivita` di ricerca (tratto dal progetto europeo SOCS) alcune delle potenzialita` del settore.**

Outline

1. **Introduction to agents and their applications**
2. **Agent Architectures**
3. **Towards Multi Agent Systems (MAS): Agent Communication Languages and Protocols**
4. **Logic programming-based approaches to multi-agent systems: a computational logic model for the description, analysis and verification of global and open Societies Of heterogeneous *ComputeeS (SOCS)***

Part One

Introduction to agents and their applications

What is an (intelligent) Agent?

Fields that inspired the Agent field?

- Artificial Intelligence
  - Agent Intelligence, Micro-aspects of Agents
- Software Engineering
  - Agent as an abstraction
- Distributed Systems and Computer Networks
  - Agent Architectures, Multi-Agent Systems, Coordination
- Game Theory and Economics
  - Negotiation

There are many definitions of agents

Problem-solving agents

```
function SIMPLE-PROBLEM-SOLVING-AGENT( percept) returns an action
   static: seq, an action sequence, initially empty
           state, some description of the current world state
           goal, a goal, initially null
           problem, a problem formulation

   state ← UPDATE-STATE(state, percept)
   if seq is empty then do
       goal ← FORMULATE-GOAL(state)
       problem ← FORMULATE-PROBLEM(state, goal)
       seq ← SEARCH( problem)
   action ← FIRST(seq)
   seq ← REST(seq)
   return action
```

Agent - Definitions

Russel and Norvig:

"An agent is anything that can be viewed as <u>perceiving</u> its environment through sensors and <u>acting</u> upon that environment through effectors."

Maes, Pattie:

"Autonomous Agents are computational systems that <u>inhabit</u> some complex dynamic environment, <u>sense</u> and <u>act</u> autonomously in this environment, and by doing so <u>realize</u> a set of goals or tasks for which they are designed".

Hayes-Roth:

"Intelligent Agents continuously perform three functions: <u>perception</u> of dynamic conditions in the environment; <u>action</u> to affect conditions in the environment; and <u>reasoning</u> to interpret perceptions, solve problems, draw inferences, and determine actions.

IBM:

"Intelligent agents are software entities that carry out some set of <u>operations on behalf of</u> a user or another program with some degree of <u>independence</u> or <u>autonomy</u>, and in doing so, employ some <u>knowledge</u> or <u>representations</u> of the user's <u>goals</u> or <u>desires</u>"

Weak Notion of Agency

Wooldridge and Jennings:

”An Agent is a piece of hardware or (more commonly) software-based computer system that enjoys the following properties:

- **<u>Autonomy</u>: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;**
- **<u>Pro-activeness</u>: agents do not simply act in response to their environment, they are able to exhibit goal-directed behavior by taking the initiative.**
- **<u>Reactivity</u>: agents perceive their environment and respond to it in timely fashion to changes that occur in it.**
- **<u>Social Ability</u>: agents interact with other agents (and possibly humans) via some kind of agent-communication language.”**

Strong Notion of Agency

Weak Notion in addition to:

- **Mobility: the ability of an agent to move around a network**
- **Veracity: agent will not knowingly communicate false information**
- **Benevolence: agents do not have conflicting goals and always try to do what is asked of it.**
- **Rationality: an agent will act in order to achieve its goals and will not act in such a way as to prevent its goals being achieved"**

Object-oriented vs. Agent-oriented Programming

- *Basic unit:*
  - object
- *Encapsulates:*
  - state
- *Communication:*
  - Method invocation (client/server)
- *Types of message:*
  - call (no control)

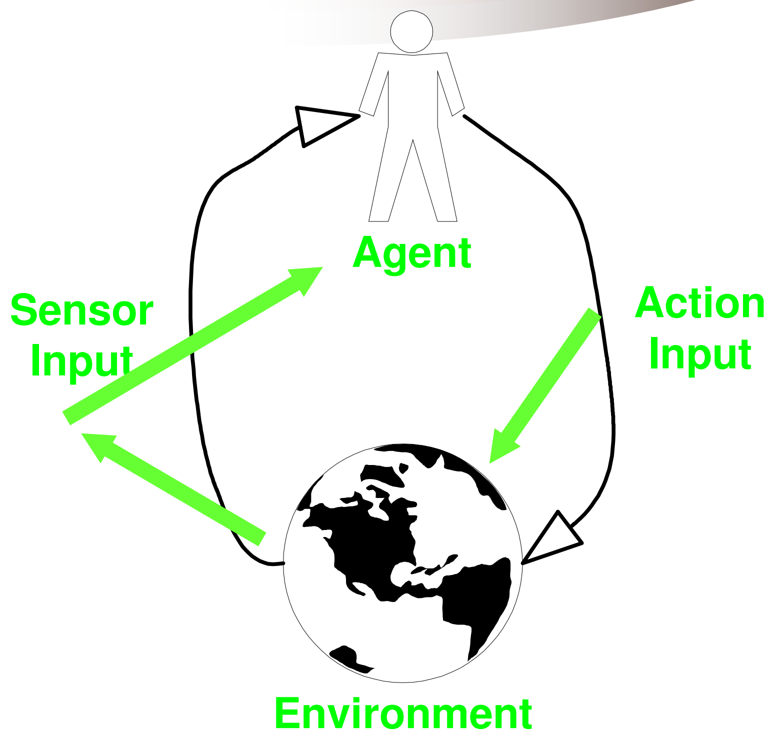
- *Basic unit:*
  - agent
- *Encapsulates:*
  - state + behaviour (can decide actions)…
- *Communication:*
  - message passing
- *Types of message:*
  - request, offer, promise, decline, actions (agents can say: no!)

Summary of Agent definitions

- **An agent <u>has</u> the <u>weak agent characteristics</u> (autonomy, pro-activity, reactivity and social ability)**
- **An agent <u>may</u> have the <u>strong agent characteristics</u> (mobility, veracity, benevolence and rationality)**
- **Generally, an agent <u>acts on behalf</u> another user or entity**

What environment?

- Phisical environment?
  - robot, SW/HW agents
  - Partially known and modifiable
  - Phisical low
- Virtual Environmemt?
  - SW agents
  - Designed by humans
  - e.g. Internet
    - Etherogeneous
    - Distributed
    - Dynamic
    - Impredictible
    - Unreliable
    - Open



Many synonyms

- Many synonyms of the term ”intelligent agent”
  - Robots
  - Software Agents or Softbots
  - Knowbots
  - Taskbots
  - Userbots
  - Computees
  - ...

Many kinds of Agents

- **Interface Agent:**
  - **Agents interacting with (human) users**
- **Information Agents:**
  - **Help users in**
    - **Find information**
    - **Gather/collect information**
    - **Select&Synthesize knowledge based on information**
- **Mobile Agents**
  - **Agents that move between runtime systems**
- **Agents in e-commerce:**
  - **Perform:**
    - **Product Brokering**
    - **Merchant Brokering**
    - **Negotiation**
- …..

Looking at agent systems…

- **When the metaphor is appropriate (*customer modelling, recommender systems, interfaces*)**
- **When there is a decision to take based on multiple sources, on large amounts of data, and in a dynamic environment (*e-markets, logistics*)**
- **For complex control tasks, when it is not possible to use a centralized controller and decentralized problem solving is needed (*supply chain management, manufacturing*)**
- **For simulation of populations of proactive individuals, when a mathematical model is not available (*traffic, games, cinema*)**
- **When it is necessary to integrate and share knowledge from multiple sources (*databases, business support*)**
- **Where autonomous problem solving is needed (*electronic trading, space crafts*)**
- **With high run-time uncertainty, or incomplete or complex information (*telecom services across multiple providers*)**

Part Two

Agent Architectures

Overview

Many existing formalisms and frameworks for agent programming

- **High-level specification languages**
- **Idea: to capture the 'essence' of agency through a set of "logical" constructs**
- **Very expressive abstract frameworks**
- **Drastically simplified concrete instantiations**

Types of Agent Architectures

Deliberative Agent Architectures (BDI and Logic-based):

- **Based on symbolic AI**
  - **Explicit symbolic model of the world**
  - **Decisision methods:**
    - **Logical Reasoning**
    - **Pattern matching**
    - **Symbolic manipulation**

Reactive architectures:

- **No central symbolic representation of world**
- **No complex reasoning**
- **Reaction to stimolous**

(Hybrid architectures)

- **Mix of Reactive and Deliberative architecture**

Deliberative Architectures

Early systems:

- **Planning Systems (STRIPS)**
- **Symbolic description of World**
- **Desired goal state**
- **Set of action descriptions**

➔ **<u>Find</u> a <u>sequence</u> of <u>actions</u> that will achieve <u>goal</u>**

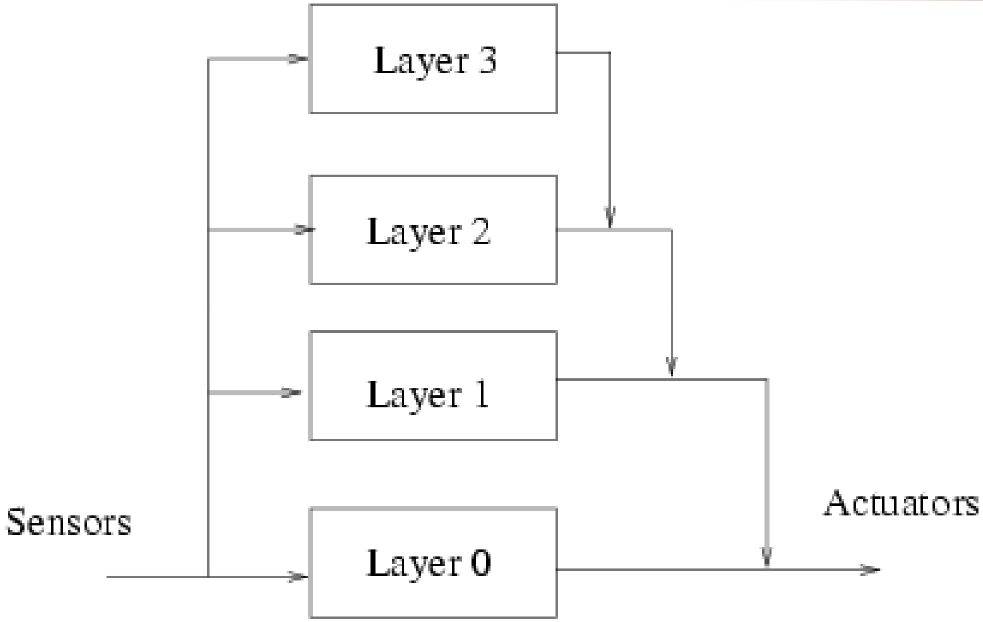
- **Use very simple planning algorithms**
- **Very inefficient planning**
- **→ towards BDI architectures**

Reactive Architectures

Brooks:

- **Intelligent Architectures can be generated <u>without</u> explicit <u>symbolic</u> (AI) <u>representation</u>**
- **Intelligent behavior can be generated without explicit abstract symbolic reasoning (AI) mechanisms**
- **Intelligence is an emergent property of certain complex systems**
  - ➔ **Effect of combined components > effect of each component times number of components**
- **”Real” intelligence is situated in the real world, not in disembodied systems such as theorem provers or expert systems**
- **Intelligent behavior arises as a result of an agent’s interaction with its environment (e.g. Ant colony)**

Reactive sub-sumption Architectures



Reactive Architecture Example

Robot's objective:

explore a distant planet (e.g. Mars), and more concretely, collect samples of a particular type of precious rock

1. **If detect obstacle then change direction**
2. **If carrying samples and at the base the drop samples**
3. **If carrying samples and not at the base, go to base**
4. **If detect a sample then pick up sample**
5. **If true then move randomly**

Deliberative Architecture: BDI

- **BDI aims to model Agents that are *rational* or *intentional***
- **The symbols representing the world correspond to *mental attitudes***
- ***Three cathegories:***
  - ***Informative (knowledge, beliefs, assumptions)***
  - ***Motivational (desires, motivations, goals)***
  - ***Deliberatives (intentions, plans).***

BDI Architectures

- **Beliefs: information about the state of the environment (*informative state*). What an agent think to know now.**
- **Desires: objectives to be accomplished, choice between possible states (*motivational state*). What an agent wishes to become true. Adopted desires are often called Goals.**
- **Intentions: currently chosen course of action (*deliberative component*). What an agent will try to make true.**

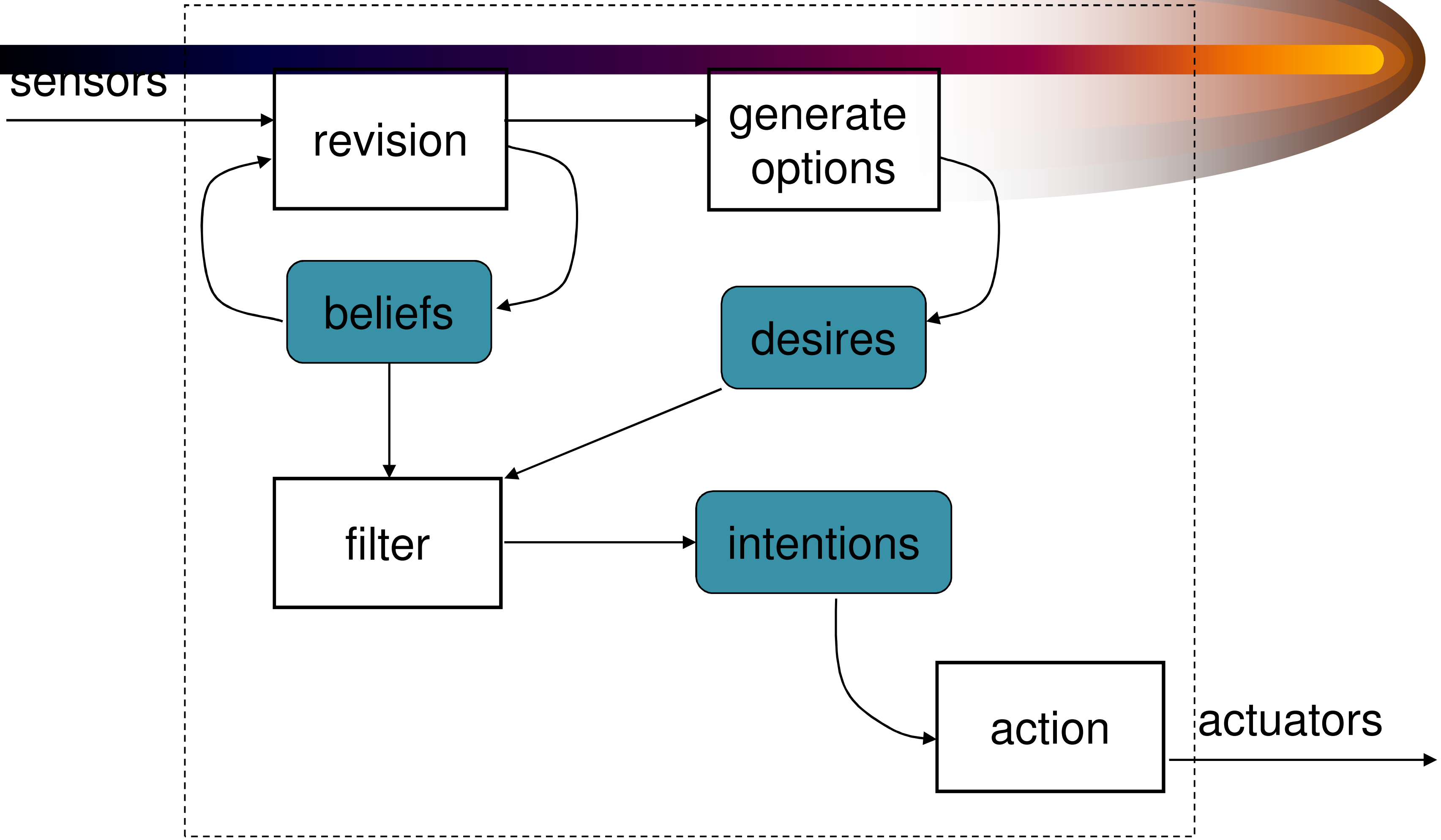
An example:

- **I *believed* the tutorial today was at 9:30am and *desired* not to be late, so I *intended* to arrive yesterday from Bologna.**

BDI formalization

- **BDI formalization has 2 main objectives:**
  - **To build practical systems**
  - **To build formally verifiable systems**
- **Building blocks:**
  - **Interpreter and cycle theory**
  - **Logics and Semantics**

BDI architecture



Intentional Notions in Modal Logic

- Classic logic is not suitable for intentional notions.
- Possible Worlds semantics
- There are a number of states of affairs, or ”worlds”
- Possible worlds may be described in modal logic
- Modal logic can be considered as the logical theory of necessity and possibility
  - The formula   A is true if A is true in <u>every</u> world accessible from the current world
  - The formula ◊A is true if A is true in <u>at least one</u> world accessible from the current world

Logic of agent knowledge

The formula   A is read as ”it is known that A” or ”agent knows A”

For group knowledge we have an indexed set of modal operators

K_1, .., K_n for

K_1 A is read ”agent 1 know A”

Example:

$K_1K_2p \wedge \neg K_2K_1K_2p$

Agent 1 knows that Agent 2 knows p, but Agent 2 doesn’t know that Agent 1 knows that Agent 2 knows p

A Logic for BDI

- **Agent *i* believes *p* to be true: $B_i p$**

- **Agent *i* desires that *p* be true: $D_i p$**

- **Agent *i* intends to make it so that *p* be true: $I_i p$**

Is BDI logic implemented in practical systems?

- **The abstract architecture is an *idealization* that faithfully captures the theory, not a *practical* system for rational reasoning**
- **Modal Logics are used with abstract semantics**
- **Many implemented systems are *inspired* to BDI concepts**
- **Solution: some important ‘choices of representation’ (simplifications) must be made…(PRS)**
- **Problem: no concrete relationship between theory and system.**

Approaches using logic

- **Many approaches in literature!!**
  - **Logic Programming**
  - **Temporal Logic – Concurrent MetateM (Fisher)**
  - **Situation Calculus – ConGolog (De Giacomo, Lespérance, Levesque)**
  - **Dynamic Logic – DyLOG (Patti)**
  - **Linear logic**

→ ***Logic Programming based approaches* in the remainder of the tutorial**

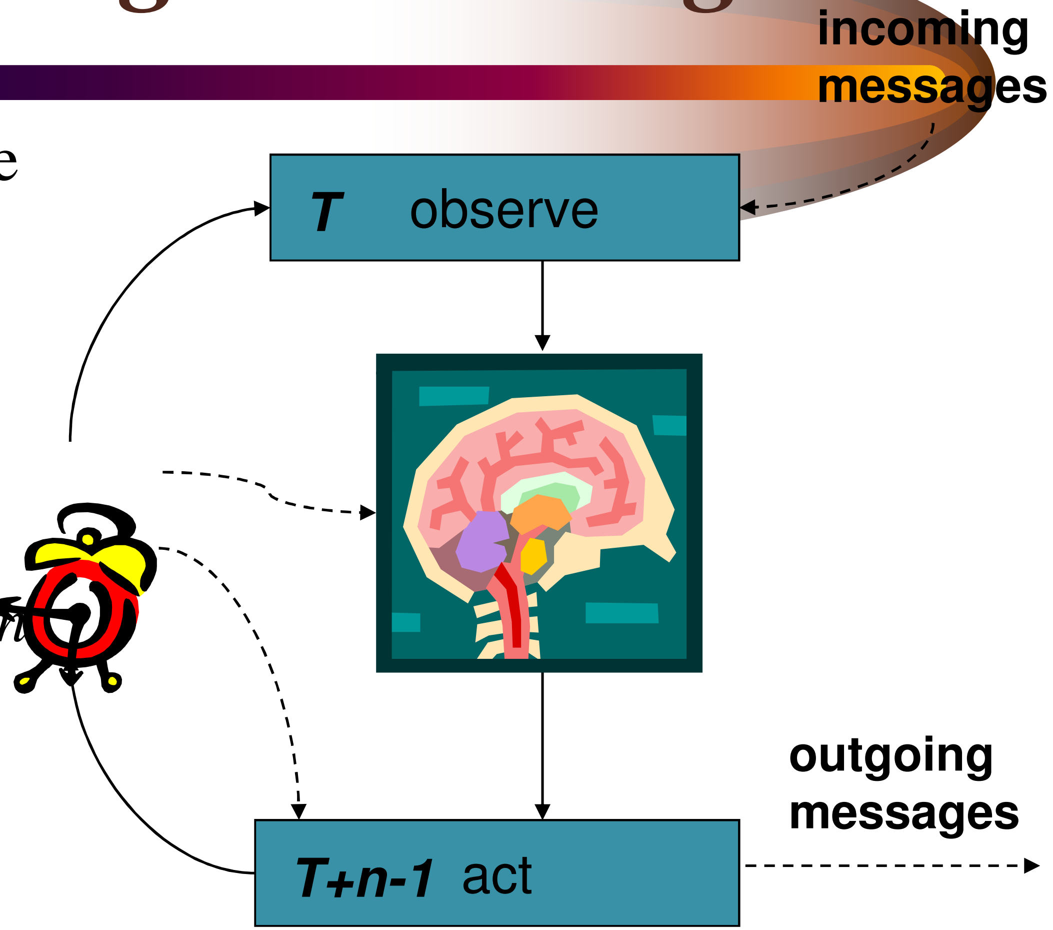
Why logic programming

- **Many agent programming frameworks**
  - **operational specification is often grounded on logic programming!**
- **Logic programming useful**
  - **for the specification of (simplified subsets of) richer programming languages,**
  - **for agent reasoning,**
  - **for knowledge manipulation,**
  - **for verification of properties of agent systems**

Logic-based agents: KS-agents

The *observe-think-act* cycle

- *To **cycle** at time T*
- ***observe** any inputs at time T*
- ***think***
- ***select** one or more actions to perform*
- ***act***
- ***cycle** at time T+n*



Thinking component

- **Backward reasoning (ALP) combined with forward reasoning (ICs)**
- **IFF proof-procedure [FK97]: handles IFF definitions and forward integrity constraints (IC)**

- **Backward reasoning based on based on IFF definitions:**
  - **it unifies a goal G'**
  - **with a IFF definition $G \leftrightarrow D_1 \vee \ldots \vee D_n$**
  - **finding a subgoal $D_1 \vee \ldots \vee D_n$**

- **Forward reasoning based on IC**
  - **it matches an observation or atomic goal: O**
  - **with a condition of an IC $O' \wedge Q \rightarrow R$**
  - **finding a new IC (to be true) $Q \rightarrow R$**

Example

happens **(become-thirsty, T)**
$\rightarrow$ *holds* (quench-thirst, [T1, T2]) & $T \leq T1 \leq T2 \leq T+10$

holds (quench-thirst, [T1, T2]) $\leftrightarrow$ _holds_ (drink-soda, [T1, T2]) or
holds (drink-water, [T1, T2])

holds (drink-soda, [T1, T2]) $\leftrightarrow$ _holds_ (have-glass, [T1, T']) &
holds (have-soda, [T'',T2]) &
_do_ (drink, T2) &
$T1 < T'' < T2 \leq T'$

holds (have-soda, [T1, T2]) $\leftrightarrow$ _do_ (open-fridge, T1) &
_do_ (get-soda, T2) &
$T1 \leq T2$

holds (drink-water, [T1, T2]) $\leftrightarrow$ _holds_ (have-glass, [T1, T']) &
_do_ (open-tap, T'') &
_do_ (drink, T2) &
$T1 < T'' < T2 \leq T'$

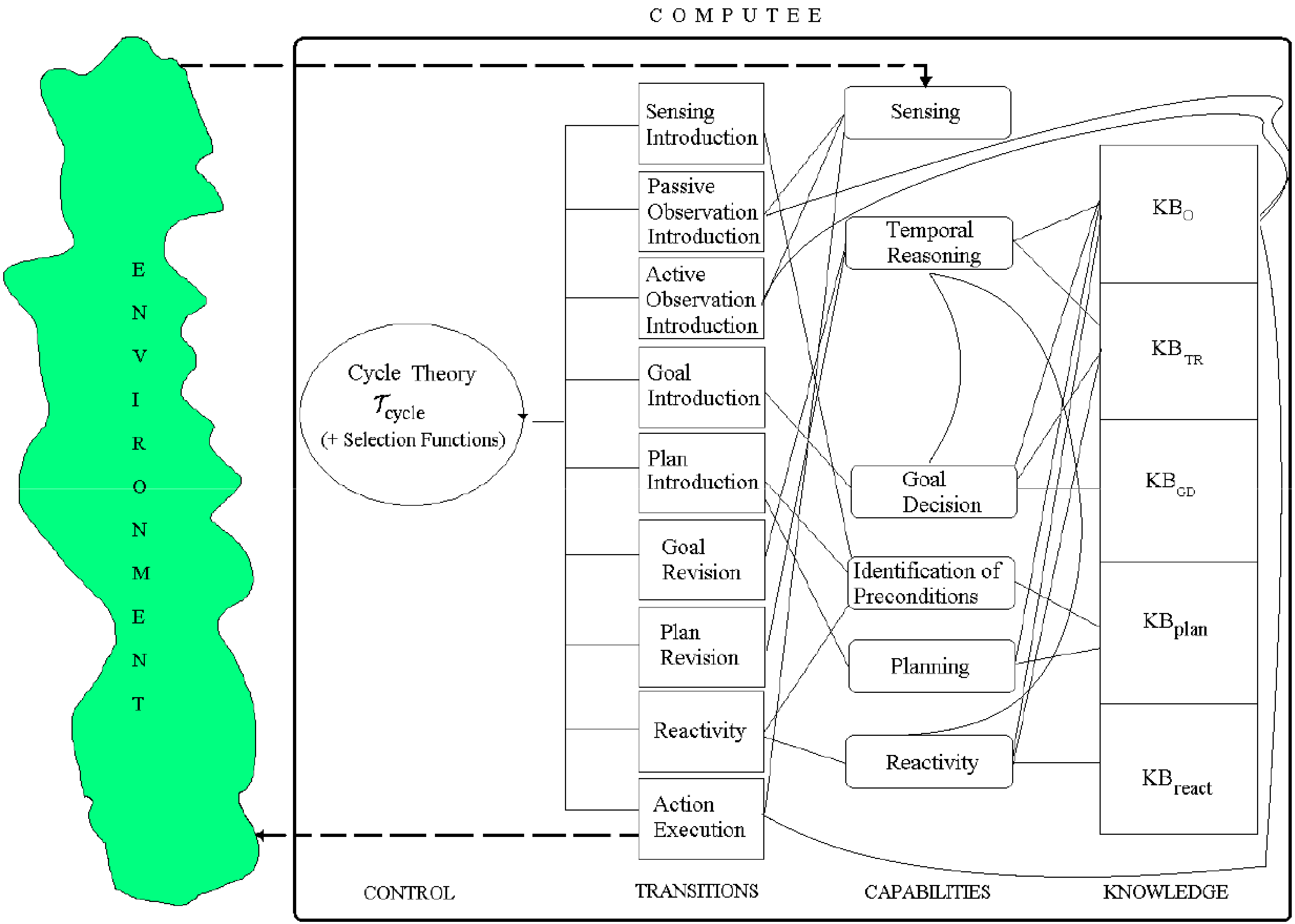
KS-agents vs. BDI

- BDI: uses two languages (modal logic specifications / procedural implementation);

- KS: uses the same language for specification and implementation

The SOCS computee: *a computational logic based intelligent agent*

- An *internal* (*mental*) *state*;
- A set of *reasoning* capabilities for performing
  - planning,
  - temporal reasoning,
  - identification of preconditions of actions,
  - reactivity, and
  - goal decision;
- A *sensing* capability;
- A set of formal *state transition rules*;
- A set of *selection functions*;
- A *cycle theory*.


COMPUTEE
E N V I R O N M E N T
Cycle Theory
$\mathcal{T}_{cycle}$
(+ Selection Functions)
Sensing Introduction
Passive Observation Introduction
Active Observation Introduction
Goal Introduction
Plan Introduction
Goal Revision
Plan Revision
Reactivity
Action Execution
Sensing
Temporal Reasoning
Goal Decision
Identification of Preconditions
Planning
Reactivity
KB_O
KB_{TR}
KB_{GD}
KB_{plan}
KB_{react}
CONTROL
TRANSITIONS
CAPABILITIES
KNOWLEDGE

Part Three

Multi Agent Systems (MAS): Agent Communication Languages and Protocols

Motivation behind MAS

- To solve problems too large for a centralized agent
- To provide a solution to inherently distributed problems
- To provide solutions where expertise is distributed
- To offer conceptual clarity and simplicity of design

Benefits:

- Faster problem solving
- Flexibility
- Increased reliability
- Different heterogeneity degrees

Cooperative and Self-interested MAS

- Cooperative
  - Agents designed by <u>interdependent</u> designers
  - Agents act for increased good of the system (i.e. MAS)
  - Concerned with increasing the systems performance and not the individual agents
- Self-interested
  - Agents designed by <u>independent</u> designer
  - Agents have their own agenda and motivation
  - Concerned with the benefit of each agent (”individualistic”)

⇒ The latter more realistic in an Internet-setting?

Motivation for Agent Communication and MAS

- **Communication is required for cooperation between agents**
- **Societies can perform tasks no individual agent can**
- **Autonomy encourages disregard for other agents' internal structure**
- **Communicating agents need only know a ”common language”**
- **Supports heterogenous agents**

A layered architecture

SOCIETIES

PROTOCOLS

ACL LANGUAGE

PLATFORM

Basic Architecture

Platform

- **handle simple objects with no associated semantics**
- **support communication mechanisms (e.g., RPC) and low-level protocols (e.g., TCP/IP).**

Agent Communication Language (ACL)

- **provides agents with a means to exchange information and knowledge.**
- **handles propositions, rules, actions etc..**

Protocols

- **represent the allowed interactions among communicating agents of a society.**

Society

- **intended as a group of agents possibly with roles, common protocols, and laws.**

Features of ACLs

- Efficient
  - Few bytes but much meaning, rich semantics for each message
  - Easy-to-use for both machines and humans
- Based on Open Standards
  - Allow agent and agent systems by different vendors to communicate
- Flexible
  - Easy to extend without changing the language, using ontologies
  - Support several syntactic representations
- Have clear non-ambigious semantics and syntax
  - ”logic features”
  - Avoid contradictions
- Expressive and High-level
  - Be inspired by natural language

Speech Act Theory and ACLs

- **Theory of human communication with language.**
  - **Consider sentences for their effect on the world**
  - **A *speech act* is an act carried out using the language**
- **Several categories of speech acts.**
  - **Orders, advices, requests, queries, declarations**
- **Agent Communication Languages use messages.**
  - **Messages carry speech act from an agent to another**
  - **A message has *transport slots* (sender, receiver,…)**
  - **A message has a *type* (request, tell, query..)**
  - **A message has *content slots*.**

Say What?

- **An Agent Communication Language captures:**
  - **The speaker (sender) and the hearer (receiver) identities**
  - **The kind of speech act the sender is uttering.**
  - **Is this enough? ("I request that you frtafs the fgafag")**
- **Not only words but also the world!**
  - **There are also things**
  - **A common description of the world is needed**
  - **Describing actions, predicates and entities: ontologies**

Cosa sono le ontologie

- Filosofia/Computer ScienceAI: area dell'intelligenza artificiale che studia i metodi per rappresentare correttamente l'universo che ci circonda.

Perchè servono in CS?

- Condivisione di conoscenza: per non duplicare sforzi nello sviluppo di sistemi software

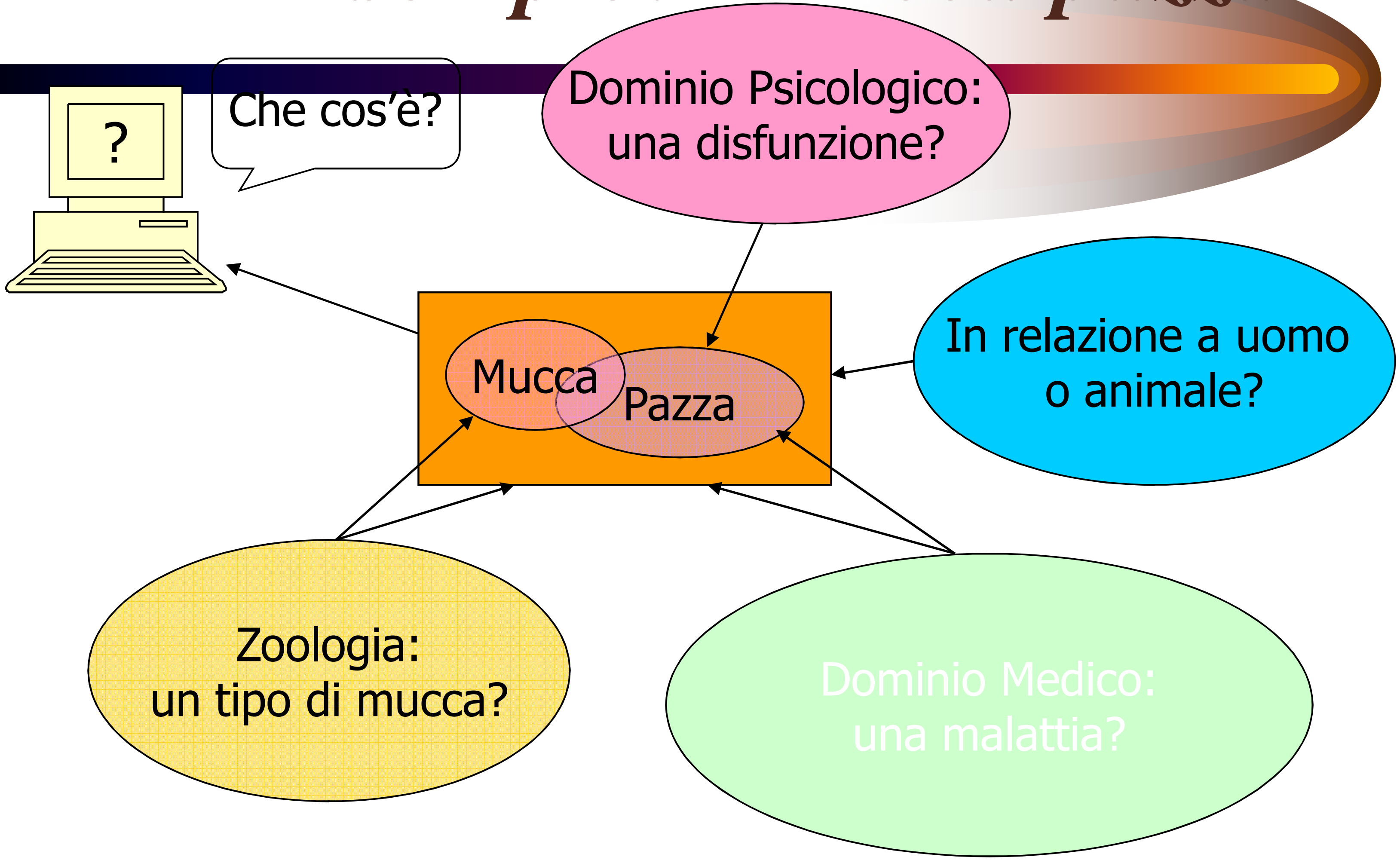
- Comunicazione: sia tra agenti software (tra di loro) che tra agenti software e esseri umani

Semantic Web!

Ontogie e Web Semantico

- Possibilità di accesso e acquisizione della conoscenza tramite www
- Costo trascurabile per acquisire basi di conoscenza
- Necessità di organizzare, integrare e interrogare basi di conoscenza
- Necessità di sorgenti di conoscenza facilmente accessibili da macchine e processi automatici
- Necessità di una conoscenza riutilizzabile e condivisibile (in contesti e forme differenti)

Esempio: "mucca pazza"



Problemi di fondo

1. Occorre eliminare la confusione terminologica e concettuale ed individuare le *entità* cui un pacchetto di conoscenza si riferisce.
2. Organizzare e rendere esplicito il *significato referenziale* permette di *comprendere* l'informazione.
3. Condividere questa *comprensione* facilita il recupero e il riutilizzo della conoscenza tra agenti e in contesti diversi.

⇓

ONTOLOGIE

Ontologia

Definizione formale di un dominio di conoscenza

Isolare una parte del mondo
e i suoi concetti fondamentali

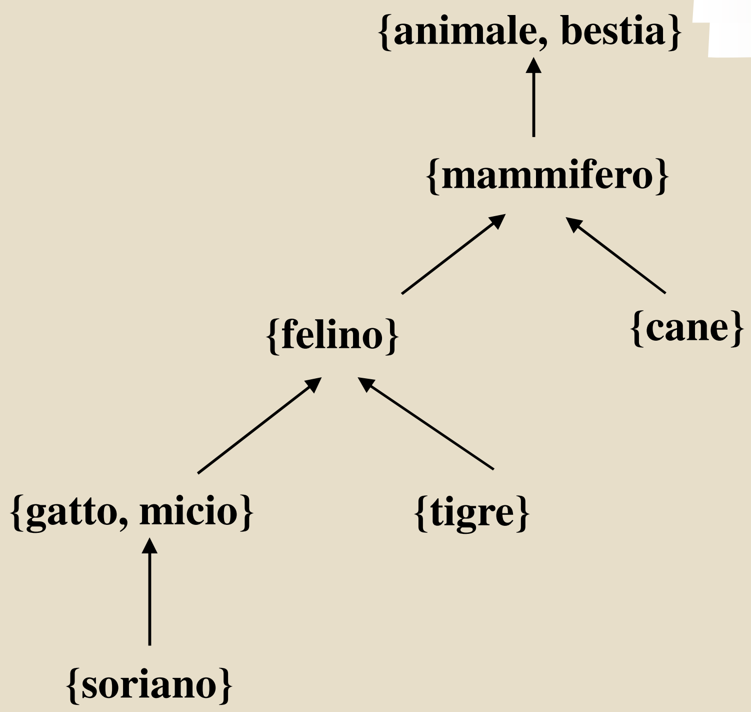
Enumerare e definire (in modo più o meno formale)
i concetti e le relazioni che tra essi sussistono:
→ classi, proprietà, assiomi, individui

⇓

Una descrizione strutturata gerarchicamente dei concetti importanti e delle loro proprietà che trovi il consenso di diversi attori interessati a condividerla e utilizzarla.

Esempio di Ontologia: WordNet

<tigre, cane, animale, mammifero, bestia, micio, soriano, gatto, felino>



Speech acts – Types

- **Assertives: ”It rains”**
- **Directives: ”Close the window”**
- **Commisives: ”I will”**
- **Expressives: ”Excuse me”, ”congratulations”**
- **Declaratives: ”In name of this city”**
- **Permissives: ”You <u>may</u> shot the door”**
- **Prohibitives: ”You <u>may not</u> shot the door”**

Agent Communication Languages

- **Two major proposals**
  - **KQML (1993 - ~1998)**
    Knowledge Query and Manipulation Language
    Basis: work by the ”Knowledge Sharing Effort” group
  - **FIPA ACL (1996 - now)**
    Defined by The Foundation for Intelligent Physical Agents (FIPA)

- **Define a number of *communicative actions / performatives***
- **Semantics based on *mental states*.**

KQML Statement Structure

KQML Statements consists of

1. **A performative**
2. **Parameters and context information**

General syntax:

(KQML-performative

:sender word

:receiver word

:language word

:ontology word

:content expression

...)

KQML example

(tell

:sender Agent1
:receiver Agent2
:language KIF
:ontology Blocks-World
:content (AND (Block A)(Block B)(On A B))

(inform

:sender i
:receiver j
:language Prolog
:ontology weather42
:content weather(today,raining)

FIPA ACL – competing/extending KQML

- FIPA vs KQML
  - Both are based on speech act
  - Different (richer) set of performatives
  - FIPA has a more formal basis
  - FIPA can describe interaction protocols

What is FIPA?

- The Foundation for Intelligent Physical Agents (FIPA) is a non-profit association.
- FIPA's purpose is to promote the success of emerging agent-based applications, services and equipment.
- FIPA operates through the open international collaboration of member organisations, which are companies and universities active in the agent field.
- URL: http://www.fipa.org/
- Recentemente ha terminato il suo lavoro.

ACL (BDI-based) Semantics

- **Mentalistic approaches define ACL semantics in terms of agents' mental state (BDI)**
- **Semantics based on *mental states*:**
  1. **An *intuition* given in natural language**
  2. **An expression describing the illocutionary act**
  3. **Pre-conditions for sender and receiver**
  4. **Post-conditions in case of successful receipt**
  5. **Any comments**

- **The formal semantics of a FIPA communicative act (CA) comprises:**
  - **What must be true for the sender before sending a CA (feasibility precondition)**
  - **Which intentions of the sender could be satisfied as a consequence of sending the CA (rational effect)**

FIPA ACL semantics for inform

$<i, \text{INFORM}\ (j, \phi)>$

FP : $B_i\ \phi$ *and not* $B_i\ (B_j\ \phi$ *or* B_j *not* $\phi)$

RE : $B_j\ \phi$

- The sender informs the receiver that a given proposition is true.
- The content is a predicate
- The sender believes the content
- The sender wants the receiver to believe it.

FIPA ACL semantics for request

`< Sender, REQUEST (Receiver,a)>`

FP: FP(a)[Sender/Receiver] *and*
B_{Sender} Agent (Receiver,a) *and*
not B_{Sender} $I_{Receiver}$ Done(a)

RE: Done(a)

- **`FP(a)[Sender/Receiver]` denotes the part of the FPs which are mental attitudes of the Sender (and do not directly involve the receiver).**
- **B_{Sender} `Agent (Receiver,a)` means that Sender believes that Receiver can perform a;**
- ***not*** **B_{Sender} $I_{Receiver}$ `Done(a)` means that the Sender does not believe that the Receiver intends to perform a.**

ACL (BDI-based) Semantics

- **Agent `Sender` should not only be aware of his own mental state, but also have beliefs (in this case, negative) about agent `Receiver` 's mental state.**

- **Critics to BDI ACL semantics**:
  - **in general agents cannot read each other's minds**
  - **in open societies of heterogeneous agents it is not always possible to rely on agent mental states [Singh98]**

ACL "Social" Semantics

- **An ACL's formal semantics should better emphasise social agency.**
- **Communication is inherently public and thus depends on the agent's social context.**
- **The social approach defines ACL semantics in terms of the social effects of the communicative acts.**
- **Some questions…**
  - **Why constrain agents' social acts?**
  - **Why refer to a particular agent architecture?**
  - **How to *verify* communication?**
  - **How to approach *openness* and heterogeneity?**

Conclusions on current ACLs

- Agent Communication Languages have a common basis – speech act
- Can all desirable communication primitives be modeled after speech acts? Should they?
- Syntax is well specified, but current research is on describing semantics (versus a social approach)
- Intentional level description: which mental attitudes, what definitions?
- Problems with mental attitudes: from theory to practice
- How can we test an agent’s compliance with the ACL?

Interaction Protocols

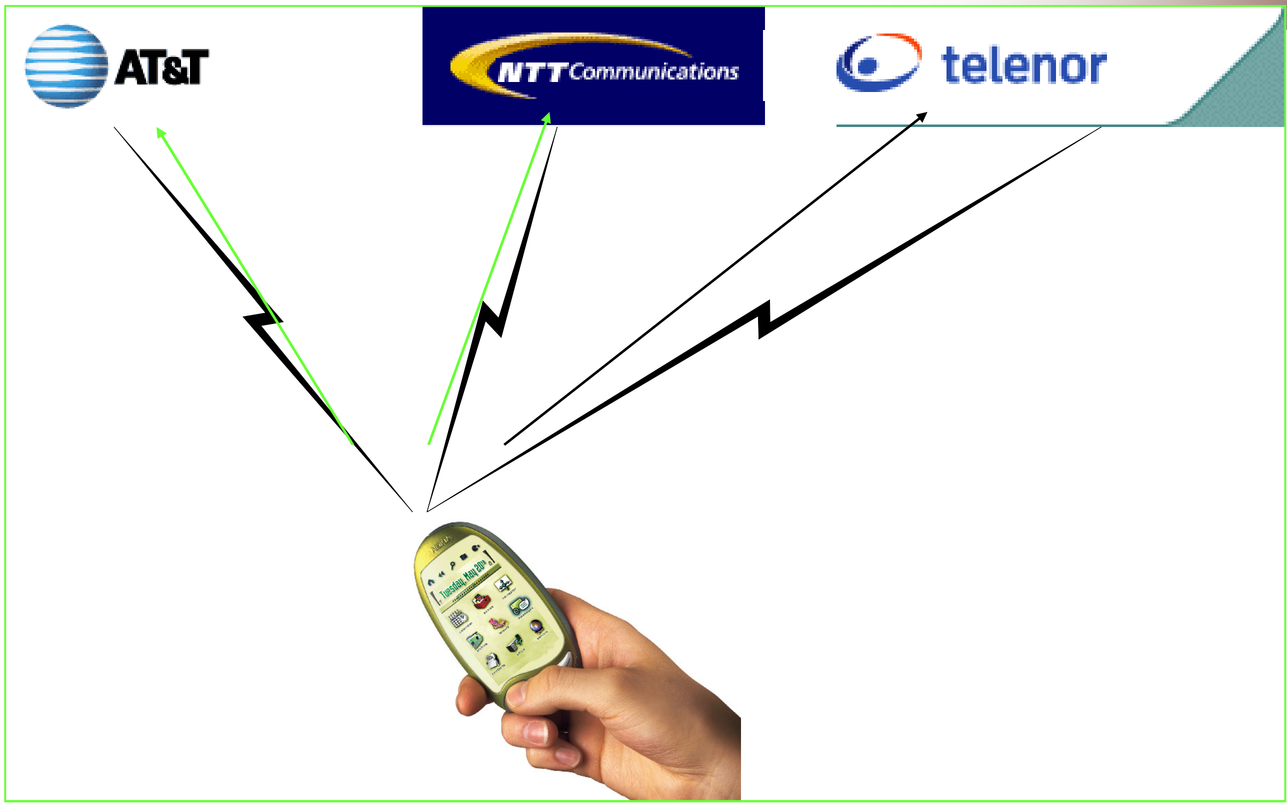
- Observing a single CA says nothing about the receiver.
- We must move from utterances to conversations: desirable sequences of messages for particular tasks.
- Protocol: set of interaction rules
  - what actions each agent can take at each time.
- Formalisms for modeling protocols (e.g. Petri-Nets, finite state machines, AUML diagrams), specify protocols as legal sequences of actions.
- FIPA specifies an IP Library, containig conversation templates

UMTS Provider Competition Protocol

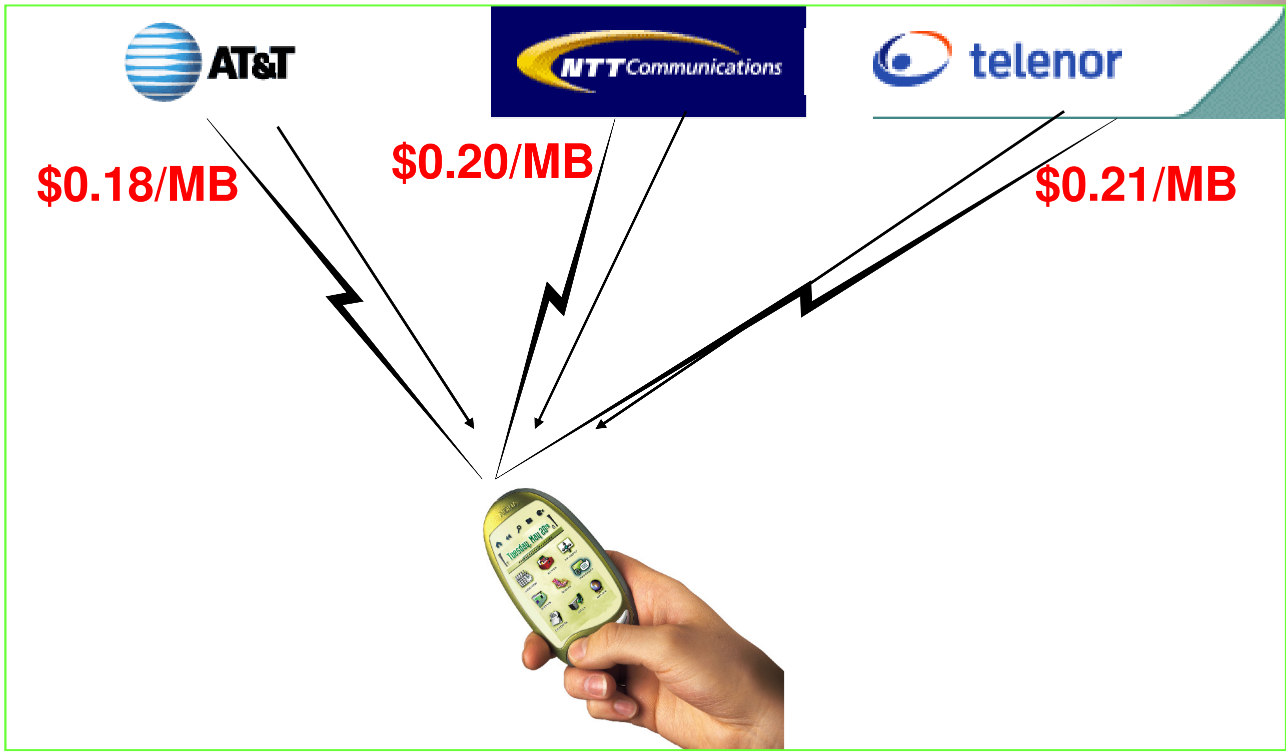
Description of problem

- **Automatic Selection of UMTS provider**
- **Mobile Device automatically negotiates for a price with the possible providers**

Market Situation (Fiction Example)



Bids



Lowest Bidder wins

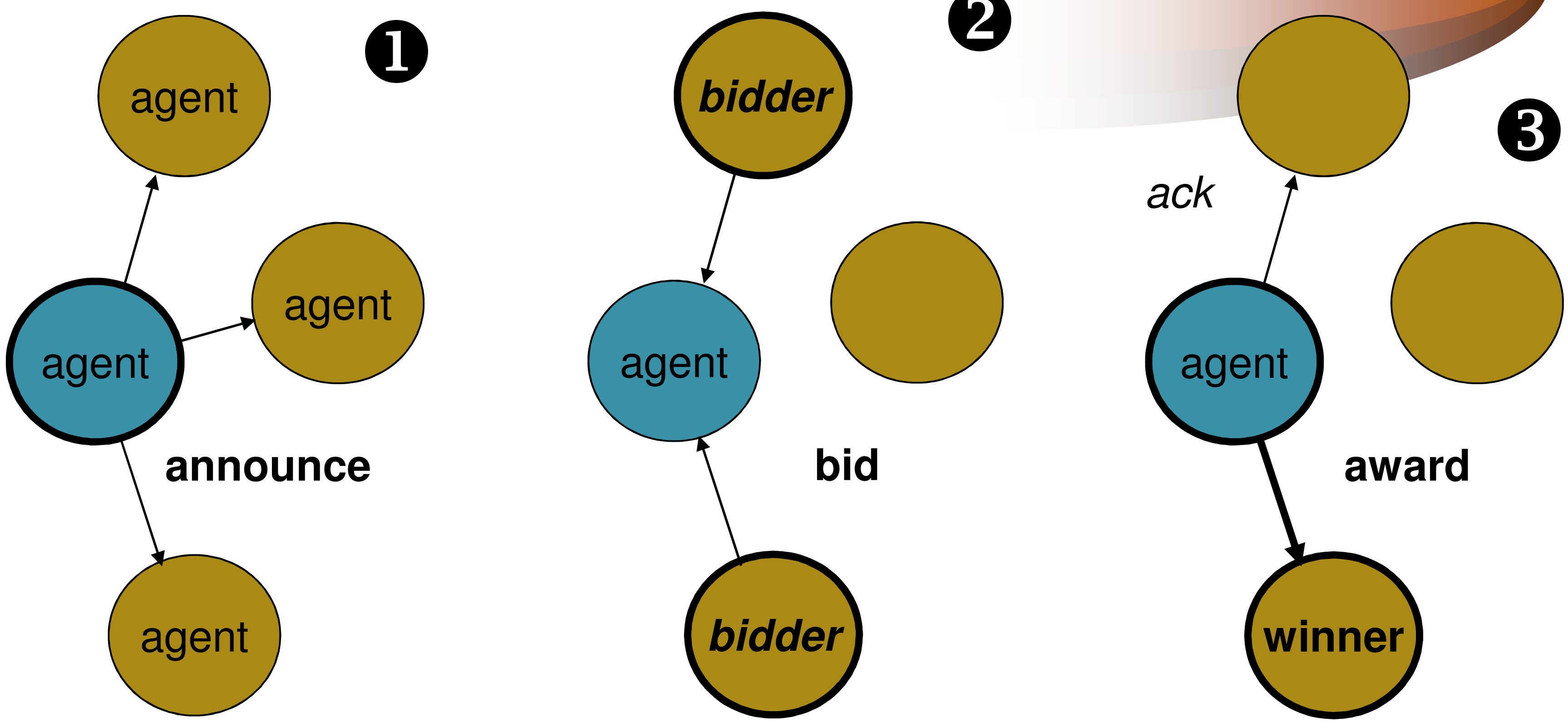
$0.18/MB

Negotiation: Contract-Net

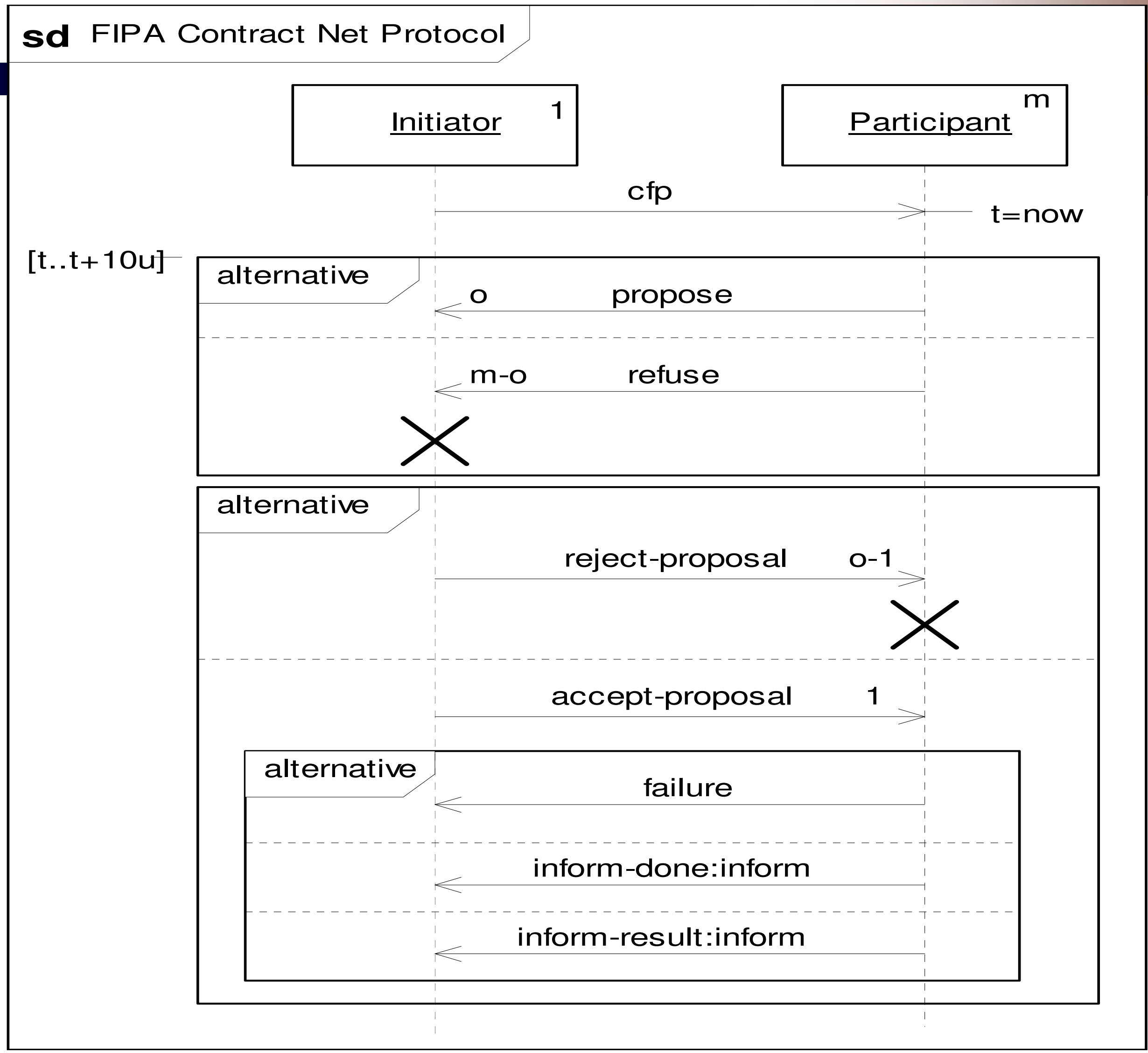
- **Davis&Smith**

  Negotiation is a process of improving agreement (reducing inconsistency and uncertainty) on common viewpoint or plans through the exchange of relevant information
- **Complex Interaction Protocol**
- **It embeds policies**
- **One-to-many IP**
  - **One manager agent**
  - **N contractor agents**
  - **A call for proposals is issued**
  - **A contractor is selected among proponents**

Negotiation for task allocation (Contract Net Protocol)



(AUML) FIPA Contract Net Protocol



Protocols and Properties

- ***Protocols*** **are used to define the allowed sequences of utterances that agents can exchange**
- **Many protocols can be used to achieve the same objective (e.g. resource sharing)**
- ***Properties*** **are important!!**
  - **properties of protocols (fairness, guaranteed termination, privacy, …)**
  - **properties of participants**
    - **statically verifiable**
    - **dynamically verifiable (e.g. compliance)**

Protocols and social semantics

- **Protocols are over-constrained thus affecting autonomy, heterogeneity, opportunities, exceptions.**

- **According to Yolum, Singh:**

  "Participants must be constrained in their interactions only to the extent necessary to carry out the given protocol and no more"

  Protocol: set of constraints on the social behaviour (motivations for commitment and committed-based semantics).

Society

A MAS is more than a bunch of Agents

- Functional definition of a society
- Society defined by specifying:
  - roles;
  - rules (allowed actions, communication protocols, social commitments);
  - operations to join and exit the society.

Society

- **Society modelling**
  - **teamwork model, benevolence is presumed;**
  - **deontic model, based on obligations, authorizations, committments;**
  - **reactive and evolving/auto-organizing models;**
- **Consequently, different types of society:**
  - **open/closed;**
  - **centralized/decentralized;**
  - **with common or individual goals.**

Society

- **Assumptions:**
  - Members must conform (and agree) to its laws
  - Members have a common communication language and ontology w.r.t. communicative acts
  - Roles are assigned to agents when they enter a society (and they could change over time)
- **These specifications imply:**
  - a mechanism establishing and enforcing conventions that standardize interactions (Institution).
  - the presence of a Social Management Infrastructure.

New challenges: Logics

- **Logics?**
  - **For prototyping**
  - **For intelligence (reasoning, goals, consistency)**
  - **For verification (individuals, interactions)**

Where do we use logics?

strongly logic-based approach

agent

society

rationality and pro-activeness

reactivity to external stimuli

protocols and norms

emerging behaviour

formal results?

efficiency?
easy integration?
legacy systems?

Why Logic Programming

- **Logic programming can be used to bridge the gap between**
  - **theory (high level specification) and**
  - **practice (execution model) of agents**
- **Most research on logic programming-based agents focusses on single aspects of agency (reasoning, updates, anticipation, interaction)**
- **We show a full-fledged agent model (SOCS) based on logic programming, and a computational model for agent interaction**

Verification for open systems

- Guerin & Pitt, 2002: 3 kinds of verification:
  1. verify that an agent will always comply
  2. *verify compliance by observation*
  3. verify protocols' properties
- 1) we need to know the agent behaviour
- 2) is particularly suited for *open societies*
- 3) e.g. termination, e other specific properties.

Conclusions

- Logic useful for
  - modelling & specification
  - operational model ⇒ implementation/prototyping
  - identification and verification of properties
- Computational logic used to tackle several different aspects of agent-based programming
- Theory and practice can work together!
- Formal results from logic programming to multi-agents systems!

Pointers to Agent Research

- Web sites:
  - AgentLink II: http://www.agentlink.org
  - UMBC Agent WEB: http://agents.umbc.edu/
  - Agent Based Systems: http://www.agentbase.com/survey.html
  - Agent Construction Tools: http://www.agentbuilder.com/AgentTools/
- Journals
  - Journal of Autonomous Agents and Multi-Agent Systems
- Conferences and Workshops
  - International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS) – next in New York, deadline: 16 January 2004
  - Past events: ATAL, ICMAS, AA and related WS (LNAI, IEEE, and ACM Press)

Pointers to Computational Logic

- Journals
  - Artificial Intelligence
  - Journal of Logic and Computation
  - Annals of Mathematics and Artificial Intelligence
  - The Knowledge Engineering Review
  - Journal of Group Decision and Negotiation
  - Theory and Practice of Logic Programming
  - Journal of Cooperative Information Systems
- Conferences and Workshops
  - Workshop on **Computational Logics in Multi-Agent Systems** (**CLIMA**)

Declarative Agent Languages and Technologies (**DALT**) – *watch AAMAS'04 website*

Pointers to MAS

- Surveys on multi-agent systems

  [JSW98] N. Jennings, K. Sycara, and M. Wooldridge, *A Roadmap of Agent Research and Development*. AAMASJ 1998.

  [WC00] M. Wooldridge and P. Ciancarini, *Agent-Oriented Software Engineering: The State of the Art.* In Proc. First Int. Workshop on Agent-Oriented Software Engineering, LNCS, 2000

  [LMP03] M. Luck, P. McBurney, C. Preist, *Agent Technology Roadmap. 2003*. Available electronically http://www.agentlink.org/roadmap/

- Books

  [Wei99] G. Weiss (ed.), *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999

  [Woo02] M. Wooldridge, *Introduction to Multi-Agent Systems*. John Wiley & Sons, 2002.

Pointers to Research Groups on Computational Logic and Agents

- **3APL**: Intelligent Systems Group, University of Utrecht, http://www.cs.uu.nl/groups/IS/agents/agents.html
- **BOID**: http://boid.info/
- **RMIT**: http://www.cs.rmit.edu.au/agents/
- **GOLOG**: Cognitive Robotics Group, University of Toronto, http://www.cs.toronto.edu/cogrobo/
- **IMPACT**: University of Maryland, http://www.cs.umd.edu/projects/impact/
- **JACK**: The Agent Oriented Software Group, http://www.agent-software.com/
- **MetateM**: Logic and Computation Group, University of Liverpool, http://www.csc.liv.ac.uk/~michael/
- **DESIRE**: http://www.cs.vu.nl/vakgroepen/ai/projects/desire/
- **CaseLP**: DISI, Università di Genova, http://www.disi.unige.it/index.php?research/ai-mas
- **ALIAS**: DEIS, Università di Bologna, http://lia.deis.unibo.it/research/ALIAS/
- **DyLOG**: DI, Università di Torino, http://www.di.unito.it/~alice/
- **SOCS**, EU Project, http://lia.deis.unibo.it/research/socs
- **ALFEBIITE**, EU Project, http://www.iis.ee.ic.ac.uk/~alfebiite/
- **Dagstuhl** seminar 02481 on logic based MAS: http://www.cs.man.ac.uk/~zhangy/dagstuhl/