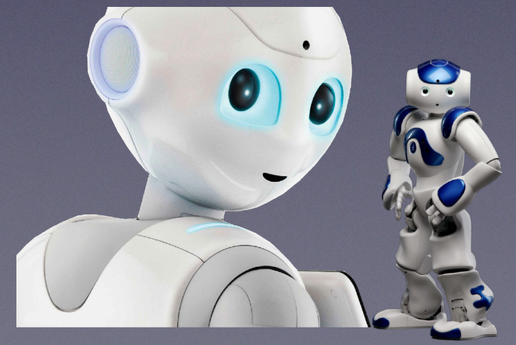
UNIVERSITÀ DEGLI STUDI DI PALERMO
dipartimento di ingegneria unipa

AGENTS AND ROBOTS JUSTIFY ACTIONS

Implementing trustfulness and explainability

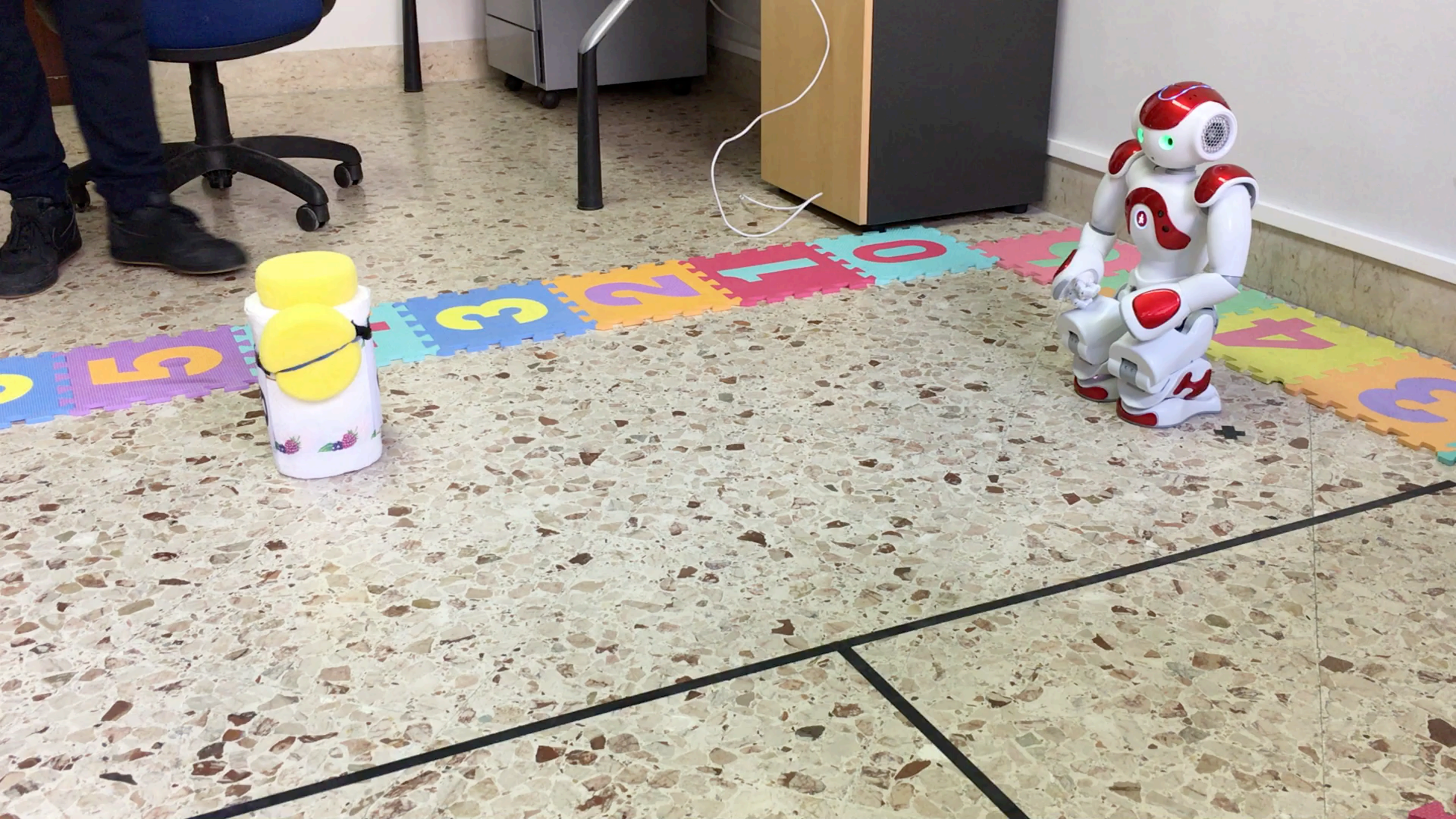
Angelo Maria Pio Sabella, Valeria Seidita and Antonio Chella

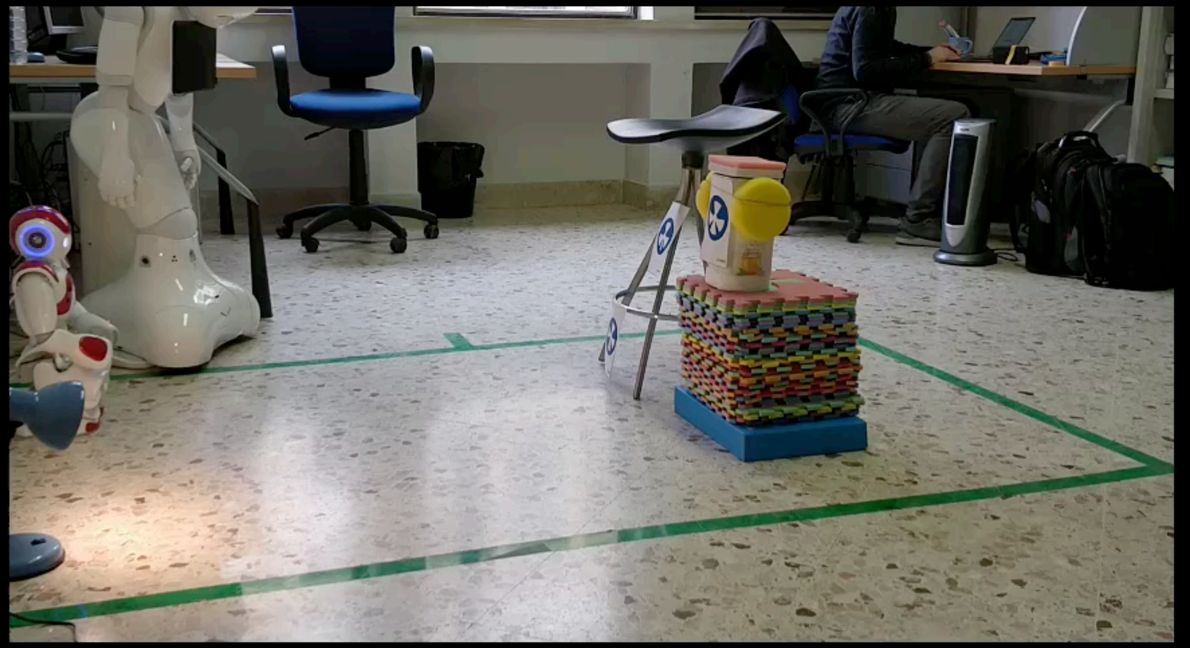


Objectives

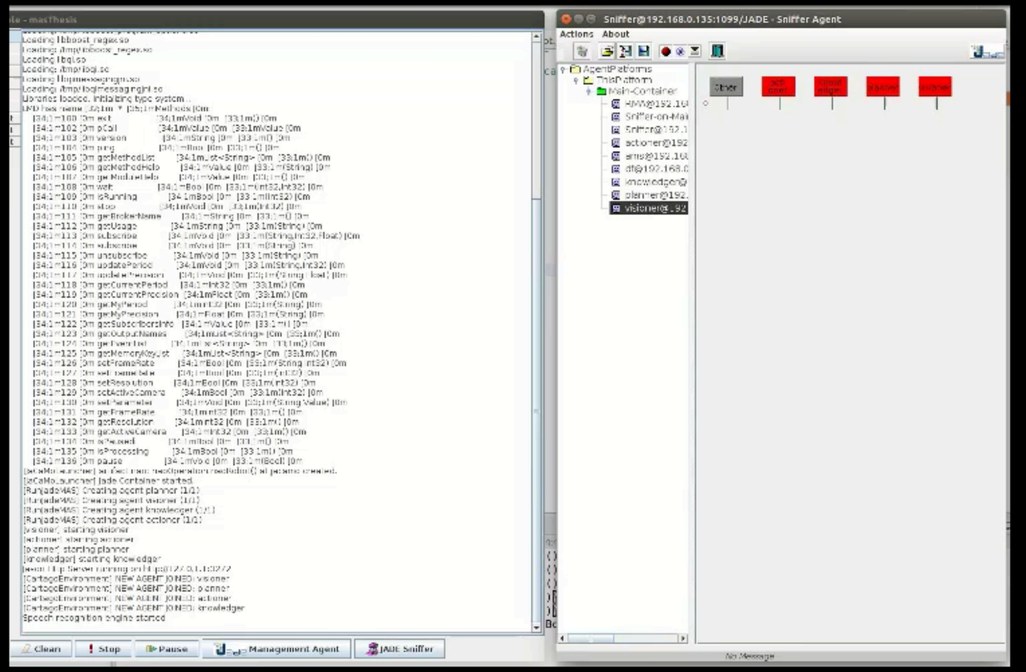
How to model and develop teammate agents (robots) performing trustful interaction with humans?

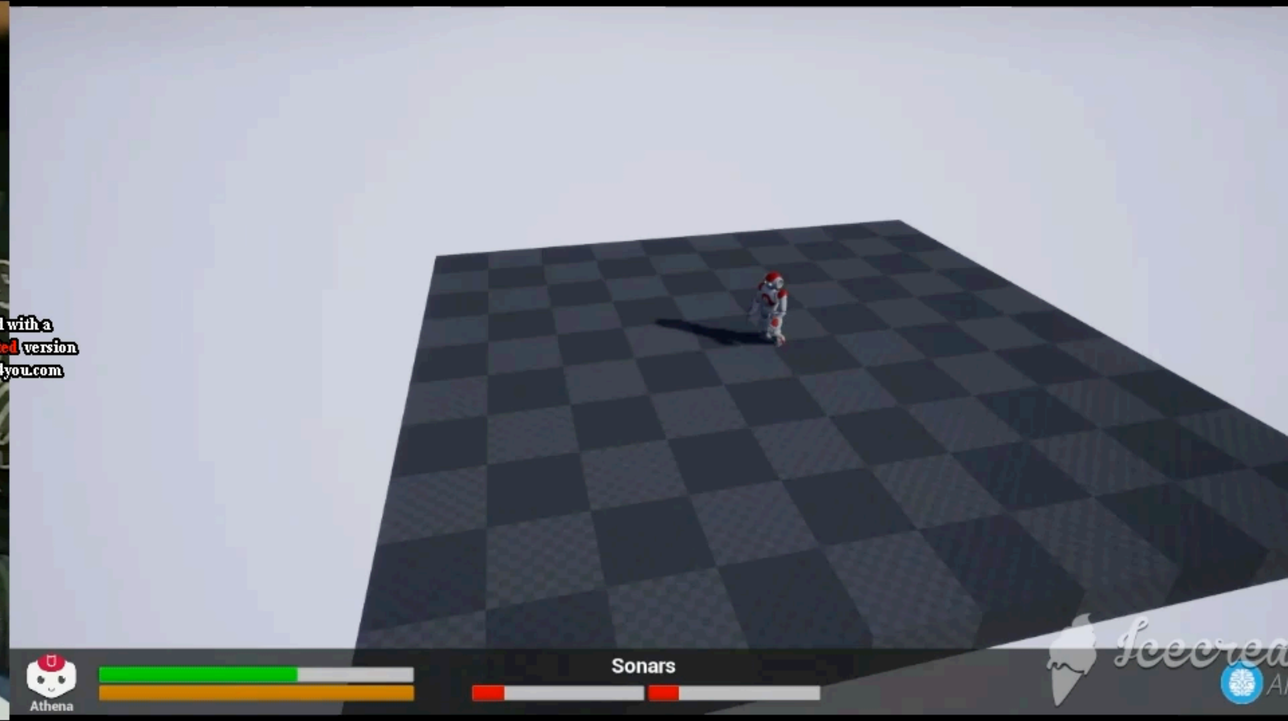
- Deciding and acting in autonomous fashion
- Modelling and representing agents' knowledge
- Explainability -> Trustworthiness





owl:Thing


Sniffer@192.168.0.135:1099/JADE - Sniffer Agent
Actions
About
Clean
Stop
Pause
Management Agent
JADE Sniffer




LIGHTING NEEDS TO BE REBUILT (145 unbuilt object(s))
Athena
Sonars
Icecream APPS

Human-Robot Teaming Interaction

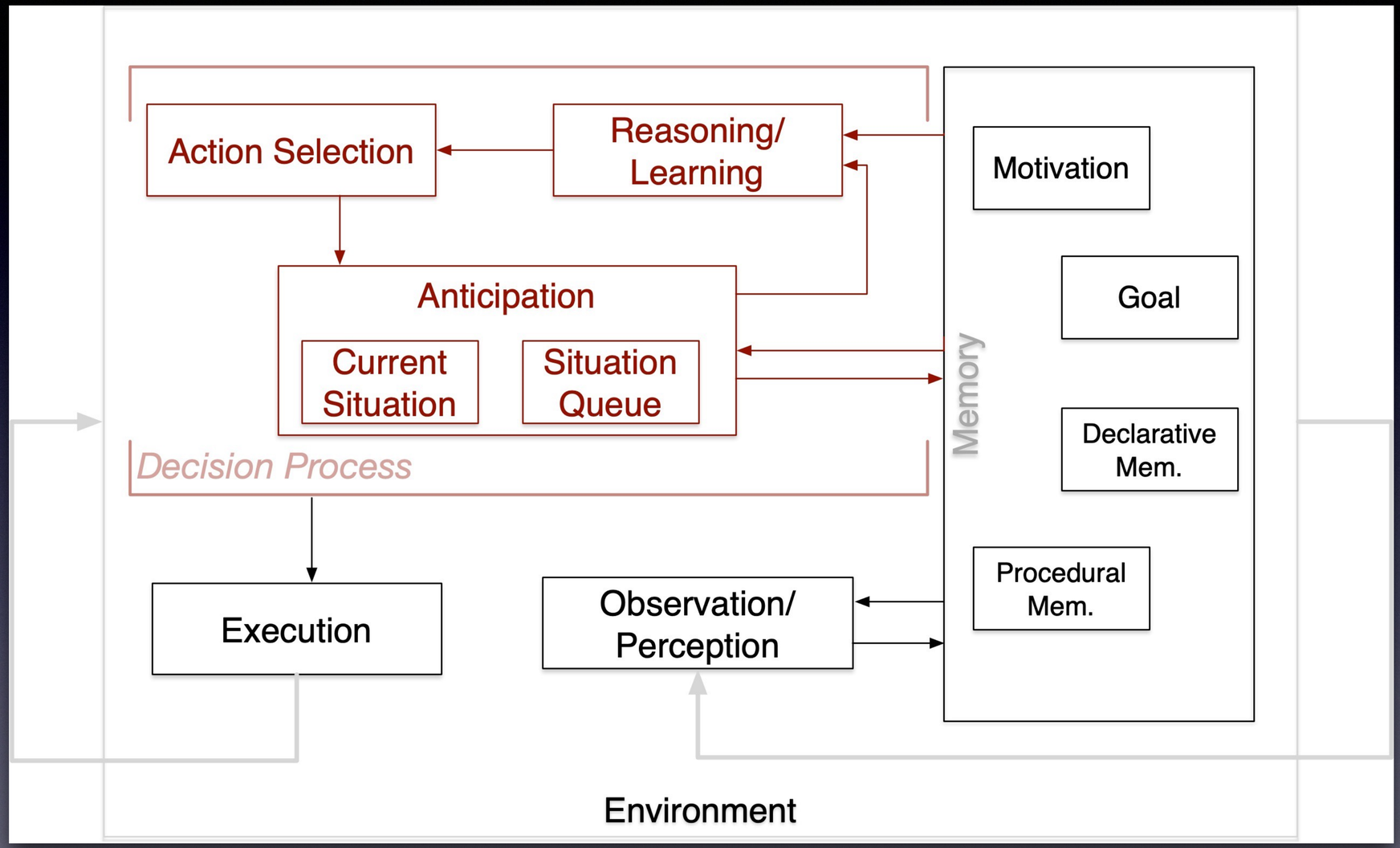
- improve the level of trust from humans to robots in a collaborative task

- manage collaborative tasks
- manage shared knowledge on environment and objectives
- decision making on the base of the other actions
- explainability and trustfulness

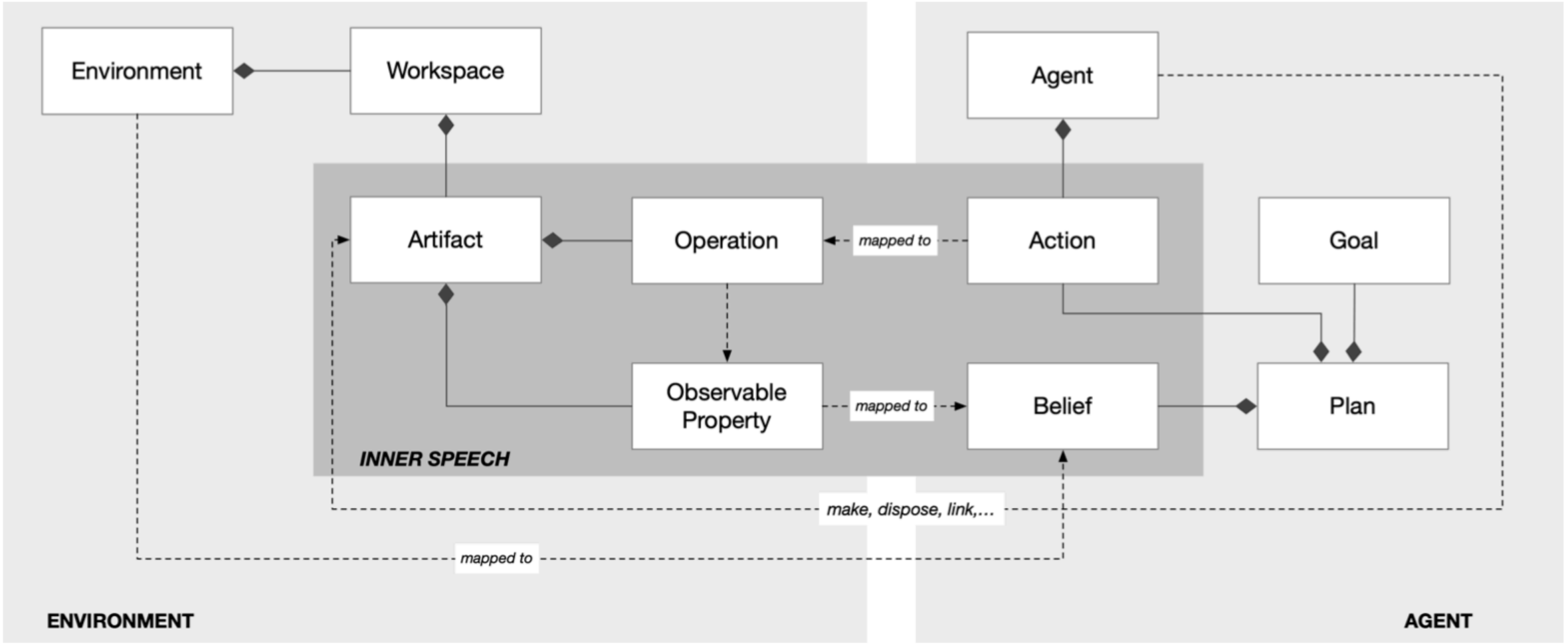
A twofold approach

- Methodological and implementation
  - design abstractions for being trustful and explainable
  - design abstractions for decision process, anticipation and knowledge
- Implementation
  - the question is: which are the paradigms or technical issues allowing to pass from the design to the code and the working system?

Methodological Perspective



Going towards the implementation



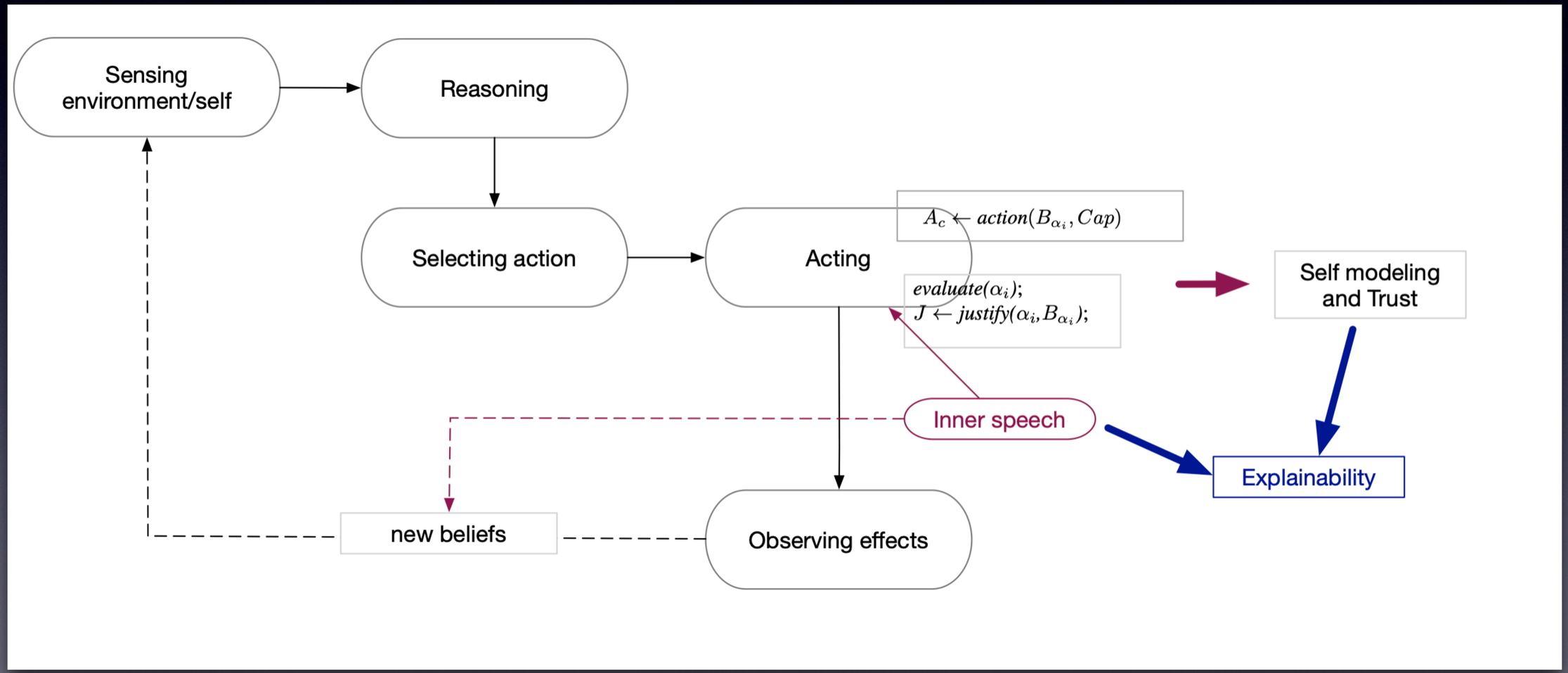
```
1.  B ← B0;          /* B0 are initial beliefs */
2.  I ← I0;          /* I0 are initial intentions */
3.  while true do
4.    get next percept ρ via sensors;
5.    B ← brf(B, ρ);
6.    D ← options(B, I);
7.    I ← filter(B, D, I);
8.    π ← plan(B, I, Ac); /* Ac is the set of actions */
9.    while not (empty(π) or succeeded(I, B) or impossible(I, B)) do
10.     α ← first element of π;
11.     execute(α);
12.     π ← tail of π;
13.     observe environment to get next percept ρ;
14.     B ← brf(B, ρ);
15.     if reconsider(I, B) then
16.       D ← options(B, I);
17.       I ← filter(B, D, I);
18.     end-if
19.     if not sound(π, I, B) then
20.       π ← plan(B, I, Ac)
21.     end-if
22.   end-while
23. end-while
```

$A_c \leftarrow action(B_{\alpha_i}, Cap)$

$evaluate(\alpha_i)$
$J \leftarrow justify(\alpha_i, B_{\alpha_i})$

Self-Modeling
Trustful interaction

Towards the implementation - a proposal



Extending practical reasoning with emergent inner speech

foreach α_i **do**

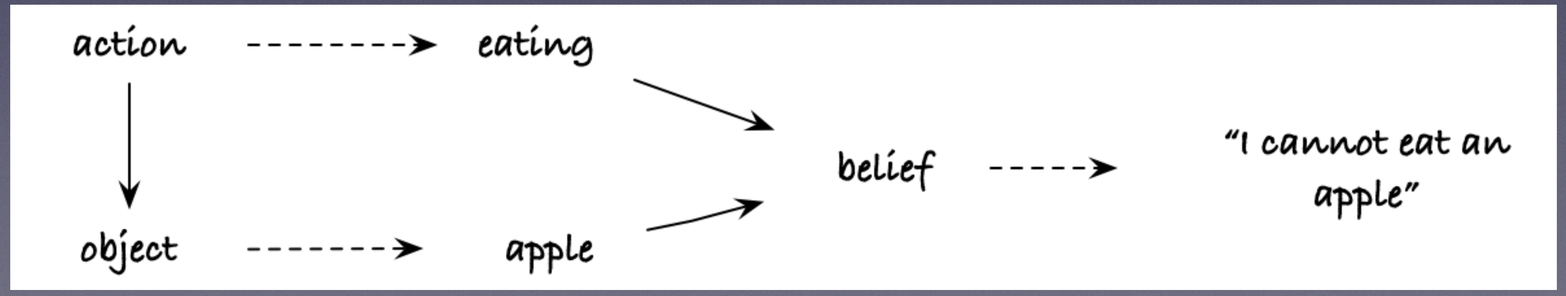
$evaluate(\alpha_i)$;

$R \leftarrow rehearsal(\alpha_i, B_{\alpha_i}, D)$;

$update(B, D)$;

$J \leftarrow justify(\alpha_i, B_{\alpha_i})$;

end



Inner Speech

- A concept used and developed in psychology
- Self-awareness and self-consciousness
  - *the ability to become the object of one's attention*
  - mental states
- The subjective experience of language in relation to one's action feeling and experiences
- Important role in self-regulation, self-direction, problem solving, planning and memory

Technological Perspective

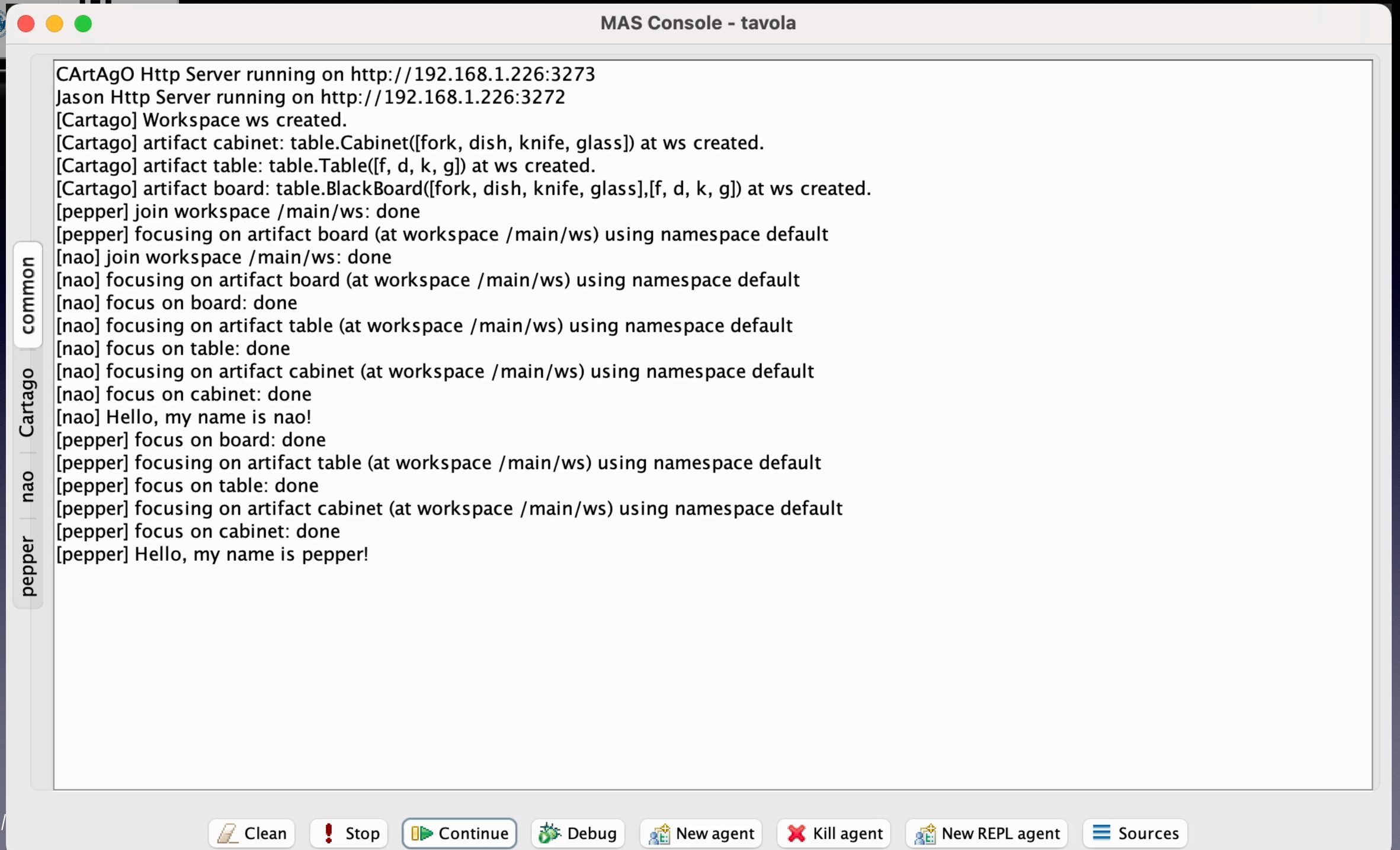
- BDI paradigm and JASON - the obvious implementation counterpart
- JaCaMo
- At runtime a BDI agent
  - Belief, Plan, Intention, Event, Action
  - Selection Functions: S_E, S_O, S_I
- The idea is to link internal events with **speech acts** to generate explanation

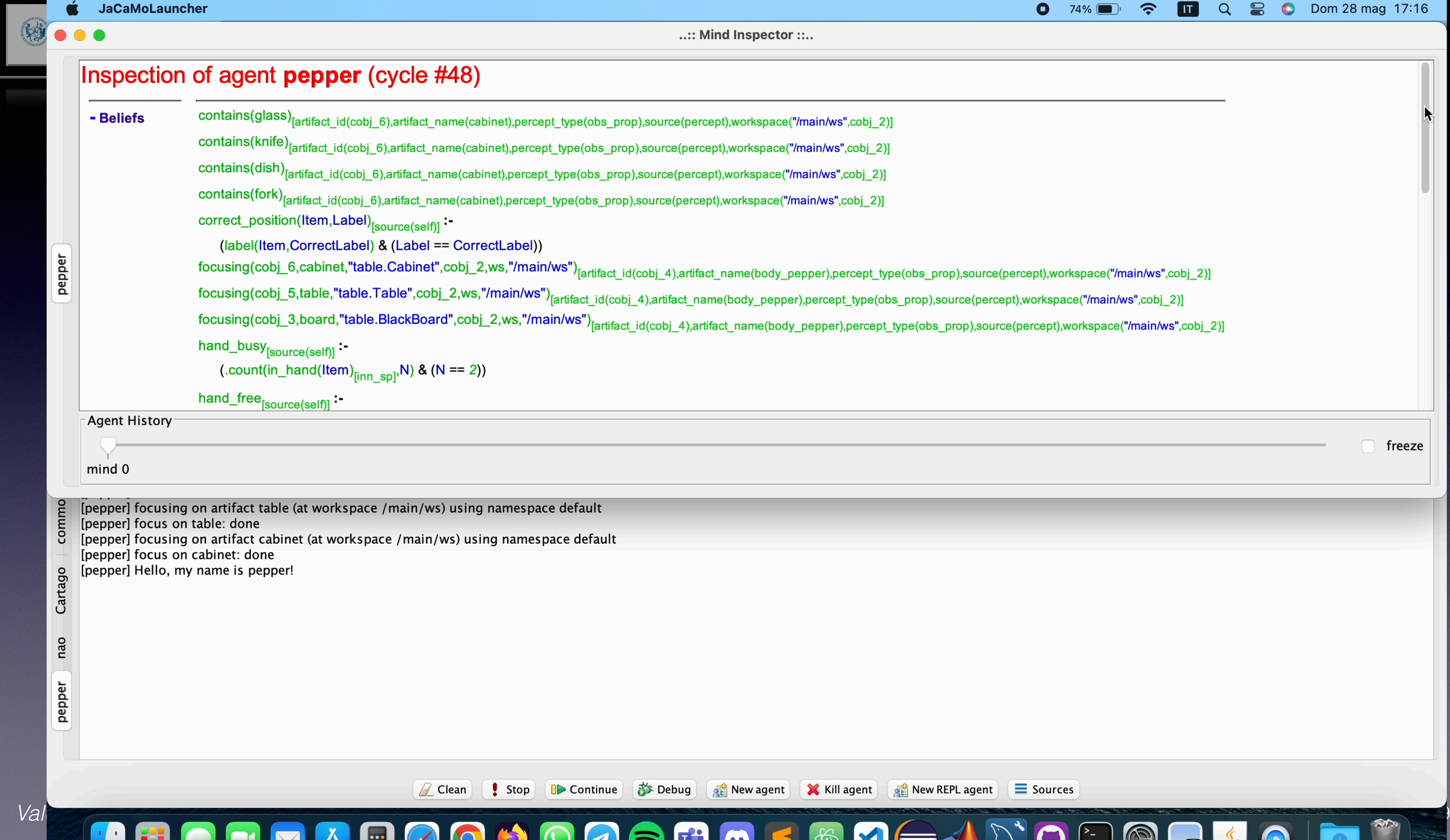
Extending practical reasoning with inner speech

- In MASs communication is based on the theory of speech acts
- Inner speech is a way of thinking about oneself -> under certain condition a speech act can address the agent itself
  - Plan
    - actions to achieve team goals
    - message for changing beliefs or goals
    - making the message available outside

Speech act and setting a table

Agent	**Responsibility**
pepper	Leader: it initiates the activities to set the table and, when all the tasks are done, checks that all the objects are in the right place to see if the goal has been achieved;
nao	It waits for a message from the leader to start performing the tasks indicated on the blackboard artifact and to cooperate with him;
Workspace	**Description**
room	This is the predefined workspace provided by CArtAgO and represents the space where the two agents are involved in setting the table by pointing at the cointainer;
Artefact	**Description**
blackboard	It contains the list of tasks to be done by the agents and information about the correct position of the objects on the table;
table	It contains predefined positions in the form of labels where objects can be placed. Designed for agents that perform simple actions, such as placing or picking up an object;
cabinet	It is a simple artifact that serves as an initial container for the various items placed on the table by the agents;


MAS Console - tavola
CArtAgO Http Server running on http://192.168.1.226:3273
Jason Http Server running on http://192.168.1.226:3272
[Cartago] Workspace ws created.
[Cartago] artifact cabinet: table.Cabinet([fork, dish, knife, glass]) at ws created.
[Cartago] artifact table: table.Table([f, d, k, g]) at ws created.
[Cartago] artifact board: table.BlackBoard([fork, dish, knife, glass],[f, d, k, g]) at ws created.
[pepper] join workspace /main/ws: done
[pepper] focusing on artifact board (at workspace /main/ws) using namespace default
[nao] join workspace /main/ws: done
[nao] focusing on artifact board (at workspace /main/ws) using namespace default
[nao] focus on board: done
[nao] focusing on artifact table (at workspace /main/ws) using namespace default
[nao] focus on table: done
[nao] focusing on artifact cabinet (at workspace /main/ws) using namespace default
[nao] focus on cabinet: done
[nao] Hello, my name is nao!
[pepper] focus on board: done
[pepper] focusing on artifact table (at workspace /main/ws) using namespace default
[pepper] focus on table: done
[pepper] focusing on artifact cabinet (at workspace /main/ws) using namespace default
[pepper] focus on cabinet: done
[pepper] Hello, my name is pepper!
common
Cartago
nao
pepper
Clean
Stop
Continue
Debug
New agent
Kill agent
New REPL agent
Sources


JaCaMoLauncher
74%
IT
Dom 28 mag 17:16
..:: Mind Inspector ::..
Inspection of agent pepper (cycle #48)
- Beliefs
contains(glass)[artifact_id(cobj_6),artifact_name(cabinet),percept_type(obs_prop),source(percept),workspace("/main/ws",cobj_2)]
contains(knife)[artifact_id(cobj_6),artifact_name(cabinet),percept_type(obs_prop),source(percept),workspace("/main/ws",cobj_2)]
contains(dish)[artifact_id(cobj_6),artifact_name(cabinet),percept_type(obs_prop),source(percept),workspace("/main/ws",cobj_2)]
contains(fork)[artifact_id(cobj_6),artifact_name(cabinet),percept_type(obs_prop),source(percept),workspace("/main/ws",cobj_2)]
correct_position(Item,Label)[source(self)] :-
(label(Item,CorrectLabel) & (Label == CorrectLabel))
focusing(cobj_6,cabinet,"table.Cabinet",cobj_2,ws,"/main/ws")[artifact_id(cobj_4),artifact_name(body_pepper),percept_type(obs_prop),source(percept),workspace("/main/ws",cobj_2)]
focusing(cobj_5,table,"table.Table",cobj_2,ws,"/main/ws")[artifact_id(cobj_4),artifact_name(body_pepper),percept_type(obs_prop),source(percept),workspace("/main/ws",cobj_2)]
focusing(cobj_3,board,"table.BlackBoard",cobj_2,ws,"/main/ws")[artifact_id(cobj_4),artifact_name(body_pepper),percept_type(obs_prop),source(percept),workspace("/main/ws",cobj_2)]
hand_busy[source(self)] :-
(.count(in_hand(Item)[inn_sp],N) & (N == 2))
hand_free[source(self)] :-
pepper
Agent History
freeze
mind 0
commo
Cartago
nao
pepper
[pepper] focusing on artifact table (at workspace /main/ws) using namespace default
[pepper] focus on table: done
[pepper] focusing on artifact cabinet (at workspace /main/ws) using namespace default
[pepper] focus on cabinet: done
[pepper] Hello, my name is pepper!
Clean
Stop
Continue
Debug
New agent
Kill agent
New REPL agent
Sources

Conclusions and Remarks

- We are working on a possible solution to endow agents (robots and intelligent systems) with the ability to explain its actions
  - creating agents that are reliable, explainable and believable
- The key is the deliberation made before taking an action
- We experimented the BDI technology and Jason/JaCa for validating our approach
  - simplicity in handling abstractions
  - significant effort to learn and use Jason for not skilled users
- Abstractions and requirements need to be carefully analysed and transformed into Jason elements

Conclusions and Remarks

- Need for accuracy in the agent knowledge representation
- Our last experiment is on a very simple scenario
  - in the immediate future -> connect actions to beliefs
- Only the part of the work related to the explainability has been presented
  - decision making process will use the same logic

Conclusions and Remarks

- Further validating the approach in a more complex scenario and using more agents
- We will refine the methodological approach
- We will also add ethical aspects to the agents' knowledge
  - to analyse how ethical reasoning may influence the human

Thanks for your attention

valeria.seidita@unipa.it