Inside the Robot's Mind During Human-Robot Interaction

Francesco Lanza[1][0000−0003−4382−6366], Valeria Seidita[1,2][0000−0002−0601−6914], Cristina Diliberto[1], Paolo Zanardi[1], and Antonio Chella[1,2][0000−0002−8625−708X]

[1] Dipartimento di Ingengneria, Università degli Studi di Palermo
[2] C.N.R., Istituto di Calcolo e Reti ad Alte Prestazioni, Palermo, Italy
{francesco.lanza,valeria.seidita,antonio.chella}@unipa.it
{cristina.diliberto01,paolo.zanardi}@community.unipa.it

Abstract. Humans and robots collaborating and cooperating for pursuing a shared objective need to rely on the other for carrying out an effective decision process and for updating knowledge when necessary in a dynamic environment. Robots have to behave as they were human teammates. To model the cognitive process of robots during the interaction, we developed a cognitive architecture that we implemented employing the BDI (belief, desire, intention) agent paradigm. In this paper, we focus on how to let the robot show to the human its reasoning process and how its knowledge on the work environment grows. We realized a framework whose heart is a simulator that serves the human as a window on the robot's mind.

Keywords: Cognitive Architecture, Agent Reasoning Cycle, Decision Process, Human Robot Interaction

1 Introduction

Robots are increasingly present in our everyday lives. We are heading towards and hoping for a reality in which robots cooperate and collaborate with human beings by exhibiting autonomous behaviors as if they were human beings. In this scenario, to enhance interaction, humans feel the need to understand and check what robots are going to do. In other words, the need to look at the robot's mind is spreading.

For this purpose, we are working on the development of a framework, including a simulator, for looking inside the robot's mind. The simulator's functionalities allow having a view on *(i)* the robot's knowledge base which changes at runtime and *(ii)* the robot's ability to generate anticipations of its own actions. In this way, during the interaction we achieve two objectives, to give the human the awareness of what the robot is doing and give the robot the ability to re-plan at runtime even using plans that have not been pre-set during the design phase.

The first functionality serves because robots and humans operate in a dynamic context and the robot cannot be provided with a representation/model of knowledge about its environment that appears totally exhaustive. We must, in fact, take into account that environment changes during execution. The second functionality serves because it is necessary for the robot to be able to re-plan by choosing from a list of useful plans to achieve the goal, shared with the human, or a sub-goal thereof. The decision-making process and the choice are conveyed by the ability to anticipate the scene and then to compare it with the expected results.

The framework we propose represents the implementation of a cognitive architecture that we developed to model the computational cognitive processes of a robot operating in unknown environments and in a team with humans. Indeed, some challenges the Human-Robot Interaction (HRI) research field are facing concern *(i)* knowledge acquisition and representation, including memory management; *(ii)* representation of the external environment; *(iii)* plans selection and creation; *(iv)* learning. Cognitive architectures are a good means for meeting these challenges, they allow modeling the human-robot interactions through the classical perception-action cycle of a cognitive agent. In the literature, a broad set of cognitive architectures exist and we took inspiration from them for creating our one. Our cognitive architecture expands the classical perception-action cycle of a cognitive agent with the modules for the representation of the internal state of the robot and the representation of knowledge about the internal state of the others in the world. This is a way for integrating self-modeling and the theory of mind in team interaction and also for including the whole set of mental states that are typical in the human, therefore emotions, levels of trust in oneself and the others, etc.

In this paper, we illustrate the first prototype of the real implementation of the cognitive architecture shown in [7, 10], which allows us creating robotic systems interacting with humans. The cognitive architecture underpinning the robotic systems lets the interaction happen in a human-like fashion. Moreover, the features of the framework enrich the interaction by realizing the self-modeling abilities of the robot and by allowing the human being aware of the robot's behavior.

The rest of the paper is organized as follows: in section 2 we give an overview on our previous work mainly the cognitive architecture we developed; in section 3 we detail the proposed framework and in section 4 we give some working example; finally in section 5 we draw some conclusions and future works.

2 Cognitive Architecture and HRI: Towards a Window on the Robot's Mind

The application scenario we consider in our work is that of robots and humans that have to cooperate and collaborate to reach a common objective. It is like a teamwork domain. Teams composed by humans, in whatever domain, follow some precise, and we may say somewhat instinctive, procedures to pursue a shared goal. Main ingredients of human-human interactions are the knowledge each human has on the surrounding environment, on himself and on the others and the ability to select an action to perform. Normally, an action is selected not because someone has said what to do but because he is able to select the right action among a known set or to generate new useful ones. Our work deals with reporting human-human behavior in the human-robot interaction context.

In this context, several challenges exist if humans and robots work in a dynamic and partially known environment. If everything is known at design time, that is:

- the environment and all the objects it is composed of;
- the common goal and its potential subdivision in sub-goals;
- all the actions the robot can perform
- the tight mapping between actions and sub-goals;
- which actions to assign to human and which to the robot;
- all the possible changes of state in the environment as a result of robot action or of human action;

then the robot's mission may be designed in details and assigned without the risk that something may go wrong during execution.

Instead, as it often happens, if the environment is not a-priori known and it changes as the result of the robot and the human actions on it, we need to equip the robot with some kind of human abilities to learn, to self-adapt and to choose the best action to perform among a set of actions. Also, the robot has to be endowed with the ability to adopt or delegate an action to the human on the base of its knowledge. Adoption or delegation is the result of a decision process that in the human is triggered by knowledge on himself and the other, by some particular mental states such as beliefs, desires, intentions and also emotions, stress, trust and so on.

To face these challenges, we decided to represent the structure of the robot's mind by employing a cognitive architecture. Cognitive architectures are useful means for representing the cognitive perception-action cycle, agents, and its decision process. At the same time, we are exploring the possibility to use the robot's self-modeling abilities and its theory of mind to trigger the decision process.

Our hypothesis is based on the observation of humans while working in a team. Humans frequently commit a purpose if they are aware of having the abilities (physical or not) to perform all the helpful actions for pursuing the objective. Also, they continuously re-plan if they do not succeed in doing something and ask other humans for acquiring new useful knowledge. They observe and analyze themselves and the others to know what they can do, what the other is able to do and, at the same time, is going to do. The first part of the behavior exploits the human's ability to create a model of self where elements such as own capabilities and inner states (emotional state, etc.) are represented. The second part exploits the human's theory of mind, hence the ability to attribute to the other some mental states.

We studied several cognitive architectures in literature, the more interesting for implementing our hypothesis are: CLARION, SOAR, ACT-R and LIDA [2, 12, 15, 20]. In addition, we based on the standard model of the mind [14] and hypothesized a set of modules of a cognitive architecture for a robot working in a team: the *knowledge* and *memory module*, the *perception module*, the *communication system* and the *reasoner* that allows the robot to choose actions by taking into account the recovered data. The robot's behavior is decided by the *planner* module that interacts with the context in which the robot is immersed. We, therefore, drew inspiration from the standard model to create an agent-oriented architecture for the acquisition and the representation of the robot's knowledge [10, 7]. We decided to employ the Belief-Desire-Intention agent paradigm [17, 13]. In our experience agents are the most efficient tool for implementing the theories underpinning cognitive architectures into a real working implementation [8, 9].

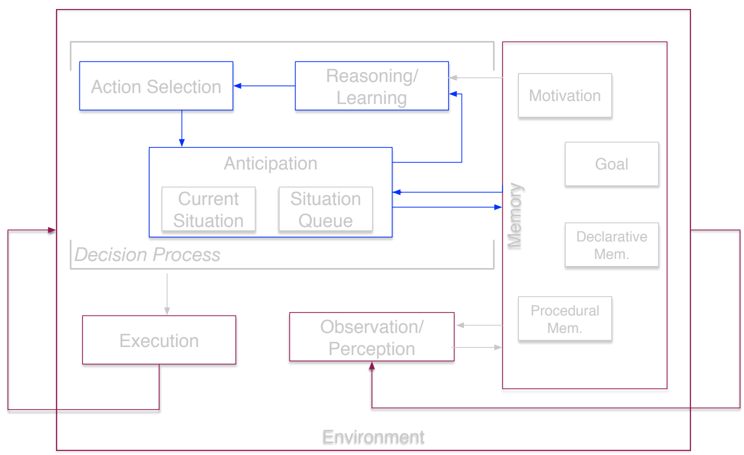

Fig. 1. Architecture for Human Robot Teaming Interaction

Fig. 1 shows the theoretical cognitive architecture we proposed in [10, 11] to implement human-robot teaming interaction.

The architecture presents a basic action-perception loop involving the environment. The loop was enriched with some modules realizing decision functionalities. Also, it is based on the hypothesis that the environment is not composed of all that is out of the robot but also of its inner world. Indeed, in the Memory module we inserted the Motivation and the Goal modules. So, the Memory module represents all the knowledge the robot possesses on the environment, the shared goals and itself in terms of beliefs, emotions, level of trust and so on.

We claim that all these elements are the trigger elements for activating the robot's decision process. The Observation/Perception and the Execution are the standard modules for interacting with the environment. Reasoning and Action Selection realize the cognitive ability to select an action after the reasoning process, which takes inputs from Memory and the new inserted module, the Anticipation.

The Decision Process part of the architecture is centered on Anticipation and Motivation. The robot acts after the reasoning process based on a certain data stored in the memory, but also and mainly after evaluating the anticipation of its actions.

Anticipation allows the robot to imagine the result of its action and to compare it with the situation desired by the post-condition of a goal. The result of the action is given in the form of current situation and situation queue. During the anticipation process, the current situation is generated; it represents the state of the world corresponding to the currently selected action.

The current situation is elaborated on the basis of motivations, goals and all those elements that are in the memory thus getting the execution launched. A queue of possible situations is also created, intended as a set of pre-conditions, objectives and knowledge to achieve them, and post-conditions on the objectives. The robot can draw on all these elements at any time to respond to changes and still maintain its initial target.

The Motivation module is the one triggering the anticipation and the action selection. The process is similar to what we have adopted in other work with the robot NAO for the implementation of what we called perception loop in previous works [19, 18]. Roughly speaking, the robot is designed for carrying out a mission. During the mission execution, in background, the robot imagines the mission and compares its results with the prescribed one. If something differs, it stops the mission, acquires new knowledge and selects another action, or task, to pursue the same objective. The Motivation module is the one triggering the anticipation and the action selection.

During the reasoning and decision making process, the robot continuously interacts with the environment and the human to improve and increase the knowledge base.

A cognitive architecture shapes and represents the cognitive processes of the robot. We are in a context of dynamic interaction in which the decision-making process is triggered by the mental states of the robot and its model of self and the world around. In this paper, we restrict our interest in the cognitive processes related to *(i)* the representation of the knowledge of the robot and its modification in the execution phase and *(ii)* the ability to find independently the plan to be pursued to achieve an objective. Hence, the anticipation (blue colored part of Fig. 1) and the observation/perception of the environment (red part of Fig. 1).

These two parts of the cognitive architecture have been implemented through a framework that includes a simulation software. Software runs in background with the development of the human-robot interaction and acts as a window on the robot. Through the simulator, the robot imagines the development of the action it has selected and re-plans if it does not give the desired result. At the same time, the software shows the knowledge the robot has and how it grows over time. In the next section, we detail how we implemented these parts of the cognitive process.

3 Inside the Robot Mind: a Simulation Environment to Access Robot's Cognitive Processes

To effectively put in place the modules described at the end of the previous section, the idea we propose is to develop a framework in which the anticipation is implemented by means of a simulator and the knowledge representation through some gaming graphic elements. Each graphic element useful for the interaction has a direct mapping with the OWL [3] ontology used for modeling robot's knowledge. Moreover, the robotic system is managed by a Belief Desire Intention (BDI) [13, 21] agent system, implemented employing the *JASON* [5, 4] programming language, and the Robotic Operating System (ROS) [16].

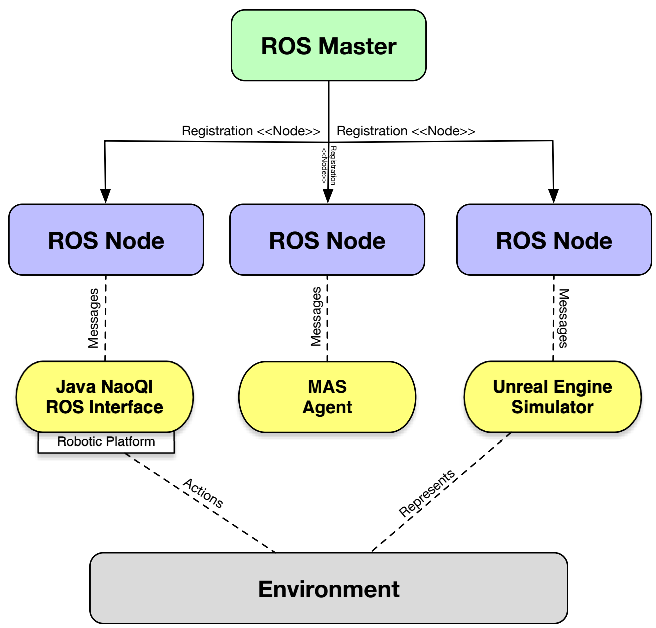

Fig. 2. Human-Robot interaction system software architecture that realizes the cognitive architecture of Fig. 1

We decided to use the Nao robot by Softbank Robotics but it is worth to note that the choice of the robotic platform does not affect the operation of the framework.

In a first phase, the robotic system is designed and built in a standard way, it is provided with a knowledge base on the environment in the form of an ontology and the mission that it has to accomplish. Once running, the robot interacts with the simulator and with the human. The robot sends all the elements to the simulator to generate anticipation. The agent system evaluates the result of the comparison and selects the action to be performed. The robot continuously interacts with the human and the environment to acquire new elements in the knowledge base that are then shown by the interface of the simulator itself.

We use Unreal Engine (UE) [1] for the simulator and the Robotic Operating System (ROS) [16] for the development of robotic applications.

The simulation process, executed by an unreal engine background process, is merged with the cognitive reasoning cycle of our architecture using the ROS publisher/subscriber architecture, as shown in Fig. 2. On the left branch of the scheme, a ROS node is registered into the ROS architecture to handle the robot through the Java NaoQI library. This node lets accessing that robot part including motion, vision and sensory modules and memory handler. On the right branch of the scheme, another node is registered into the ROS architecture to handle the Unreal Engine simulator. More details about the integration between ROS and UE are defined below. On the center branch of the scheme, the last node is registered to handle the communication through the multi-agent system, or whatever kind of implementation for the robotic system one wants to choose.

Mainly, on a high-level perspective, the framework is composed of three main components: ROS, Unreal Engine 4 and a plugin for Unreal Engine 4, called ROSIntegration[3].

ROS (Robot Operating System) is an open source framework designed to encourage the development of robot applications by providing tools that allow communication between different systems and the reuse of code in robotic development. Unreal Engine 4 is a graphic engine, mainly used to develop modern video games, utterly programmable in C++. It provides a lot of tools such as real-time rendering engine for 3D graphics, physical engines for collision detection, animations support, allowing us to create an environment as realistic as possible. Furthermore, Unreal Engine 4 functionalities can be expanded employing plugins, developed by its community. In our case, we used the ROSIntegration plugin to allow communications between ROS, and then the robot, and Unreal Engine 4.

The powerful and the features of Unreal Engine 4 and ROS let the communication process be very simple and effective:

1. one or more Java applications, named Talker, are responsible for communicating with the robot, by means of a set of libraries: ROSJava. It was developed by ROS community and allows to instantiate the ROS concept (nodes, services and topics) inside the Java code. The Java applications use the NaoQI library, provided by Softbank Robotics, to interface with Nao robot. NaoQI extrapolates information of interest (such as robot temperature, robot joints position, battery level and so on), and write them into ROS topics;
2. the ROSIntegration plugin, installed in the UE4 editor, creates a bridge between ROS and UE4, allowing UE4 to subscribe to the ROS topics created by the Java applications and recover the information from them;

[3] https://github.com/code-iai/ROSIntegration

3. on the UE4 side, one or more C++ classes, called Listeners are responsible for retrieving and elaborate robot data. We use UE4 functionalities to display such information in a friendly graphic manner.

At the time of writing, only communications “from Robot to UE4” have been implemented and tested. However, we designed the architecture in such a way it performs bilateral communications by integrating Talkers in the UE4 Editor and Listeners in the Java applications and thus allowing the users to communicate with the robot through the simulator interface.

4 Looking Inside the Robot: the Framework at Work

After designing and implementing such an architecture, we validated our idea through two different scenarios, aimed to show if and how the simulator might be used to support the interaction between robot and the human. These two scenarios delineate the two characteristics of the framework we are illustrating in this paper, namely: the capability of generating the anticipation of actions and the capability of modeling and showing owns knowledge at runtime. The starting

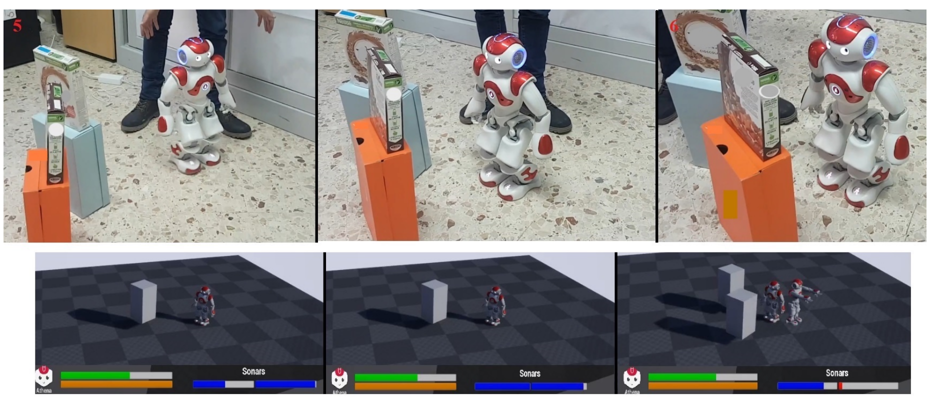

Fig. 3. The robot’s mission seen in the real and the simulated environment.

point is: “Let’s suppose we can equip the robot with a cognitive architecture that allows it to emulate the decision-making process of the human being, and therefore to be able to react adequately and pro-actively in dynamic situations. How can the human be aware of this similarity, and therefore include robot’s limitations and strengths as components of the decision-making process that leads to the creation of sub-goals aimed at achieving the final goal?” In a nutshell: how can a human treat the robot like any other team member, considering it as a resource for achieving the common goal?

Our scenarios are very simple. However, the conclusions drawn from such tests support the idea that a simulation software aimed to display the robot’s mental processes would be an effective intermediary between the robot and the human.

First scenario: generating the anticipation. The first scenario is a simple path finder: using a navigation algorithm, we used the simulator to show the robot's decision-making process in finding the best path to reach a destination goal in a changing environment. So, the mission to design is to reach a specific position. The working environment is made of the human, the robot and one obstacle.

The robot is designed for performing the mission and is equipped with an essential ontology representing the working environment. As shown in Fig. 3, the simulation environment allows the human to be aware of what the robot knows about the surrounding environment, at the beginning of the mission and during its execution. Fig. 3 has two parts, the part above shows the real world in which the robot moves while the part below shows the interface of the simulator, i.e. the robot's mind. As it can be seen, going from left to right:

1. at the beginning of the mission execution, the robot is aware of its position and the goal's position. An avatar appears in the simulation environment. The avatar represents a robot's mental extension, the image it has of itself in the environment. The avatar executes the designed path to reach the goal (the middle couple of figures). It realizes the anticipation of the scene;
2. at the end of the simulation/anticipation the robot starts to execute the path;
3. during the execution of the task, the environment could suddenly change or may be different from the one designed in the initial ontology. Indeed, in this scenario, we put a second obstacle without inserting it in the ontology. When perceiving the new situation, the robot stops its navigation. The left ROS node (Fig. 2) and the NaoQI interfaces communicate with the ROS master for handling the new situation. Through the right ROS node the obstacle appears in the simulation environment and the robot recomputes the best path. Then, the new anticipation is showed by means of its avatar (right part of Fig.3);
4. the algorithm proceeds in such a way until the robot reaches the goal or until there are no more available paths to follow.

In this first scenario, we limited to describing only the part of the generation of the current situation and we left out the situation queue generation. We preferred to make the description simpler to better illustrate the validation results and the considerations we make below in the following paragraph.

Remarks on the first scenario. Software usefulness is evident when the robot behaves differently from expectations, i.e. when the robot cannot reach the goal although there is an available path. In this application, the human inserts the obstacles in the environment, so he knows if the robot could reach the goal or not. If there is an available path, why can't the robot use it to reach the

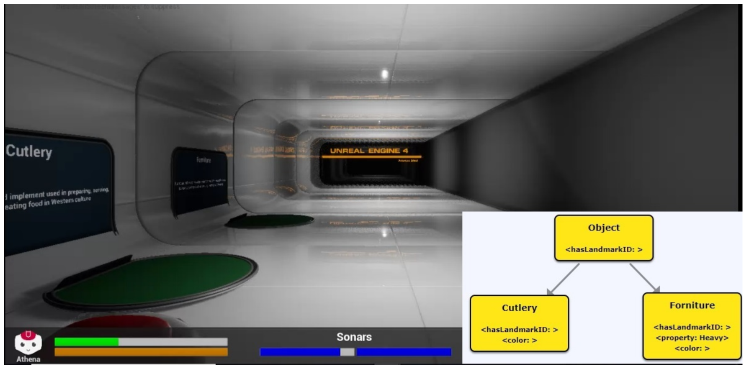

Fig. 4. The simulator interface showing a view of the robot's mind and how it represents its knowledge

goal? The answer is that the inaccuracy, albeit minimal, of the rotations aimed at circumventing the obstacle has caused an incorrect reading of the sensors, bringing the robot to identify the same obstacle it was trying to avoid as a new one, and thus blocking the actually available path. For the human being, identifying the correspondence between the unusual behavior of the robot and the imprecision of the rotations is a non-trivial task, especially if this inaccuracy is not evident.

Using the simulation software as a tool representing what the robot knows about the surrounding environment was crucial: incorrect readings of the sensors result in the simulation environment, with the positioning of the first obstacle correctly, and the appearance of a second obstacle, not expected by a human, in front of the robot after the rotation. In this case, the software allowed the human to understand not only that the robot represented the world incorrectly, but also to connect this error to the reasons that led the robot to conclude that there were no paths available to reach the goal. In a nutshell, the human was able to understand the motivations behind the robot's behavior.

The second scenario: representing changing knowledge. In the second scenario, a robot works with a human in transporting different objects from a starting position to a destination. The final destination of the object is indicated by the human and the robot is designed accordingly. The robot has knowledge of the environment represented at design time by an ontology, created using OWL. The ontology dynamically grows as soon as it discovers new concepts or an instance of them[4]. The method used for enhancing and enlarging knowledge at runtime is shown in [6].

[4] It is worth to point out that we represent the robot's knowledge through an ontology containing at the same time all the concepts the robot knows and all the objects really present in the environment as instance-of the concepts. For details about that refer to [6, 8, 9].

At the beginning of the mission, the robot's ontology has been designed with three concepts: Object, the main concept, Furniture and Cutlery, sons of Object (Fig. 4). Furniture has an attribute "*heavy*", it indicates and allows the robot reason about that whatever object in the world is instance-of Furniture cannot be transported by the robot. Fig. 4 shows the simulator interface where the previous situation is reported. The interface simultaneously shows the ontology (in the bottom right corner) during the mission execution and its representation on the robot's mind through stratified "mind corridors". The corridors follow the same structure of the ontology.

Also, the interface shows the inner state of the robot, what it knows about itself and the world around. In this way, we realize and prove the self-model ability of the robot to the interacting human.

Corridors are linked to each other through showcases that may be navigated using the simulator interface. For instance, in the Object corridor, two showcases open respectively on the Furniture and the Cutlery corridor. Showcases are used to maintain the relationship between concept and to help during the interaction.

Let's make an example to clarify it (see Fig. 5). During the execution of this second scenario, the robot sees an object it doesn't know and then asks the human information so to add that object to its knowledge base. The object is a Fork, which is then attached to the concept of Cutlery. In the simulation environment, the Fork corridor appears and then the human can navigate the Cutlery showcase and verify that the Fork showcase appears thus having proof that robot "has correclty understood" the element. Moreover, in the Fork corridor, the human finds another showcase, which serves as a view of the real Fork in the world. In the meanwhile, as shown in Fig. 5, in the external view a Fork will appear, representing the Fork in the real world.

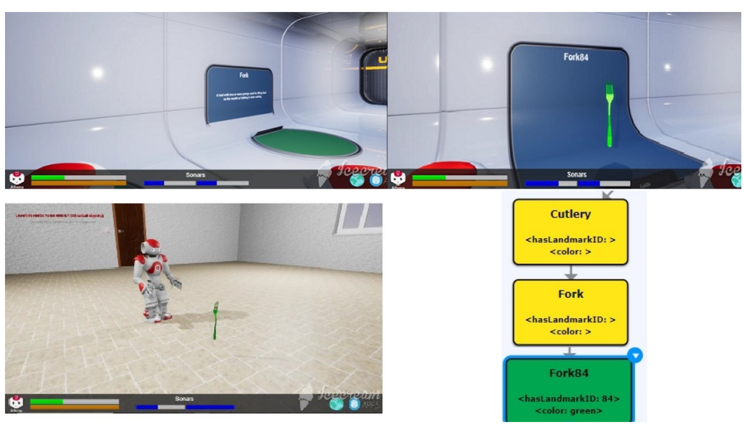

Fig. 5. The simulator interface showing an internal and an external view of the object recognition situation. The object is represented in the robot's knowledge base and in the world as perceived by the robot in relation to itself.

After populating the mind corridors, and thus increasing the robot's knowledge, the execution of the task starts. If the recognized object is a kind of furniture, such as a chair, the robot cannot move it because of the "heavy" property. If the identified object is a kind of CUTLERY, such as a FORK, the robot asks the human where it has to put the object (left or right), executes the task and goes back to the starting position, ready for another iteration.

Remarks on the second scenario. The software effectiveness, like in the previous scenario, became explicit when the robot refused to carry the FORK item. The simulator allowed us to explain this unusual behavior, due to a misinterpretation of the nature of the object: the FORK concept was erroneously interpreted as the son of the FURNITURE concept, making the FORK instance no-transportable. So, we can conclude that human can use software as a continuous feedback on the agent's representation and updating of the surrounding world and its internal conditions. Using the software, a human can:

- verify that the robot knows the item related to the application domain;
- verify that the acquisition of new information is done correctly, making it possible for the human being to delegate tasks to the robotic agent;
- understand the motivations behind the robot behavior, and thus foresee its intentions and update his plans accordingly;
- understand the motivations behind an unexpected behavior, and thus making justifiable such behavior, increasing the human level of trust in the robot.

5 Discussions and Conclusions

Robotic systems are now able to solve very complex problems even in highly dynamic environments. But what if the scenario is the one of human-robot interaction? When robots and humans work in a team where they have to cooperate and collaborate and the behavior and capabilities of one strongly affect those of the other. Especially about the possibility of making reasoned decisions.

In this case, we need to have a tool to support interactions that allow equipping the robot with a decision-making process based on the possibility of adding new elements in its knowledge base and selecting at runtime the best action to do to achieve a common goal. At the same time, this tool must allow the human to look inside the robot's mind to make the interaction more aware.

In this paper, we propose a framework for supporting the interaction among humans and robots. The framework implements the cognitive architecture we established for human-robot teaming interaction [10, 7, 9]. This framework is a first prototype where we focused on the representation of the robot's knowledge, of its inner and outer world, and on the ability to create the anticipation of the mission. Both the two aspects are fundamental in our approach because human-robot interaction happens in a dynamic and unknown environment. The robot continuously perceives new elements to act on and has to enlarge the representation it has of them. Moreover, it must be able to imagine the result of its actions to direct its reasoning process towards the best plan to implement.

All this is possible by employing the cognitive architecture we developed in [10, 7] and that is resumed in section 2.

With the framework illustrated in this paper, we looked more at the point of view of human but at the same time, we validated our cognitive architecture. We also made some consideration of the interaction process. Through some first tests, we can affirm that the quality of interaction increases by using the simulator software that realizes the self-modeling in the robot. This goes in our future direction of including in the interaction also elements of the Theory of Mind from both the sides. From human to robot and vice-versa.

In the future, we plan to provide the software with all the functionalities that instantiate the whole cognitive architecture and to use it in experiments for measuring the level of confidence and trust of humans in robots equipped as said.

References

1. https://www.unrealengine.com.
2. Anderson, J.R., Matessa, M., Lebiere, C.: ACT-R: A theory of higher level cognition and its relation to visual attention. Human-Computer Interaction **12**(4), 439–462 (1997)
3. Bechhofer, S.: Owl: Web ontology language. In: Encyclopedia of database systems, pp. 2008–2009. Springer (2009)
4. Bordini, R.H., Hübner, J.F.: BDI agent programming in agentspeak using JASON. In: International Workshop on Computational Logic in Multi-Agent Systems. pp. 143–164. Springer (2005)
5. Bordini, R.H., Hübner, J.F., Wooldridge, M.: Programming multi-agent systems in AgentSpeak using Jason, vol. 8. John Wiley & Sons (2007)
6. Chella, A., Lanza, F., Pipitone, A., Seidita, V.: Knowledge acquisition through introspection in human-robot cooperation. Biologically Inspired Cognitive Architectures **25**, 1–7 (2018). https://doi.org/10.1016/j.bica.2018.07.016
7. Chella, A., Lanza, F., Pipitone, A., Seidita, V.: Human-robot teaming: Perspective on analysis and implementation issues. vol. 2352, pp. 12–17 (2019)
8. Chella, A., Lanza, F., Seidita, V.: Representing and developing knowledge using Jason, CArtAgO and OWL. In: Proceedings of the 19th Workshop "From Objects to Agents", WOA 2018. vol. 2215, pp. 147–152 (2018)
9. Chella, A., Lanza, F., Seidita, V.: Decision process in human-agent interaction: extending jason reasoning cycle. In: International Workshop on Engineering Multi-Agent Systems (EMAS 2018). Revised Selected Papers.Springer-Verlag, *(in press)*
10. Chella, A., Lanza, F., Seidita, V.: A cognitive architecture for human-robot teaming interaction. In: Proceedings of the 6th International Workshop on Artificial Intelligence and Cognition. Palermo (July 2-4 2018)
11. Chella, A., Lanza, F., Seidita, V.: Human-agent interaction, the system level using JASON (2018)
12. Franklin, S., Madl, T., D'Mello, S., Snaider, J.: Lida: A systems-level architecture for cognition, emotion, and learning. IEEE Transactions on Autonomous Mental Development **6**(1), 19–41 (2014)
13. Georgeff, M., Rao, A.: Rational software agents: from theory to practice. In: Agent technology, pp. 139–160. Springer (1998)

14. Laird, J.E., Lebiere, C., Rosenbloom, P.S.: A standard model of the mind: Toward a common computational framework across artificial intelligence, cognitive science, neuroscience, and robotics. AI Magazine **38**(4) (2017)
15. Laird, J.E., Newell, A., Rosenbloom, P.S.: Soar: An architecture for general intelligence. Artificial intelligence **33**(1), 1–64 (1987)
16. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: Ros: an open-source robot operating system. In: ICRA workshop on open source software. p. 5. Kobe, Japan (2009)
17. Rao, A.S., Georgeff, M.P., et al.: Bdi agents: from theory to practice. In: ICMAS. vol. 95, pp. 312–319 (1995)
18. Seidita, V., Cossentino, M., Chella, A.: Software design of an AGI system based on perception loop. In: 3rd Conference on Artificial General Intelligence (AGI-2010) (2010)
19. Seidita, V., Cossentino, M.: From modeling to implementing the perception loop in self-conscious systems. International Journal of Machine Consciousness **2**(02), 289–306 (2010)
20. Sun, R.: The importance of cognitive architectures: An analysis based on clarion. Journal of Experimental & Theoretical Artificial Intelligence **19**(2), 159–193 (2007)
21. Wooldridge, M., Jennings, N.R.: Intelligent agents: Theory and practice. The knowledge engineering review **10**(2), 115–152 (1995)