DALI: An Architecture for Intelligent Logical Agents

Stefania Costantini and **Arianna Tocchio**
Department of Computer Science, University of L'Aquila
Via Vetoio, 67100 L'Aquila
{costantini,tocchio}@di.univaq.it

Abstract

Many interesting architectures for defining intelligent agents have been proposed in the last years. Logic-based architectures have proved effective for reproducing "intelligent" behavior while staying within a rigorous formal setting. In this paper, we present the DALI multi-agent architecture, a logic framework for defining intelligent agents and multi-agent systems.

Introduction

An architecture can be seen as a unifying and coherent form or structure whose instances exhibit certain behaviors and/or certain properties. According to the most recent studies in neuropsychology and cognitive sciences, our brain can be considered a sharp architecture where 100 billions of neurons work together, each one belonging to a specialized area, for generating the human intelligence.

In conformity with recent studies in neuropsychology and medical cognitive sciences, the frontal lobe of the cortex is responsible of high-level brain function such as thought and action; the cerebellum is the region that plays an important role in the integration of sensory perception and motor output; the limbic system produces the "emotional brain"; the hippocampus is involved in transferring memories from short term to long term. The deep temporal lobes are very important in the storage of past events, while the frontal ones are related to speech production. Finally, according to recent studies, a particular brain area, called the "MrD", is implied in learning activities.

Researchers in Artificial Intelligence and Intelligent Agents since long have tried to understand what intelligence is, and to reproduce the mechanisms of human intelligence. BDI (Belief, Desires, and Intentions) [(Rao & Georgeff 1991), (Novák & Dix 2006)], Layered [(Muller 1997), (Lisetti & Marpaung 2005)], Cognitive [(Laird, Newell, & Rosenbloom 1987), (Byrne 2001)] and Logical [(Hindriks *et al.* 1999), (Jo Alferes, & Pereira 2002), (Kakas *et al.* 2004)] architectures have been some of the results of this long-lasting effort (we apologize with the proposers of the many interesting approaches that we cannot mention for lack of space: the reader may refer to (Tocchio 2005) for a more comprehensive survey). Generally these architectures describe autonomous, reactive, proactive and socially able entities, capable of reaching their goals by means of specific reasoning processes. BDI architectures adopt concepts as beliefs, desires and intentions for simulating the intelligence, while layered architectures interconnect different specialized layers for determining the global intelligent behavior. The cognitive approach explores the fundamental interaction of implicit and explicit cognition as well as cognitive social simulation. Finally logical architectures try to reproduce intelligence through a symbolic representation of the world manipulated by means of inferential processes.

Surprisingly enough, many of these architectures actually reproduce (parts of) of the architecture of the human brain as it is understood at present by specialists. In fact, Figure 1 shows a cognitive model of natural intelligence presented in the paper (Wang *et al.* 2003) and considered to be a synthesis of the brain mechanisms. Sometimes, agent architectures are even more detailed in their description of the *functions* that contribute to producing "intelligence". Clearly however, the structural properties and the highly parallel functioning of the brain areas constituting the "modules" of the brain architecture are far from being reproduced, as they are not even fully understood.

The similarities however suggest that, to some (for now necessarily limited) extent, functionalities of the cortex, the cerebellum, the hippocampus, the lobes and the MrD can be reproduced in computational architectures, and in particular in architectures based on computational logic. Starting from this consideration, in this paper we describe in a non-standard way a logic-based agent architecture: in particular, we organize the description by pointing out the analogies with the brain architecture. Clearly, we are conscious of our limits and we only mean to make the description more appealing to a reader. In particular, we describe the fully-implemented architecture based on the logic language DALI ((Costantini & Tocchio 2002), (Costantini & Tocchio 2004a)).

Can a logical agent which is an instance of the DALI architecture "think"? Maybe this is at present a fairly too ambitious objective. However, to a more limited extent DALI agents can reason, are in principle able to survive in partially-known environments and to perform complex tasks.

Copyright © 2008, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

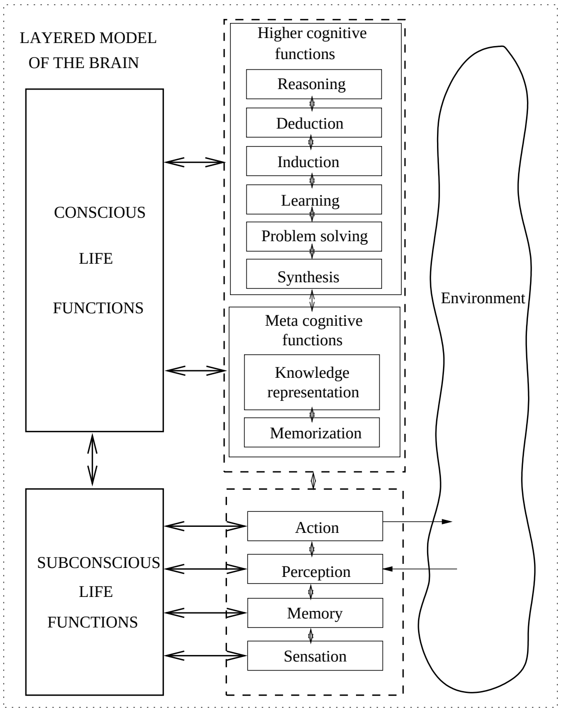

Figure 1: Layered brain architecture

The DALI language has been introduced and discussed under various aspects in (Costantini & Tocchio 2002; 2004a; 2006b; 2005; 2004b). DALI has a full declarative semantics for the single-agent case (Costantini & Tocchio 2006a) where however more work is needed for an effective extension to the multi-agent case. The interpreter underlying the architecture is fully implemented, according to the operational semantics described in (Tocchio 2005).

The DALI architecture

Similarly to the human brain, which is a collection of several specialized areas interconnected among them, agents architectures in general embody different functionalities managed by a global life cycle. Functionalities correspond to either basic or advanced abilities that agents, according to the particular application domain, have to exhibit. It is widely acknowledged that some of these abilities, namely reactivity, proactivity, social ability and learning, are essential components of *intelligence*. The global life cycle interleaves the agent activities, so as to properly balance reactivity and proactivity, as emphasized in (Kowalski 2006). The brain abstract architecture in Figure 1 includes reasoning, learning, induction, deduction, problem solving and synthesis as higher cognitive processes useful for describing the human intelligence. Reasoning layer infers possible causal outputs from given inputs based on known causal relations between a pair of cause and effect proven true. In other words, it manages the human reactive capabilities. Problem solving is a brain cognitive process that searches a solution for a given problem while synthesis combines objects or concepts

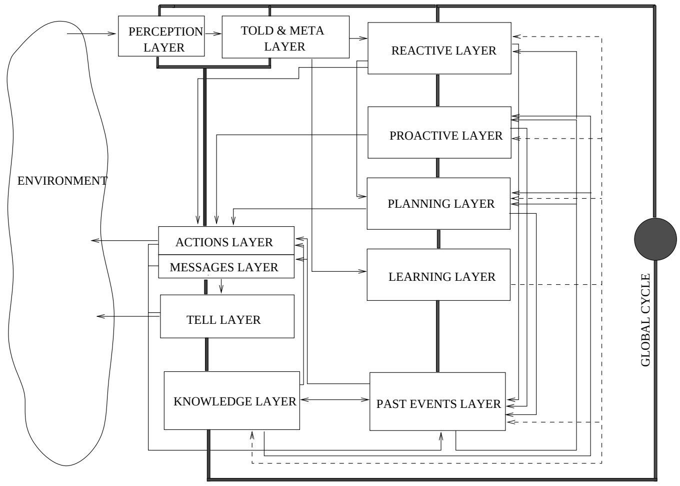

Figure 2: DALI architecture

(Wang *et al.* 2003). Their activity can be considered as a source of the human proactivity. Perceptions coming from the environment trigger several activities, from the reasoning stage where the input is recognized to the learning, induction or deduction stages where the capabilities for interpreting new perceptions are exploited and combined.

The components of the DALI architecture are shown in Figure 2. The perception Layer is the direct interface between an agent and its environment. When a stimulus has been captured by this layer, the TOLD/Meta layer has a role of "filter". TOLD rules perform a selection according to some constraints whose purpose is to discard those perceptions coming from untrusted or unwelcome or irrelevant sources. Perceptions that pass the TOLD check but that the agent is unable to "understand" are processed by the Meta layer. Meta rules employ ontologies for trying to make perceptions understandable. Stimuli which are recognized by the agent proceed to the Reactive Layer, where reaction takes place: some actions or goals are to be undertaken and thus the corresponding layers, Action and Planning, are activated. Past perceptions, goals and actions are recorded by the agent and become past events. Past events are useful for triggering the reasoning activity of the Proactive Layer.

There is a general agreement upon the assumption that intelligence derives from exploiting a combination of different kinds of sources: knowledge, experience, instinct, emotions and so on. The proactive Layer exploits this principle by means of proactive rules whose body may contain in conjunction: facts coming from the Knowledge Layer, past events and the results of logical inference. The agent keeps trace of proactive conclusions by means of past events. The Learning Layer supports the acquisition of new facts and rules, where an agent can either learn from experience or from being told by others (Costantini & Tocchio 2005). Finally, the Tell Layer filters the out-coming messages by means of constraints rules similar to Told ones. In the next sections we explain in some detail reactivity, proactivity and learning features of DALI agents architecture.

Reacting via external events

Reaction is a primary and essential function, strictly related to survival and however to the interaction with the environment. In our brain, sensory information is perceived and interpreted, and then the stimulus elicits a certain learned or reflexive emotional response. As shown in Figure 1, perceptions coming from the environment are interpreted by the higher cognitive functions of the brain. If there is no predefined reaction, our brain presumably tries to find through deduction, induction or learning the most plausible reaction. In logic approaches, observation of the world and action in the world are mediated by a reasoning process that prevents behaviors form being extremely instinctive: a perception coming from the environment is interpreted according to a set of rules and contexts and the correspondingly selected action is the one that will be performed.

Formalizing the environment as the set $E = \{e_1 : t_1, ..., e_n : t_n\}$ of possible environmental states, where t_i is the time in which the perception e_i has been received, we can consider agents as entities capable of perceiving a subset of E, since omniscience is not realistic. In the DALI language the environmental states that an agent perceives are called *external events*. Rules belonging to the Reactive Layer that specify the reaction in response to external events are called *reactive rules*. A reactive rule has the syntax:
$e_i E :> Body$
where postfix E indicates the external event and *Body* is a sequence of actions, goals or atoms. Each external event in the rule head is associated to a context. In this way, reaction happens only if the particular rule defining the context authorizes it. This rule has the syntax: e_i $:<$ $Context\_Decription$ where $Context\_Description$ is a sequence of atoms describing the agent experience and knowledge.

Consider a shipwrecked agent being in a desert island. It has to spy the sea searching a ship for going home and, if a ship appears on the horizon, it has to move the arms for capturing the attention. This can be synthesized by the rule:
$see\_shipE :> move\_armsA, shoutA.$
where the postfix A indicates that $move\_arms$ and $shout$ are actions. Instead, an agent which is on holiday with the family and sees a ship on the horizon will not adopt the same behavior. This is the purpose of the following context rule:
$see\_ship :< i\_am\_shipwrecked, in\_island.$
Only if the agent is in the condition of shipwrecked in an island, the reaction is allowed. After the reaction, the external event is transformed into a *past* one. This means that the agent is capable of remembering that a reaction has taken place and can use the information for further proactive or planning activities.

"Thinking" via internal events

Like humans, agents need not only to react but also to reason about their own internal state. This is a complex activity depending on several factors such as knowledge, experience and reasoning. The Proactive Layer is composed of *internal events* rules. In particular, the inferential capability of this layer is based on pairs of rules that work together. The body of the first one contains invocations to the Past Events Layer, the Knowledge Layer etc. The body of the second one expresses the reaction to the internal conclusion reached by the first one. Body can contain actions, goals and what is expressible in prolog. More precisely, the syntax is:
$i_i : -p_1P, kb_2, p_3P, .., kb_n.$
$i_iI :> Body$
where $n > 0$, i_i is the internal event, p_iP are past events and kb_j are knowledge base facts or atoms. After reaction, each internal event becomes a past event and may occur in the body of the first rule of other internal events. This device supports complex reasoning processes where an internal conclusion can fire another one and so on. The first rule of each internal event is attempted from time to time according to specific directives. Whenever conditions in the body are satisfied, the reaction happens.

Consider again the shipwrecked agent in the island. Reactivity allows it to try to be rescued if a ship is near the island. While waiting for rescue however, it risks to starve. So, it has to search for food and it can eat at most every k hours for sparing the available food. To these aims, internal events can be useful:
$find\_food(X) : -in\_island, not(see\_shipP),$
$\quad have\_fish\_hookP, X = fish.$
$find\_food(X) : -in\_island, not(see\_shipP),$
$\quad see\_coco\_palm, X = coconut.$
$find\_foodI(X) :> take\_foodA(X).$
$i\_am\_hungry(X) : -i\_am\_hungryP(T1), now(T),$
$\quad T - T1 > k, find\_foodP(X).$
$i\_am\_hungryI(X) :> eatA(X).$

We suppose that if our agent has a fish hook then it can go fishing or else, if there is a palm, it can take a coconut. see_ship and $have\_a\_fish\_hook$ have been expressed as past events (denoted by postfix P) because we suppose they record past perceptions. If the agent found some food, the first internal event becomes a past one and is used by the second internal event for drawing the conclusion. In fact, if the agent has food and k hours have elapsed from the last meal, it can eat again. This example shows how different internal events can be related.

Affecting the environment via actions

Human beings not only think but also act. Actions are determined both by reactive and proactive activities and, in general, intelligence can be identified as the process bringing to choice the best action to do. How can the best action be determined? Not only according to the reasoning but also to the context. In fact, if the shipwrecked agent decides to cook a fish, it must be sure to have the possibility of making a fire. Or, the action of swimming is advisable only if no shark is nearby. Preconditions to actions, typical of the human reasoning, can be translated into a logical formalism by means of specific rules. In the DALI language, these rules are called *action rules* and have the following syntax:
$a_i :< Body$
where a_i is an action and Body is a sequence of past events, atoms or beliefs. Actions rules are included in the Action Layer. In Figure 2 it can be seen that this layer exploits

information coming from the Knowledge and Past Events Layers. Also, it updates the Past Events Layer because each action, when performed, becomes a past event. Consider a scenario where our agent observes in the sea a fish. A fish can possibly be dangerous: e.g., a Blue-Ringed Octopus injects a neuromuscular paralyzing venom. Our agent must avoid to get this kind of fish.
$see\_fish(X) :> get\_fishA(X).$
$get\_fishA(X) :< not(dangerous(X)).$
$dangerous(blue\_ringed\_octopus).$
Actions can also be messages. Messages are managed by the Message Layer and can be associated to *actions rules* that specify the preconditions for sending that. Out-coming messages can be subjected to the TELL Layer that filters them according to some constraints involving the addressee and/or the content. A particular kind of internal events is used in the DALI language for coping with easy planning problems. These rules are contained in the Planning Layer and are based on particular kind of events called *goals*. Features of DALI planning mechanisms are explained in (Costantini & Tocchio 2004b).

Remembering via Past Events

Our brain is capable of remembering facts and sensations, successes and errors. In particular, the brain architecture in Figure 1 considers the "memory layer" and the "memorization" process. Memory layer comprises Short Term and Long Term memory. Short Term memory(STM) allows us to record temporary data like a telephone number that we have to use only once, while Long Term memory(LTM) maintains information (reactions, internal conclusions, experiences) that the brain considers relevant. Relevance can be either pre-set by a biological predisposition or established by past repeated experiences. Memorization is a meta cognitive process that retrieves, encodes and stores information in LTM.

As emphasized in previous Sections, the DALI architecture contains a particular Layer dedicated to remember the more relevant activities of the agent life. Reactions, internal conclusions, actions, goals, facts are transformed into past events and stored in the Past Events Layer. Past events are managed in different ways according to their role and their expected validity. In fact, not all past events are remembered forever: particular directives determine how long or until when conditions they have to be kept. Moreover, past events are divided into two sets: P and PNV. The former one contains current "valid" past events that describe the present state of the world as perceived by the agent, while the latter one contains old instances. Past events in PNV may have however a relevant role for the entity decisional process. In fact, an agent could be interested in knowing how often an action has been performed or a particular stimuli has been received by the environment. For instance, consider again the life of our shipwrecked agent. It might for instance infer if a sea area is dangerous according to how often it observed a shark there.

Repetition of past events in PNV can be also exploited for reinforcing the confidence in some facts. If the agent for instance observes for a long period that each *king crab* is big, it can conclude with a certain degree of confidence that all *king crabs* are big. Past events are also useful for checking the correctness of the agent behavior via constraints. We can find a certain similarity between the meta cognitive Memorization process and the management of past events in P and PNV. More information about this DALI language feature can be found in (Costantini & Tocchio 2006b).

Acquiring new knowledge

What does it happen if a perception is not recognized? Both humans as agents have to "overcome" their limits by adopting suitable strategies capable of guaranteeing their survival. In our brain, learning is a cognitive process that gains knowledge of something or acquires skills in some action or practice by updating the cognitive model in LTM. In general, humans apply two strategies for improving their knowledge: individual learning and social learning. In individual learning, humans try to overcome the information lack problem by exploiting their internal capabilities. In social learning, one or more external individuals are involved. Individual learning can involve deduction or induction. Induction, from available inputs examples, tries to elicit a general rule; deduction applies general principles to reach specific conclusions. The learned rules resulting from these processes are provisionally saved in the long-term memory and experimented. Rule whose experimentation has been positive are permanently stored in the long-term memory.

A social learning process is more complex: it involves concepts such as trust, competence, roles and so on. In general, this kind of learning in humans is based on imitation processes. In 1796, for the first time Dawkins introduced the concept of *meme* as a way of propagating knowledge. According to the Oxford Dictionary, meme is an element of behavior or culture passed on either by imitation or by other non-genetic means. Meme could be a melody, an idea, a phrase, a method for, e.g., modeling pots or building arches. Imitation in the social learning is the key factor of many variations in the evolutionary process. In fact, an individual transmitting a meme could be in bad faith or the individual receiving the meme could interpret it in a different way. This potentially harmful behaviors sometimes can generate new abilities and advantageous strategies.

What about agents? Agents usually live in open environments where social learning strategies are possible but also risky. Agents come from different technologies, have different roles and competence. Learning by imitation must be a long process where each meme has to be considered under many points of view: its origin, its initial context and its potential role. I.e., the acquired information must be experimented before being incorporated into the agent's knowledge base. Some of the complex mechanisms of social learning have been reproduced in the DALI architecture (Costantini & Tocchio 2005). In DALI, agents are able to exchange rules for improving their abilities. While retrieving and acquiring rules comparatively easy, the relevant problem to learn a correct information remains. Intelligent agents can have different specializations for different contexts and a learning rules process cannot ignore this. Even agents having the same specializations can adopt behavioral rules that can be mutually inconsistent.

We introduced in each DALI MAS a particular entity, the $yellow\_rules\_agent$, keeping track of the agents specialization and reliability. When an entity needs to learn something, it asks the $yellow\_rules\_agent$ for knowing the names of agents having the required specialization and being reliable. When an agent will decide whether to incorporate the learned rules into its knowledge base, it will notify the $yellow\_rules\_agent$ about its degree of satisfaction. Thus, the reliability rate of the provider agent will be accordingly affected. In a first stage, the learned rules will be added to the agent knowledge base as past events and will be subject to evaluation. This preliminary check will verify some properties as, for example, the syntactic correctness or consistency. Moreover, it avoids that conflicting rules are learnt. Selected rules survived will be used by the agent during its life and their utility and efficiency will be established. According to the evaluation, the acquired rules will be finally either learned or eliminated.

DALI Declarative Semantics in a nutshell

The evolutionary semantics proposed in (Costantini & Tocchio 2006a) has the objective of providing a unifying framework for various approaches to reactive and proactive logic-based agents. This semantics is based upon declaratively modeling the changes inside an agent which are determined by changes in the environment as well as agent's own self-modifications. The key idea is to understand these changes as the result of the application of program-transformation functions. In this view, a program-transformation function is applied upon reception of an event, internal or external to the agent. In fact, the perception of an event affects the program defining the agent: for instance, an event can be stored as a new fact in the program. Similarly, actions which are performed can be recorded as new facts. All the "past" events and actions will constitute the "experience" of the agent.

Recording each event or action or any other change that occurs inside an agent can be semantically interpreted as transforming the agent program into a new program, that may procedurally behave differently than before: e.g., by possibly reacting to the event, or drawing conclusions from past experience. Furthermore, the internal event corresponding to the decision of the agent to undertake an activity triggers a more complex program transformation, resulting in version of the program where the corresponding *intention* is somewhat "loaded" so as to become executable.

Then, every agent will be equipped with an initial program P_0 which, according to these program-transformation steps (each one transforming P_i into P_{i+1}), gives rise to a Program Evolution Sequence $PE = [P_0, ..., P_n]$. The program evolution sequence will have a corresponding Semantic Evolution Sequence $ME = [M_0, ..., M_n]$ where M_i is the semantic account of P_i according to the specific language and the chosen semantics. The couple $\langle PE; ME\rangle$ is called the *Evolutionary Semantics* of the agent program P_{Ag}, corresponding to the particular sequence of changes that has happened. The evolutionary semantics represents the history of an agent without having to introduce the concept of "state".

Various agent languages and formalisms will affect the following key points:

1. When a transition from P_i to P_{i+1} takes place, i.e., which are the external and/or internal factors that determine a change in the agent.
2. Which kind of transformations are performed.
3. Which semantic approach is adopted, i.e., how M_i is obtained from P_i. M_i can be for instance a model or an initial algebra. In general, given a semantics $\mathcal{S}$ we will have $M_i = \mathcal{S}(P_i)$.

A transition from P_i to P_{i+1} can reasonably take place, for instance: when an event happens; when an action is performed; when a new goal is set; upon reception of new knowledge from other agents; in consequence to the decision to accept/reject the new knowledge; in consequence to the agent decision to revise its own knowledge. We say that at stage P_{i+1} of the evolution the agent *has perceived* event ev meaning that the transition from P_i to P_{i+1} has taken place in consequence of the reception of ev. It is reasonable to assume that at the stage P_{i+1} the agent will cope with the event by executing a selected reaction.

In our approach, we assume to perform an *Initialization step* by which the program P_{Ag}, written by the programmer, is transformed into a corresponding initial program P_0 via some sort of knowledge compilation. This compilation can on one extreme do nothing, while on the other extreme it can perform complex transformations by producing "code" that implements language features in the underlying logical formalism. P_0 can be simply a program (logical theory) or can have additional information associated to it.

The evolutionary semantics allows properties of an agent behavior to be formally proved (e.g., by induction) and potentially enable model-checking techniques to be employed for verifying that certain states are finally reached. However, the extension to the multi-agent case is not trivial and is still work in progress, as coping in a general way with possible interactions is far from easy.

DALI practical applications: a case-study

The DALI architecture has been fully implemented and has been experimented in real-world applications. In this section, we report about a recent application that has been developed in the context of the CUSPIS European Project[1]. An Ambient Intelligence scenario has been built, where DALI agents have been put at work at Villa Adriana in Tivoli, a fascinating and enormous archaeological area.

The multi-agent system that we have designed and implemented in this project is called DALICA. The main goal of the DALICA MAS system is that of supporting visitors of

[1]The work reported in this section has been partially supported by the project CUSPIS (GJU/05/2412/CTR/CUSPIS) "A Cultural Heritage Space Identification System". The CUSPIS project has been co-funded by Galileo Join Undertaking User Segment call 1A. Since Galileo satellites will be operative in 2008, experimentations of the CUSPIS project have been performed by means of EGNOS, European Geostationary Navigation Overlay Service, a precursor of Galileo.

either museums or archeological areas. In particular, visitors receive on their mobile devices appropriate and personalized items of information regarding the cultural assets they are visiting. Each user is assisted by an agent that acts as a guide during a visit, suggests routes and proposes suitable pieces of information. This by starting from a standard user profile which the agent later on tries to improve by eliciting the user's cultural awareness, habits and preferences. The enhanced profile will lead the agent to update the route and to propose customized information, including suggestions for future possible visits either to the same or to other locations. The Galileo satellite signal allows agents to follow the users during their visit and to capture their habits and preferences.

Reactivity allows the agents to adopt a specific behavior in response to what the user does and where she/he goes. Pro-activity has a main role because the reasoning process that leads to the interests deduction is based on the correlation of several data coming from the environment, from the ontology and from some basic inferential processes. In particular, the deduction process has been based on several observations such as the visitor route, the visited cultural assets, the time spent in visiting a specific cultural asset or explicit questions proposed directly to the visitor. More information about Users Profile Deduction via DALI agents in the CUSPIS project can be found in (Costantini *et al.* 2007).

Conclusions

We have presented the DALI agent architecture, that in our view includes all aspects that are commonly consider necessary in order to produce "intelligent" behavior. We have in fact emphasized the similarity between this architecture and a plausible functional model of the human brain. DALI is fully implemented, has a logical semantics and has been experimented. In the future, as a better understanding of the human brain will hopefully be reached, agent architectures in general will presumably evolve, and agents will become a more intelligent and useful support for the human life.

References

Byrne, M. D. 2001. ACT-r/PM and menu selection: applying a cognitive architecture to HCI. *International Journal of Human Computer Studies* 55(1):41–84.

Costantini, S., and Tocchio, A. 2002. A logic programming language for multi-agent systems. In *Logics in Artif. Intell., Proc. of the 8th Europ. Conf., JELIA 2002*, LNAI 2424. Springer.

Costantini, S., and Tocchio, A. 2004a. The DALI logic programming agent-oriented language. In *Logics in Artif. Intell., Proc. of the 9th European Conference, Jelia 2004*, LNAI 3229. Springer.

Costantini, S., and Tocchio, A. 2004b. Planning experiments in the dali logic programming language. In: J. Leite and J. Dix (eds.), Proc. of CLIMA IV, Fourth International Workshop on Computational Logic in Multi-Agent Systems.

Costantini, S., and Tocchio, A. 2005. Learning by knowledge exchange in logical agents. Simulazione ed Analisi Formale di Sistemi Complessi, Proceedings of WOA05, Universit di Camerino, Novembre 2005.

Costantini, S., and Tocchio, A. 2006a. About declarative semantics of logic-based agent languages. In Baldoni, M., and Torroni, P., eds., *Declarative Agent Languages and Technologies*, LNAI 3229. Springer. Post-Proc. of DALT 2005.

Costantini, S., and Tocchio, A. 2006b. Memory-driven agent behaviour for an agent-oriented logic programming language. CILC 2006, 26-27 giugno, Bari.

Costantini, S.; Inverardi, P.; Mostarda, L.; Tocchio, A.; and Tsintza, P. 2007. User profile agents for cultural heritage fruition. *Artificial Societies for Ambient Intelligence (ASAMI07).*

Hindriks, K. V.; Boer, F. S. D.; Hoek, W. V. D.; and Meyer, J.-J. C. 1999. Agent programming in 3apl. *Autonomous Agents and Multi-Agent Systems* 2(4):357–401.

Jo a. A. L.; Alferes, J. J.; and Pereira, L. M. 2002. Minerva - a dynamic logic programming agent architecture. In *ATAL '01: Revised Papers from the 8th International Workshop on Intelligent Agents VIII*, 141–157. London, UK: Springer.

Kakas, A.; Mancarella, P.; Sadri, F.; Stathis, K.; and Toni, F. 2004. The kgp model of agency. A. C. Kakas, P. Mancarella, F. Sadri, K. Stathis, and F. Toni. The KGP model of agency. In Proc. ECAI-2004, 2004. To appear.

Kowalski, A. 2006. How to be artificially intelligent - the logical way. Draft, revised February 2004, Available on line. http://www-lp.doc.ic.ac.uk/UserPages/staff/rak/rak.html.

Laird, J. E.; Newell, A.; and Rosenbloom, P. S. 1987. Soar: an architecture for general intelligence. *Artif. Intell.* 33(1):1–64.

Lisetti, C. L., and Marpaung, A. H. 2005. A three-layered architecture for socially intelligent agents: Modeling the multilevel process of emotions. In *ACII*, 956–963.

Muller, J. P. 1997. *The Design of Intelligent Agents: A Layered Approach.* Secaucus, NJ, USA: Springer Inc.

Novák, P., and Dix, J. 2006. Modular bdi architecture. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, 1009–1015. New York, NY, USA: ACM Press.

Rao, A. S., and Georgeff, M. P. 1991. Modeling rational agents within a BDI-architecture. In Allen, J.; Fikes, R.; and Sandewall, E., eds., *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, 473–484. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.

Tocchio, A. 2005. Multi-agent systems in computational logic. Ph.D. Thesis, Dipartimento di Informatica, Universitá degli Studi di L'Aquila.

Wang, Y.; Patel, S.; Patel, D.; and Wang, Y. 2003. A layered reference model of the brain. In *ICCI '03: Proceedings of the 2nd IEEE International Conference on Cognitive Informatics*, 7. Washington, DC, USA: IEEE Computer Society.