The DALI Agent-Oriented Logic Programming Language: Summary and References 2015

Stefania Costantini
Dept. of Information Science and Engineering, and Mathematics
University of L’Aquila Via Vetoio snc, I-67100 Coppito, L’Aquila, Italy
stefania.costantini@univaq.it

1 DALI: a Short History with References

DALI is a logic programming agent-oriented language defined in [1,2,3,4] and formalized in [5,6,7]. DALI has been implemented, and has been used in several (also industrial) applications [8,9,10,11,12,13,14,15,16,17,18,19]. A stable but not recent release of the DALI interpreter is publicly available at [20]. Latest releases are available upon request. DALI provides external, internal and events, reactivity and proactivity, a flexible communication architecture including a filter upon communications, and more additional features which have been developed over time [21].

For the definition of DALI we have built upon our past work about meta-reasoning and reflection in logic programming languages [22,23,24,25,26]. Semantics of such constructs is provided by means of Reflection Principles [26], inspired to the ones introduced in Symbolic Logic by Feferman in 1962. The operational semantics consists in an extended resolution procedure [23,24].

In [1,2,3] one can find the basic definition of DALI. In [6] we present the declarative semantics, and in [7] an approach to the operational semantics. Both approaches are general and can be applied to a variety of computational-logic-based agent systems.

We have initiated in [27] and [28] the design for DALI (but in principle for any logic-based agent-oriented environment) of a flexible management of memories by means of “past constraints”, which are language constructs that allow for memory maintenance, including forms of belief revision.

DALI agents are able to learn [29,30,31], either via “deep” learning or by learning from other trusted agents. This in order to enlarge the set of perceptions they can recognize, elaborate on and react to, and in order to expand their range of expertise.

In [32], we have proposed kinds of ASP (Answer Set Programming) modules to be invoked by a logical agents. Such modules allow for: complex reactivity based upon forms of commonsense reasoning; (meta-)reasoning about possibility and necessity; planning.

In [33] we have introduced an extension to the well-known LTL Linear Temporal Logic called A-ILTL, for “Agent-Interval LTL”, which is tailored to the agent’s world in view of run-time verification. A-ILTL provides a basis for approaches to agents run-time self-checking and assurance [30,34,35,36,37,16] and to complex event processing, also with preferences [38,39], following the seminal work presented in [40].
References


