A run-time perspective of agent modelling through AgentSpeak

Devindra Weerasooriya and Kotagiri Ramamohanarao
Dept. of Computer Science, University of Melbourne, Parkville 3052, Australia.
email: {dev, rao}@cs.mu.oz.au

Abstract
The needs of the application domain require that agent-systems be capable of supporting agents that are autonomous, concurrent, communicative and mobile. These systems should therefore be based on models of computation where concurrency has been dealt with explicitly. In order to realize the benefits of the agent-methodologies, it is also necessary to maintain a strong conceptual relevance to the domain of application. AgentSpeak supports multiple, autonomous, communicating agents that also possess properties of persistence.

Plan Language
The plan-language of AgentSpeak was designed from the perspective of a planning system; consisting of autonomous, but co-operating agents. The relevant constructs are categorised as follows.

- Agent-creation : Every agent is defined in-terms of an agent-family, which is essentially a template of what the agent consists of and what is generally visible; i.e. in-terms of its services, plans and structures of persistence.

- Plan : By itself a plan is a passive-object that resides within the plan-library. It can be activated by a matching service call which in-turn would be embedded within another plan. AgentSpeak is a forward planning system.

- Message-passing : Standard primitives of communication are enhanced with those based on agent-families; where a request may be sent to, either one or many of a family of agents.

- Scheduling : The language has a system of priorities that enables it to deal with real-time constraints.

Architecture
The architecture of AgentSpeak is scalable and therefore enables the creation of lightweight agents. Although the following description is of the default architecture, modules can be merged to achieve code-compactness. For this and other details regarding interaction between architectural modules, the reader should refer to [1].

- Actions Manager : This is where the goal processing takes place. Possibilities of processing multiple compatible goals exist, through the invocation of multiple threads.

- Plan Library : This is a passive module where the plans of the agent are contained.

- Persistence management system : This is an active module where the storage of the current state of simulation as well as local data about the agent state is managed. This module also helps achieve reliability of persistence.

- Communications Manager : The primitives of communication were sufficiently extensive and abstract to warrant a separate active module.

- Housekeeper : This module synchronises the actions of the three management modules and is also the interface of the agent to the underlying run-time system.

References