AgenTalk: Coordination Protocol Description for Multiagent Systems

Kazuhiro Kuwabara[†] **Toru Ishida**[‡] **Nobuyasu Osato**[†]

[†]NTT Communication Science Laboratories
2-2 Hikaridai, Seika-cho, Soraku-gun
Kyoto 619-02 Japan
{kuwabara, osato}@cslab.kecl.ntt.jp

[‡]Department of Information Science
Kyoto University
Sakyo-ku, Kyoto-shi, Kyoto 606-01 Japan
ishida@kuis.kyoto-u.ac.jp

In multiagent systems, achieving coordination among autonomous agents is a major problem. Generally, a high-level protocol needs to be designed to achieve effective coordination. In the field of distributed artificial intelligence, protocols for achieving coordination have been proposed, such as the contract net protocol for task allocation (Smith 1980), and the multistage negotiation protocol for resource allocation under global constraints (Conry *et al.* 1991). These protocols have been further customized to suit various application domains. In addition, as software agents proliferate, demand for customized protocols will increase.

In order to facilitate the development of multiagent coordination protocols, we designed a new language called AgenTalk. AgenTalk is not meant to be a formal specification language; rather it is meant to be a programming language capable of implementing protocols and agents behaving according to the protocols. Its design policies are as follows.

Explicit state representation of a protocol: An extended finite state machine which allows variables is used as a basis to describe coordination protocols. We call the representation of this state machine a *script*. Using this model, states of a protocol are explicitly defined, and actions of an agent can easily be defined for each state.

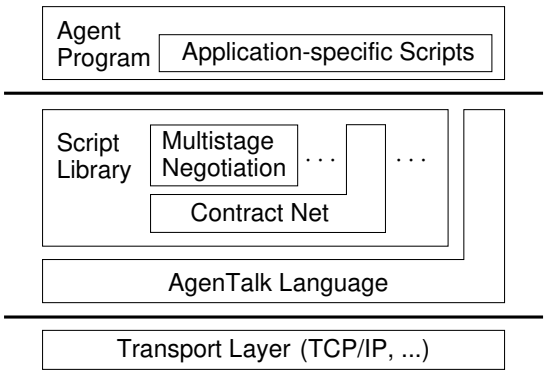

Figure 1: Architecture of an AgenTalk-based Agent

Incremental protocol definition: By introducing an inheritance mechanism in the definition of a script, protocols can be defined incrementally by extending existing scripts. For example, once a script of the contract net protocol is defined, the multistage negotiation protocol, which can be viewed as a generalization of the contract net protocol, can be defined inheriting most of the contract net protocol.

Easy protocol customization: A programming interface is defined which specifies the portion of a state transition rule that possibly needs to be customized for each application. By using this interface, only the application specific portion is required to be defined.

Conflict resolution between coordination processes: A single agent may be involved simultaneously in multiple *coordination processes* (e.g., different task allocation processes), between which some interactions exist (e.g., the agent's resources are shared). By allowing multiple scripts to be executed in an agent, and supporting communication between these scripts, a conflict resolution mechanism between coordination processes can be described.

The preliminary version of AgenTalk is implemented on top of Allegro Common Lisp, and is running on Sun and SGI workstations, where agents communicate over TCP/IP (Figure 1). AgenTalk is used in implementing the agent network called *Socia*, whose goal is to realize a *teleorganization* (Ishida 1994). On Socia, a desktop teleconferencing support system was developed. The AgenTalk software is available from the authors.

References

Conry, S. E.; Kuwabara, K.; Lesser, V. R.; and Meyer, R. A. 1991. Multistage negotiation for distributed constraint satisfaction. *IEEE Trans. Syst., Man, Cybern.* 21(6):1462–1477.

Ishida, T. 1994. Bridging humans via agent networks. In *Proc. 13th Int. Workshop on DAI.* 419–429.

Smith, R. G. 1980. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Trans. Comput.* 29(12):1104–1113.