An Intelligent Software Agent Framework for Decision Support Systems Development

Nikolaos F. Matsatsinis, Pavlos Moraitis, Vangelis Psomatakis, Nikos Spanoudakis
Technical University of Crete
University Campus, Kounoupidiana, 73100 Chania, Greece
Phone: +30-821-37348, Fax: +30-821-64824
email: {nikos, moraitis, psomat, spanoud}@dias.ergasya.tuc.gr

ABSTRACT: This paper describes an intelligent software agent framework for decision support systems development. An original consumer-based methodology for product penetration strategy selection in real world situations based on this framework was implemented. Intelligent software agent technology is the latest innovation of the distributed artificial intelligence research field and when many manufacturers and system developers adopted it, the need for a consistent, reusable and open framework arose. Agents are simultaneously considered according to two different levels: a functional and a structural level. In the functional level, we have three types of agents: task agents, information agents and interface agents assuming task's fulfilment through cooperation, information gathering tasks, and mediation between users and artificial agents respectively. In the structural level we have elementary agents based on a generic reusable architecture and complex agents considered as an agent organization that is created dynamically in a recursive way.

KEYWORDS: decision support system; intelligent software agent; distributed artificial intelligence.

INTRODUCTION

In this paper we propose a reusable intelligent software agent framework. All decision making tasks that include some or all of the characteristics considered in the literature, Jennings et al. (1996), Sycara and Zeng (1996) can be supported by information systems that are developed according to the proposed software agent framework. These characteristics are: a) the inherent distribution of problem solving abilities (the agents perform different tasks and methods), data, information, b) the necessity of flexibility, modularity (agents can appear and disappear in the system without disturbing its functionality) and reusability (customization of agents for new decision makers), c) problem solving complexity involving coordination between actors expressing different points of view.
Agents are considered simultaneously according to two different levels: a *functional* level and a *structural* level. In the *functional level*, we have a natural distinction between three different agent's functionalities: the information gathering task, the task's fulfilment by different types of cooperating specialists and the mediation between users and artificial agents, in order to allow users to control the actions of their agents. Therefore, we consider three types of agents as did Sycara and Zeng (1996): *interface agents*, *information agents* and *task agents*. In the *structural level* agents are considered as *elementary agents* and *complex agents*. We consider that complex tasks can be decomposed in a recursive way in several subtasks. In a similar way an agent structure can be considered according to different nested layers created in a recursive way. We can imagine a complex agent as a "Russian doll". The agent's layers are related to the subtasks that are carried out. This conception is inspired by the representation of a complex system through multiple layers proposed in control theory Mesarovic et al. (1970). Therefore, an agent is considered as a complex one when he realises a task involving several agents of at least one lower layer. An agent is considered as an elementary one, if he realises a primitive task. From a methodological point of view, we believe that this consideration facilitates the conception and the design of complex systems of multiple intelligent software agents, in order to achieve the modelling of complex real applications such as the one we present in this paper.

FUNCTIONAL LEVEL ARCHITECTURE

In figure 1 we present an intelligent software agent based architecture (this instance of the system is used by decision-makers, who are corporation board members, each simulating his own scenarios and finally selecting a market penetration strategy for a new or an existing product in a board meeting, which is a distributed decision making

process). All types of agents are presented in this figure. Note that the task agents shown in the figure are complex but this will be discussed in the structural level architecture section.

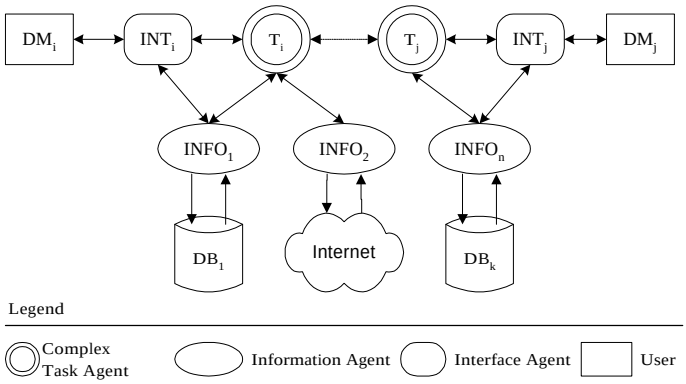

Figure 1: Agent based architecture

AGENT'S TYPES, FUNCTIONALITIES, STRUCTURE AND KNOWLEDGE HELD

The functionalities of *interface agents* are those we can find in the literature, Laurel (1997), Sycara and Zeng (1996): initiation of a task, responsibility of system interactions with the user, results presentation to user queries, in a way appropriate to the user's profile (e.g. according to the level of responsibility in an organization), determination of what categories of task agents should be involved, so that a user query is correctly taken into account.
The functionalities of *information agents* are also those we can find in the literature, Knoblock and Ambite (1997), Sycara and Zeng (1996). Their goal is to provide information and expertise on various topics, by drawing on relevant information from the system's general database, remote heterogeneous databases in the Internet, other information agents or interface agents.
Finally, *task agents* specialize in performing specific tasks. They can interact with all types of agents in order to carry out their jobs. These are the most sophisticated agents of our system and they can have an elementary or complex structure. Different types of task agents (elementary and complex) exist, each corresponding to different *generic tasks* (e.g. a generic task can be the performance of a data analysis method) depending on the decision making process that is modelled. Several occurrences of the same type are used to perform specific tasks, instances of the above generic ones (following the previous example one agent can perform the correspondence analysis method while another may perform the Q analysis method). Finally, we can also have several occurrences performing the same specific task (e.g. several agents performing the Q analysis method).
By using elementary agents that perform simple generic tasks we build complex agents, taking into account the current methodology's complex tasks achievement (elementary task agents are reusable components that can be utilised at a later time by other decision supporting systems). We consider that by using the complex agent concept to gather together agents involved in some complex task (if the task's nature allows it) achievement, the system's scale and coordination complexity can be decreased, making the application's modelling easier. Actually, coordination, even within a large-scale application, is carried out, either between agents within relatively small-scale groups or between a reduced number of complex agents that are components of an upper layer. In the latter case, coordination is carried out by intra-agent control primitives assuming interaction between lower layer agents of a complex agent.

AGENT ORGANIZATION

Agents can be geographically distributed allowing the interaction with users of different levels of responsibility and involvement in the main problem solving. Agents (elementary and/or complex) interact with each other by means of passing messages. A message is structured in such way that allows the transfer of the necessary information and semantics between agents for cooperative work accomplishment. Our agent organization allows the following interactions types (for an extended illustrative scenario see section 5):

- An agent will be informed when a new member enters his community or when another leaves
- During the problem solving process, appropriate agents activation dynamically forms an organizational structure that fits with the current goal. In our approach, interface agents activate task agents
- Activities of information agents are initiated, either top down by a user or a task agent through queries, or bottom up through monitoring information sources for a particular information
- The interface agents can receive messages/queries from users

AGENT ARCHITECTURES

In this section we present the elementary and complex agent architectures (figure 2).

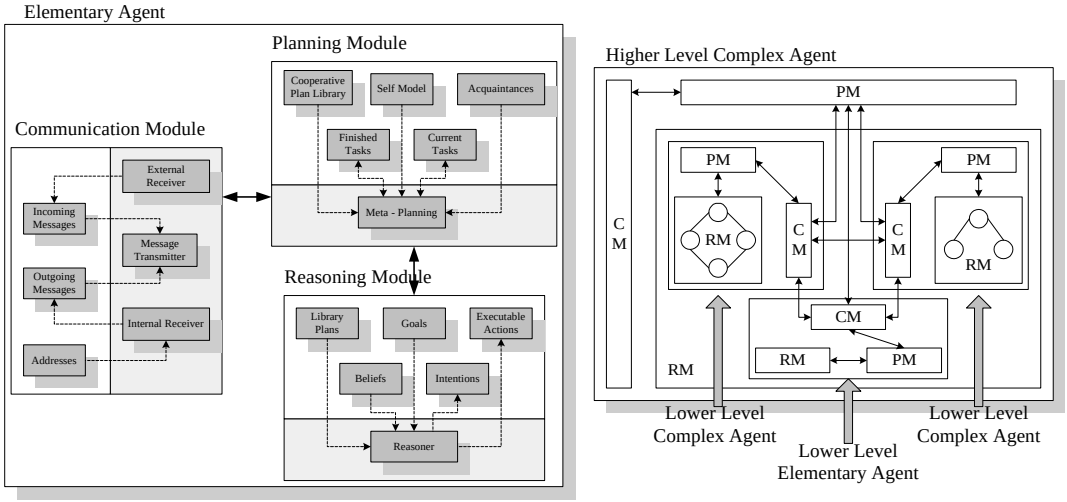

Figure 2: a) Elementary and b) complex agent architecture view in top layer

ELEMENTARY AGENT ARCHITECTURE

The agents, which are used for the presented application modelling, are based on a generic reusable architecture that we conceived, following the general BDI type philosophy Georgeff and Ingrand (1989), Rao and Georgeff (1992) and we were inspired by the different agent architectures presented in the literature Brazier et al. (1997a), Sycara and Zeng (1996), Witting (1992). Different functional agent types have the same basic architecture principles, regardless of the category they belong to. However, the different models are more or less sophisticated according to their specific type (e.g. the planning model of an information agent is simpler than that of a task agent). Our agent architecture (Fig. 3a) is composed of three modules (*Communication, Planning and Reasoning module*) that intercommunicate through internal message exchanging (called intra-agent messages). These modules run concurrently. An agent remains idle while no messages arrive to his communication module. As soon as a message arrives, the communication module determines its importance and, after transforming it to an intra-agent message, sends it to the planning module by means of a message queuing mechanism. There the message is processed and appropriate actions are subsequently undertaken either by the reasoning module (execute a task) or by the communication module (send one or more messages) or both. All modules adopt this behaviour and remain idle while no messages are available for procession. The same intra-agent queuing mechanism facilitates all modules.

COMPLEX AGENT ARCHITECTURE

The architecture of a complex agent is similar to the one of an elementary agent. Therefore he is composed of the same three modules (C*ommunication, Planning and Reasoning module)* which intercommunicate through internal message exchanging. The intra-agent control (interaction between the three components) is the one of the elementary level.
The difference is situated in the structure of the reasoning module. The group of agents (elementary and/or complex) which compose it assumes its role. The task achievement of an agent (parent) developed in *n-layer* is therefore the result of the set of agent's (his descendants) cooperation belonging to the previous (*n-1) layers*. The reasoning module could be therefore considered as an *agent organization*.

CONCLUSION AND FUTURE WORK

In this paper, we presented an intelligent software agents framework to support any decision making process for the first time. The proposed framework is using a generic reusable agent architecture. Agents are considered simultaneously in two levels, a functional and a structural level. In the functional level we have three types of agents, task, INT and INFO agents, while in the structural level we have elementary and complex agents. The difference with other works Brazier et

al. (1995), Ishida et al. (1990), Jennings et al. (1995) is, like in Rao and Georgeff (1992), the differentiation between three types of agents in the functional level, allowing efficient operation for real complex tasks achievement involving coordination, information gathering and user interaction. Compared to Rao and Georgeff (1992) the difference is that we introduce the structural level consideration for the three types of agents. We consider that by using the complex agent concept, thus gathering together agents involved in some complex task (if the task's nature allows it) achievement, the system's scale and coordination complexity can be decreased, therefore simplifying application modelling. Actually, coordination, even within a large-scale application, is carried out either between agents (elementary and/or complex) within relatively small-scale groups, or between a reduced number of complex agents, components of a distributed system or components of an upper layer. In the latter case, coordination is carried out by intra-agent control primitives assuming interaction between different layers of agents of a complex agent. In the literature, important works Brazier et al. (1997a), Brazier et al. (1997b) have introduced, in a similar way, the concept of complex (composed) agent, considering a generic agent model composed by six components. Each of them can be refined in many ways, resulting in models of agents with different characteristics Brazier et al. (1997c), Brazier et al. (1996). The difference with these works is that in our approach communication and planning modules have the same structure in elementary and complex level, while the reasoning module can be considered in the complex level as an agent organization. This organization is created dynamically in a recursive way; it can have different forms and is task dependent. Our future work will be to apply our platform to new applications and to implement complex INFO and INT agents.

REFERENCES

Brazier, F.M.T., Dunin-Keplicz, B.M., Jennings, N.R., Treur, J., 1995, "Formal specification of multi-agent systems", First International Conference on Multi-Agent Systems (ICMAS'95), San Francisco, CA, pp. 25 - 32.

Brazier, F.M.T., Dunin-Keplicz, B.M., Treur, J., Verbrugge, R., 1997b, "Modeling Internal Dynamic Behavior of BDI Agents", Proceedings of Third International Workshop on Formal Models of Agents, MODELAGE'97, Lecture Notes in AI. Springer Verlag.

Brazier, F.M.T., Jonker, C. M., Treur, J., 1997c, "Formalization of a cooperation model based on joint intentions", Proceedings of the ECAI'96 Workshop on Agent Theories, Architectures and Languages, ATAL'96, Muller, J.P., Wooldridge, M.J., Jennings, N.R., Intelligent Agents III. Lecture Notes in AI, Vol. 1193. Springer Verlag, pp. 141 - 156.

Brazier, F.M.T., Treur, J., 1996, "Compositional modeling of reflective agents", Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-based Systems Workshop, KAW'96, SRDG Publications 23/1 - 13/12.

Brazier, F.M.T., Dunin-Keplicz, B.M., Jennings, N.R., Treur, J., 1997a, "DESIRE: Modeling Multi-Agent Systems in a Compositional Formal Framework", International Journal of Cooperative Information Systems. Special issue on Formal Methods in Cooperative Information Systems: Multi-Agent Systems.

Georgeff, M.P., Ingrand, F.F., 1989, "Decision-Making in an Embedded Reasoning System", Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89), Detroit/MI, pp. 972 - 978.

Ishida, T., Yokoo, M., Gasser, L., 1990, "An organizational approach to adaptive production systems", Proceedings of AAAI-90. Boston, Mass.

Jennings, N.R., Corera, J.M., Laresgoiti, I., 1995, "Developing industrial multi-agent systems", First International Conference on Multi-Agent Systems (ICMAS'95), San Francisco/CA, pp. 423 - 430.

Jennings, N.R., Faratin, P., Johnson, M. J., O'Brien, P., Wiegand, M.E., 1996, "Using Intelligent Agents to Manage Business Processes", Proceedings of PAAM'96, pp. 345 - 360.

Knoblock, G.A., Ambite, J.L., 1997, "Agents for Information Gathering", Software Agents, pp. 347 - 373.

Laurel, B., 1997, "Interface Agents: Metaphors with Character", Software Agents, pp. 67 - 77.

Mesarovic, M.D., Marko, D., Takahara, Y., 1970, "Theory of Hierarchical, Multilevel, Systems", Academic Press, New York.

Rao, A.S., Georgeff, M.P., 1992, "An abstract architecture for rational agents", Proceedings of Knowledge Representation and Reasoning (KR'92), Cambridge, Massachusetts, pp. 439 - 449.

Sycara, K., Zeng, D., 1996, "Coordination of Multiple Intelligent Software Agents", International Journal of Cooperative Information Systems, World Scientific Publishing Company.

Witting, T., 1992, "ARCHON: An Architecture for Multi-Agent Systems", Ellis Horwood Series in AI.