

電子情報通信学会技術研究報告

AI 99-41~51

〔人工知能と知識処理〕

1999年9月6日

電子情報通信学会技術研究報告目次

CONTENTS

〔人工知能と知識処理〕

[Artificial Intelligence and Knowledge-Based Processing]

— ソフトウェアエージェント —	
(1) AI 99-41	
ADIPSフレームワークにおけるJavaオブジェクトのエージェント化実現方式	1
藤田 茂・菅原研次 (千葉工大), 木下哲男・白鳥則郎 (東北大)	
(2) AI 99-42	
ADIPSフレームワークにおけるエージェント間競合発見手法とその応用	7
菅沼拓夫・越智大介・木下哲男・白鳥則郎 (東北大)	
(3) AI 99-43	
エージェントの集団形成機構とその動作特性について	15
加藤貴司・木下哲男・白鳥則郎 (東北大)	
(4) AI 99-44	
Organizational Knowledge and Its Application in Multiagent System Development	23
Behrouz H. Far (Saitama Univ.)	
(5) AI 99-45	
FIPAの動向	31
須栗裕樹 (コムテック)	
(6) AI 99-46	
エージェント間通信における信頼度計算手法	39
寺田賢二・櫛 肅之 (NTT)	
(7) AI 99-47	
エージェントプログラミングとその形式的検証	47
櫛 肅之 (NTT)	
(8) AI 99-48	
分散オブジェクト指向システムにおける知的トレーダの構築	55
松本晋一 (富士通九州通信システム), 雨宮真人 (九大)	
(9) AI 99-49	
パターンに基づく移動エージェントシステムの設計手法	63
小松千尋・藤田 悟・山之内徹 (NEC)	
(10) AI 99-50	
マルチエージェントフレームワークBee-gentを用いた分散システムにおけるデザインパラダイムの分類と評価	71
川村隆浩・吉岡信和・長谷川哲夫・大須賀昭彦・本位田真一 (東芝)	
(11) AI 99-51	
Efficient Use of Mobile Agents	79
Sam Joseph, Masanori Hattori, Naoki Kase, Akihiko Ohsuga (Toshiba)	

Note: The articles in this publication have been printed without reviewing and editing as received from the authors.

Organizational Knowledge and Its Application in Multiagent System Development

ベルーズ H. ファー

〒 338-8570 埼玉県浦和市下大久保 255
埼玉大学情報システム工学科
Tel. 048-858-9612, Fax. 048-858-3716
E-mail. far@cit.ics.saitama-u.ac.jp
<http://www.cit.ics.saitama-u.ac.jp/~far/>

あらまし： 本研究では、複数のエージェントから構成されるネットワーク上の組織（GAG: Generalized Agency、分散型情報処理環境）と標準化されたエージェント（AG: Generalized Agent）のモデルを紹介する。GAG は、組織知識を持っている。組織知識は、個々のエージェントが所有するものではなく、エージェントの相互作用から発生する知識である。本研究では、記号構造による組織知識のモデル化、獲得、応用について述べる。様々な応用例が考えられるが、電子商取引システムを実験対象とする。

和文キーワード エージェント、知識共有、組織、記号構造

Organizational Knowledge and Its Application in Multiagent System Development

Behrouz H. Far

Department of Information and Computer Science, Saitama University
255 Shimo-okubo, Urawa 338-8570, Saitama, Japan
Tel. +81-48-858-9612, Fax. +81-48-858-3716
E-mail. far@cit.ics.saitama-u.ac.jp
<http://www.cit.ics.saitama-u.ac.jp/~far/>

Abstract: Although there are many projects focusing on multiagent systems, there are only a few focusing on systematic design of large scale multiagent system. In this paper we will formalize the knowledge representation and sharing of agents, using *symbol structures*, define agencies as *organizations* (i.e., a coalition of agents), propose a formalism to represent *organizational Intelligence*, devise a basic configuration for *generalized agents* (AG), and derive certain patterns for *generalized agencies* (GAG) in order to use them effectively in a large scale multiagent system design. The private knowledge of an AG agent is represented by *symbol structure* and AG agents can share their knowledge using combination, specialization and generalization methods. Opposite to the other works, organizational knowledge, is defined as a property of at least a pair of AG agents. GAG is defined as an organization of AG agents. Each GAG describes a problem, such as a business process that happens repeatedly, and describes the participant agents as well as the process towards the solution to the problem. Furthermore, alternative configurations, decisions and trade-offs are defined. Electronic commerce is an example of such systems.

Key words *Agent, Knowledge sharing, Organization, Symbol structure.*

1. Introduction

There are already many projects focusing on agent-based solutions for various scientific and business domains [1], [12], [14], [15]. However, at this moment, there are certain limitations to such solutions, such as, they may not be appropriate for systems and domains with global constraints, domains that rely heavily on knowledge sharing while decisions are made on a local basis, domains with decentralized control and finally domain that require achieving a globally optimal performance.

Building agent-based solutions requires research on at least three topics. First, working on agent theory, the scope and limitations of agent-based solutions. Second, making agent frameworks and infrastructure powerful, interoperable, and secure enough to support large-scale coordinated problem-solving activity. Third, developing new tools to help building agents [2].

In this paper we focus on the second topic. We formalize the knowledge representation and sharing of agents, using *symbol structures*, define agencies as *organizations* (i.e., a coalition of agents), propose a formalism to represent *organizational Intelligence*, devise a basic configuration for *generalized agents* (AG), and derive certain patterns for *generalized agencies* (GAG) in order to use them effectively in a large scale multiagent system design.

The knowledge of an AG agent is represented by *symbol structure* and AG agents can share their knowledge using combination, specialization and generalization methods (see Section 5.2). Opposite to the other works, organizational knowledge, is defined as a property of at least a pair of AG agents. GAG is defined as an organization of AG agents. Each GAG describes a problem, such as a business process that happens repeatedly, and describes the participant agents as well as the process towards the solution to the problem. Furthermore, alternative configurations, decisions and trade-offs are defined.

2. Multiagent System Issues

2.1 Knowledge sharing issues

The infamous knowledge sharing problem arises from the fact that different systems use different concepts and terms for describing domains. This makes it difficult to take knowledge out of one system and use it in another. Interest in ontologies has grown due to interests in knowledge sharing, data integration, knowledge inter-operation, and reuse [8]. In multiagent system design, ontologies that can encompass agent's internal knowledge as well as the system's organizational knowledge should be developed. Ontology research yet has to produce proposals for how to construct and maintain such ontologies [6].

Knowledge sharing requires knowledge level com-

munication. Various kinds of Agent Communication Language (ACL) have been proposed. ACL is a language with precisely defined syntax and semantics serving as the basis of communication between independently designed and developed software agents. Among ACL languages, Knowledge Query and Manipulation language (KQML), a protocol for exchanging information and knowledge [16] and Knowledge Interchange Format (KIF), a formal syntax for representing knowledge, basically based on first-order predicate logic [9], are the most famous ones.

In our project, we suggest a way of systematically building the knowledge base of individual agents and blending it with the ontology of the domain, in a seamless way, using *symbol structures*. Knowledge sharing is treated as revealing the internal symbol structure to the other agents (see Section 5.3).

2.2 Problem solving issues

There are already a number of approaches to distributed problem solving, using agent technology, such as blackboard systems, broadcast methods, delegation, contract networks [23], multiagent truth maintenance system (TMS), etc. A main goal in multiagent problem solving is to maintain the coherent performance of the agents while maintaining their autonomy. A number of protocols for cooperation, coordination and competition have been suggested (for a survey see [13]). However, there is no single formalism that can cover all. The symbol structure framework, suggested in this paper gives a coherent view of cooperation, coordination and competition in a distributed problem solving environment (see Section 5.3).

2.3 Agent design issues

Multiagent system design is a main theme of this paper. *Scalability* is a central issue in multiagent systems design. Although there are many projects focusing on multiagent systems and groupware, it is quite hard to scale them up to include a large number of actors (i.e., agents and/or human counterparts). *Reuse* is another fundamental issue. The once designed agent configurations are to be used in another configurations constantly and repetitively. Existing research projects do not specify how their proposed agent technology can be used for systematic design of large scale multiagent systems. In other words, there is no standard technique for multiagent system design and development, including techniques for requirement analysis, design and implementation.

There are a number of approaches to agent design, such as: treating agents as objects, agents as expert systems, reactive agents, agents with memory and state [3], etc. There are certain efforts to standardize the agent technology by *Federation of Intelligent Physical Agents*, FIPA (<http://www.fipa.org>).

There are already some agent developing tools available, mostly implemented in Java programming language, such as ABE, JAT and OAA [7]. Such tools and environments can facilitate the communication (for example, using KQML language in JAT as the knowledge level communication) and message passing (for example, using HTTP and IIOP protocols in ABE and OAA, respectively), but they fail to provide the appropriate aids to build the knowledge bases in the sense that we use in this project.

3. Generalized Agent (AG)

In this section we introduce *Generalized Agent (AG)* framework to represent and model individual agents. The framework supports human-life epistemology, that is, trying to follow the knowledge representation and problem solving skills of humans. People process and manipulate information in a different way than solution to a mathematical problem or formal proof structure in standard logic. Humans appear to get as input the sensory icons coming from the outside world and developing a coherent model of the situation. Then process rules are used to reason, interpret and explain the situation and derive possible actions. AG framework supports this viewpoint.

Definition 3.1 (Generalized Agent (AG)):

Generalized Agent (AG) is an information processing object, acting on the basis of representation, using as input information the sensory icons coming from the external environment, *perceiving*, *conceptualizing*, *interpreting*, and *performing actions* to direct its behavior towards a desired goal.

A basic assumption is that performing these activities is analogous to computation [17]: AG acts on the basis of representations and initiates such representations physically as cognitive codes and the actions are carried out as a consequence of operations carried out on those codes.

Among the 4 main tasks mentioned above, *conceptualization* (C) and *interpretation* (I) are studied herewith. Conceptualization is realized by the *symbol structure* (see Section 5). Interpretation encounters both interpreting properties of the environment by means of causal laws, qualitative or quantitative model of domain objects, as well as other kinds of *organizational* knowledge, including knowledge on, protocols, coalition strategies, etc.

AG agents have the following common characteristics:

Artificiality: AG agents do not exist naturally or biologically in the outside world and they are deliberately designed by the humans to fit in to their environments.

Adaptability: AG agents have dynamically changing conceptual picture of the situation, and goals, as well as selectivity among a number of mechanisms to cope with the changing environments.

Rationality: AG agents are rational. Given a set of actions, the AG selects an action which is feasible and optimal.

Individual Intelligence: AG agents have efficient problem solving and decision making skills on a personal basis, such as segmentation of the search space and jumping to conclusion.

Communication and Knowledge Sharing:

AG agents have communication and knowledge sharing skills that help their individual problem solving and decision making skills be utilized efficiently.

Memory and State: AG agents have memory and state. Their profile of actions and states over time is called *history*.

Another way to make performance level of AG agents get close to that of human being is providing them with *insights*.

Roughly speaking, an insight is defined as the ability of deriving information from a set of data through method other than monotonic inference based on causal assumption. We define insights in terms of inferential activities enabling to perform *generalization* and *specialization* from a given set of data. In Section 5, we propose methods to incorporate insights by developing method of generalization and specialization within the symbol structure (SS) framework.

4. Organizational Intelligence (OI)

Organizations, of various forms, physical, cognitive, temporal and institutional have been studied in management and computer sciences. The game theoretic approach to study organization focuses on modeling and suggesting computational algorithms for certain aspects of the coalitions, such as social welfare [19], individual rationality, voting consensus, etc.

The computational approach focuses on identifying general principles of organization and their exceptions. The proposed theories extend the information processing capabilities of individual agents to an organization level, through defining concepts such as, *bounded rationality* [21]. The main stream of computational organization theories are surveyed in [4].

The already proposed multiagent system, as a purposful organization of agents, do not offer global optimization for the knowledge resources and skills of the participant agents, i.e., agent organization's capabilities in almost all of the cases is less than the sum of capabilities of its participant agents. However in many natural coalitions, such as bees and ants societies, the organization has the merit of overcoming shortcomings of participant agents.

We believe that a main reason is the improper interpretation and implementation of the *Organizational Intelligence* (OI).

Here we briefly discuss the OI features, its underlying assumptions and the model will be introduced in the next section.

4.1 Intelligence of Pair (IoP)

There is a fundamental question whether OI resides in an agent itself, external to the agent, or an outcome of the interaction among agents.

At the first glance, it seems that OI includes knowledge distribution (who knows what?) and sharing (how it can be utilized?) as mentioned in [5]. All of the proposed theories and techniques have taken this for granted that OI exists either within or external to the agent with no respect to interactions among the agents. However there are certain difficulties in both logical formulation and actual implementation of such theories.

We believe that this is a wrong assumption. We think that in a purposful (i.e., not random) organization, OI is a property of interaction among agents and can only be ascribed to at least a pair of agents. We call this ‘*Intelligence of Pair (IoP)*’ assumption.

4.2 History Patterns (HP)

In biological coalitions, participants may have a kind of *role* or *function* (during interaction with the other participants), if they show some persistence in their profile of actions over time. The same could be devised for artificial coalitions. As a matter of fact, it is not difficult to find organizations that display non-random persistent and repeated patterns of actions [5].

Agents act and perform in a physical world. Their past experiences can be recorded and explained in terms of their *histories*, that is, their *profile* of actions and *states* that they go through.

Intuitively, histories can display certain patterns. A basic feature of state representation is that it assigns a certain characteristic to its reference agent. Therefore it is possible to define OI patterns with reference to discovery of an order in the history. A key idea is that OI patterns emerge from discovering a persisted state or an ordered pattern in the agent’s history. We call this ‘*History Patterns (HP)*’ assumption.

In biological coalitions, persistence is considered to be the most interesting characteristic and is believed to be governed by natural selection law. In artificial coalitions, besides persistence, other kinds of ordered patterns, such as repetition cycles, may also be considered as important features of the coalition.

4.3 Computation method

Based on the IoP and HP assumptions, a computation method for generating OI concepts is devised. Fig. 1 depicts the basic idea. Interaction between

agents in an organization is represented by their *inputs* and *outputs*. Inputs and outputs are described by a shared set of variables. In this sense, an agent can be viewed as an n-bit processor, whose role in the organization is dependent to first, the active bits on the shared bus with the other agents, and second, the other agents having the same bits active.

In this method, first, a pair of AG agents are selected by using combination rules (see Section 5.2) their pairwise profile is produced (see Section 5.3). Then by using generalization and specialization rules (see Section 5.2) and a simple pattern detection algorithm, possible repetition and persistence patterns are derived and added to the knowledge base of the organization.

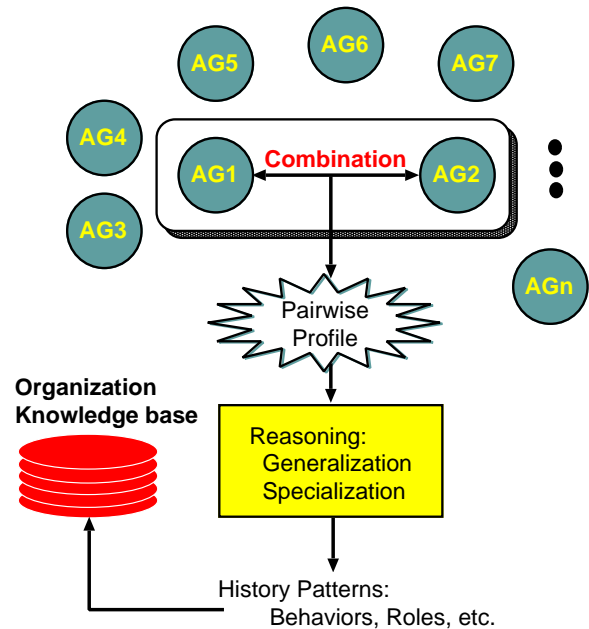


Fig. 1 Organization Intelligence concept

Deriving *organizational knowledge* based on these assumptions has many advantages: first, concepts derived in this way can be explained in terms of organization and its comprising agents without reference to any other intermediary concepts. Second, it provides a framework for comparing and evaluating completely different organizations. Third, the organizational knowledge base is updated dynamically, accounting for different configuration of the participant agents. Finally, it is an appropriate vehicle to explain the need for services of a certain agent in an organization. All of these factors are necessary in organization design.

The next section describes the knowledge structure for each participant AG and how it can be used to derive the *organizational knowledge*.

5. Symbol Structure (SS)

In this section we introduce *Symbol Structures (SS)*. SS is used to represent private knowledge of an individual agent and help deriving organizational knowledge of the agency. The SS model is multi-level graph. Formal rules of inter-layer transition are introduced. Flexibility, extedibility and interoperability are three main advantages of knowledge representation and reasoning with SS.

Species of information processors, including AG agents, are members of the family of symbol systems. A symbol system is a machine that, as it moves through time, produces an evolving collection of symbol structures [22]. One of the capabilities that all kinds of symbol systems have in common is that they act on the basis of symbolic representations.

Symbol structures are the internal representations of the outer environment to which the symbol system is seeking to adapt. A symbol structure is a way to assimilate separate facts into a coherent image that with the aid of some logical inference rules, can be used to detect contradictions and draw decisions.

AG agents can take symbol structures as the model, test for differences between structures, operators that alter the structures and goals and test for their achievements. AG agents possess a common symbol set and a number of operations that operate upon the symbol structure. Knowing the common symbol set and operations, together with the assumption that the problem solving behavior is concerned with the representations by certain general principles, makes it possible to explain an important segment of the regularities in problem solving behavior of those agents.

5.1 Definitions

We assume that the perception mechanism of the AG gets as its input the sensory icons coming from the external world and produces at its output a symbol structure. We also assume that the knowledge structure of the AG is available in the form of a SS and formalize the information processing technique by means of symbol structures.

Definition 5.1 (Symbol Structure):

A symbol structure (SS) is a finite connected multi-layer bipartite graph. There are two kinds of nodes in each layer of SS: *concepts (c)* and *relations (r)*. The concepts belong to concept set $C = \{c_1, \dots, c_n\}$ and are linked by relations belonging to relation set $R = \{r_1, \dots, r_m\}$ to form a model that approximates an entity or scene e , which is represented by the concepts from the set C and relations from the set R .

One source of difficulty when processing concepts, is distinguishing a concept at various levels of abstraction, as well as differentiating between generic concepts and their instances. Function *type* is defined to ease such differentiation. Henceforth, we basically

work with the types of concepts rather than individual concepts. We will further define a type hierarchy which is a higher order relation between various levels of abstraction. Relations, are also classified by types.

Definition 5.2 (Type): The function *type* maps concepts (C) and relations (R) onto a set T . The elements of T are called type labels.

$$type : C \cup R \rightarrow T \quad \forall x \in C \cup R \mid type(x) \in T(1)$$

Concepts c_1 and c_2 are of the same type if, $type(c_1) = type(c_2)$. Relations r_1 and r_2 are said to be of the same type if, $type(r_1) = type(r_2)$ and r_1 and r_2 have the same number of arcs.

Definition 5.3 (Instance): Instances of a concept c , shown by $I(c)$, are every occasions that c comes to existence. It is shown by:

$$I(c) : (c^1, c^2, \dots, c^j, \dots) = (\forall j, c^j) \quad (2)$$

Similarly, instances of a type t , shown by $I(t)$, are every occasions that every concept c of type t comes to existence.

Definition 5.4 (Subtype): The subtype t_s of type t is the set of all instances of type t_s which also are instances of type t .

$$\forall i, j \quad (c_i^j \in I(t_s)) \wedge (c_i^j \in I(t)) \rightarrow t_s \prec t \quad (3)$$

(\prec) is a two point symbol denoting subtype relation;

Type hierarchy provides a means of evaluating a concept at various levels.

Definition 5.5 (Type Hierarchy): The type hierarchy is a partial ordering defined over the set of type labels, T . The symbols (\prec) and (\succ) designate ordering. For the type labels, s , t and u ,

- If $s \prec t$, then s is called a subtype of t ; and t is a supertype of s , shown by, $t \succ s$.
- If s is a subtype of both t and u (i.e., $s \prec t$ and $s \prec u$), then s is a common subtype of t and u .
- If s is a supertype of both t and u (i.e., $s \succ t$ and $s \succ u$), then s is a common supertype of t and u .

5.2 Reasoning: Combinatory and insight rules

Until now we have developed the basic syntax of symbol structures. This is powerful enough to represent knowledge at various levels of abstraction. In this section we show how to reason with symbol structures.

AG agents are assumed to be goal seeking open loop artifacts. A goal is a description of the final state of reasoning with a symbol structure.

A goal can be achieved in two ways: first, reasoning on each layer; second, transition between layers of a symbol structure. Basically, there are only two reasoning rules, as described below.

Proposition 5.1 (Combination Rules): There are two combination rules for symbol structures.

- *Join rule*: Join rule merges identical concepts. If a concept c in u is identical to a concept d in v , then let w be the symbol structure obtained by deleting d and linking to c all arcs of conceptual relations that had been linked to d .
- *Simplification rule*: Redundant relations of the same type linked to the same concept in the same order can be reduced by deletion all but one. If the relations r and s in the symbol structure u are duplicates, then one of them may be deleted from u together with all its arcs.

Definition 5.6 (Insight): Insight is the ability to generalize and specialize information from a symbol structure.

We introduce the *specialization* and *generalization* rules for symbol structures. Specialization is equivalent to increasing the precision in modeling. The outcome of applying specialization rules on a structure is another structure which is more restricted than the original one. Conversely, generalization proceeds in the reverse order of specialization.

Definition 5.7 (Generalization / Specialization): For two arbitrary levels u and v of any symbol structure, if u is identical to v except that some type labels of the nodes of v are restricted to subtypes of the same nodes in u , then u is called a specialization of v , written $u \prec v$, and v is called a generalization of u .

Generalization defines a partial ordering among the levels of any symbol structure, called the generalization hierarchy. Specialization is equivalent to replacing the type label of a concept with the label of a subtype.

Definition 5.8: (Common Generalization / Specialization)

Let u_1, u_2, v and w be any arbitrary levels of a symbol structure. If $u_1 \prec v$ and $u_2 \prec v$, then v is called a common generalization of u_1 and u_2 . If $w \prec u_1$ and $w \prec u_2$, then w is called a common specialization of u_1 and u_2 .

Note that in above definition, u_1 and u_2 might be two separate symbol structures given in the same level of abstraction.

5.3 Interaction among agents

Now we have a seamless framework for representing and reasoning with the knowledge on individual as well as the organizational basis. Moving from one agent to another and moving from any level of abstraction to another is supported. Therefore we can define the basic agent interactions, i.e., *cooperation* and *competition*.

Definition 5.9 (Knowledge Sharing): Knowledge sharing is equivalent to merging two or more symbol structures using *combination* and *insight* rules.

For a pair of agents to cooperate, each should maintain a model of the other agent, as well as a model of future interactions [13].

Definition 5.10 (Cooperation): Cooperation is revealing an AG agent's *goal* and the knowledge behind it, i.e., its symbol structure to the other party. In cooperation both agents have a common goals.

Definition 5.11 (Coordination): Cooperation is revealing an AG agent's *goals* and the knowledge behind it, i.e., its symbol structure to the other party. In coordination, agents have separate goals.

Definition 5.12 (Competition): Competition is revealing only an AG agent's *goals* but masking the knowledge behind it to the other party.

Definition 5.13 (Agent's Profile): Agent i 's profile, \mathcal{P}_i is a finite sequence of (s_i, a_i) , recorded during reasoning on the symbol structure.

where, s_i is problem solving state;

a_i is action;

$a_i \in A_i$;

A_i is the action set available to agent i .

Some of the states may map to actual actions, some may not. In the latter case the action will be NULL.

Definition 5.14 (Pairwise Profile): A pairwise (i.e., joint) profile of two cooperative agents, i and j , \mathcal{P}_{ij} is a finite sequence of $([s_i, s_j], [a_i, a_j])$, during reasoning on the symbol structure, after merging their symbol structures using *combination* and *insight* rules. Each action is a pair of joint actions of the two agents.

$\forall [a_i, a_j] \quad a_i \in A_i, \quad a_j \in A_j$

A_i and A_j are the action set available to agent i and j , respectively.

Fig. 2 depicts the *cooperation* scenario. The two agents' SS is merged and compatible concepts are mapped to one another. The reasoning will progress on the joint SS, setting the goals of the agents be identical.

6. Generalized Agency (GAG)

Generalized agency (GAG) describes a particular problem, such as a business process. GAG is a description of communicating and cooperating agents, which serve as the building blocks of the agency and customized to solve a general problem described by the GAG itself. For each GAG, the participating agents, their roles, collaborations and distribution of responsibilities are defined.

Definition 6.1 (Organization): An organization is a goal directed coalition of agents in which the agents are engaged in one or more tasks and knowledge and capabilities are distributed among the agents.

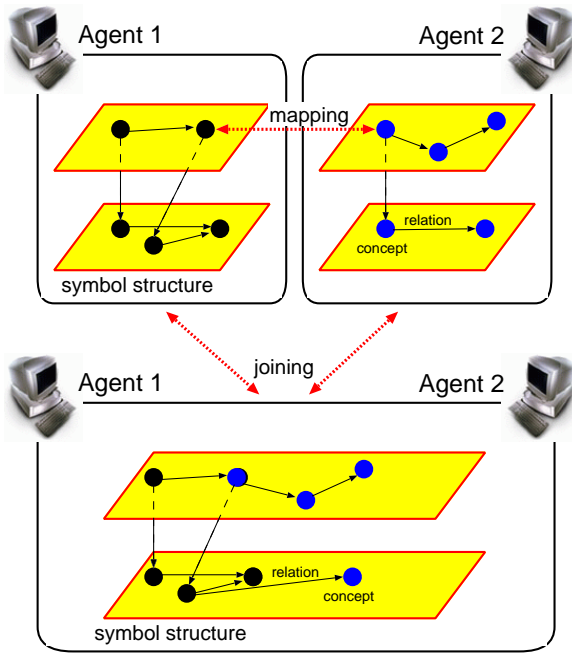


Fig. 2 Example of symbol structure

Definition 6.2 (Generalized agency (GAG)):

A Generalized agency (GAG) describes a particular problem domain, participant agents, their roles, collaborations and distribution of responsibilities and process towards the solution to the problem.

A problem domain is a class of problems that have certain properties in common and its various versions happen repeatedly e.g., variants of a business process.

GAG is too general to be used in multiagent system design. It is usually used within a certain framework, i.e, problem domain together with some additional context.

Participating agents are described by either of the followings.

Class: Specifying inner environment.

Interface: Specifying the input/output and/or functions.

AG agents are assumed to be goal seeking open loop artifacts interacting with the environment. Fulfilment of purpose or adaptation to a goal involves a relation among three terms:

- The purpose or goal;
- The character of the artifact (inner environment);
- The environment in which the artifact performs (outer environment);

The inner environment is an organization of parts capable of attaining the goals in some range of outer environments. Describing AG agents by their *class* means specifying their *internal implementation*.

On the other hand, here may be a number of functionally equivalent agents capable of attaining the goals in the same outer environment. The resemblance of the behavior without identity the inner environment is particularly feasible if the aspects in which we are interested arise out of the organization of the parts, independently of all but a few properties of the individual components. [22]. Describing AG agents by their *interface* means encapsulating and black boxing the actual internal implementation.

Definition 6.3 (Generalized agency Attributes):

GAG is defined via the following attributes:

- Name, that is used to label the problem domain; e.g., Electronic Commerce Agency.
- Framework name (optional); e.g., Electronic Commerce Agency for software goods.
- Intention, describing goal of the agency; e.g., social welfare of participant agents.
- List of participant agents, their classes (C) and/or types (T); e.g., search (C:data_agent, T:web_agent), catalog (C:data_agent), customer (C:PA_agent, T:stand_alone_agent).
- Procedures, that describe the problem and how the participant agents solve the problem; e.g., total customer support.
- Patterns, that describe the reasoning processes within an agent; e.g., issuing an offer (invoice).
- Description of participants collaboration; e.g., Message Sequence Charts for message passing among the participants.
- Critique, describing whether the GAG can achieve its goal based on procedures and patterns of participant agents.
- list of close or related GAGs and a description of their similarities, differences and trade-offs.

A library of customizable GAGs is under development.

7. Implementation and Application

Electronic Commerce (EC) is a potential application for the framework and techniques introduced in this paper.

EC model may be viewed as a set of *actors* using *media* performing business *actions*. The *actors* are providers (dealers, merchants, etc.), and consumers (users, information seekers). The *media* is composed of multimedia objects (documents, images, software, etc.) which represent goods or services. The *actions* are business processes, such as stock management and payment processing.

A unique feature of our EC project is unified system design, using GAG and AG agents for EC.

In our project, EC is viewed as a generalized agency (GAG) composed of 7 types of software agents, i.e., *customer, search, catalog, manufacturer, dealer, delivery* and *banker* agents, distributed over the Internet and interact with each other. We consider modeling and implementing AG agents acting as personal representative on behalf of casual and non-expert human actors. Actions (i.e., business processes) are viewed as inter-agent processes. Knowledge-based decision making and knowledge-sharing are modeled using the *symbol structure* and implemented as intra- and inter-agent processes. Detailed discussion on system implementation is given in [7].

Other applications using the framework and techniques described in this paper, such as a multiagent learning language (MALL) for agent communication [20], a multiagent distributed fault diagnosis system for wide area networks [10], a multiagent intelligent tutoring system (ITS) system [11], a computer aided software engineering (CASE) tool for object oriented software design (OOExpert) [18] are under investigation and development.

8. Conclusion

In this paper we have devised methods of devising, representing and incorporating organizational knowledge in multiagent system design. A distinguishing point was attributing the organizational knowledge to at least a pair of AG agents. The private knowledge of an agent is represented by *symbol structure* and agents can share their knowledge using combination, specialization and generalization methods. We gave a coherent view of agent interaction, i.e., cooperation, coordination and competition by defining Generalized Agencies (GAG) and Generalized Agents (AG). Developing a catalog of GAGs and use them in systematic multiagent design is the next step. Applications using the framework and techniques described here are under development.

References

- [1] J.M. Bradshaw (Edt.), "Software Agents," *MIT Press*, 480 p., (1997).
- [2] Bradshaw J.M., et al., "Agents for the Masses?" *IEEE Intelligent Systems and Their Applications*, Vol. 14, No. 2, pp. 53-63, (1999).
- [3] Brenner W., Zarnekow R., Wittig H., "Intelligent Software Agents," Springer Verlag, Berlin, (1998).
- [4] Carley K.M., and Prietula M.J., "Computational Organization Theory," *Laurence Erlbaum Associates*, Hillsdale, NJ, (1994).
- [5] Carley K.M., and Gasser L., "Computational Organization Theory," in *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, Weiss, G., ed., pp. 299-330, *MIT Press*, (1999).
- [6] Chandrasekaran B., Josephson J.R., and Benjamins V.R. "What Are Ontologies, and Why Do We Need Them?," *IEEE Intelligent Systems and Their Applications*, Vol. 14, No. 1, pp. 20-26, (1999).
- [7] Far B.H., et al., "An Integrated Reasoning and Learning Environment for WWW Based Software Agents for Electronic Commerce," *Transactions of IEICE*, Vol.E81-D No.12, pp.1374-1386, (1998).
- [8] Farquhar A., et al., "The Ontolingua Server: A Tool for Collaborative Ontology Construction," *Knowledge Systems Laboratory*, KSL-96-26, (1996).
- [9] Genesereth M.R., Fikes R.E., "Knowledge Interchange Format Version 3.0 Reference Manual," *The DARPA Knowledge Sharing Initiative*, (1992).
- [10] Hajji H., Far B.H., Koono Z., "Intelligent Agents for Probabilistic Diagnosis-Restoration Planning in WAN," *Technical Reports of IEICE*, AI98-56, pp. 17-24, (1998).
- [11] Hashimoto A.H., Hajji H., Far B.H., "A Probabilistic Approach for Student Modeling Task," *13th Annual Conference of Japanese Society for Artificial Intelligence*, JSAI' 99, pp. 68-69, (1999).
- [12] Huhns M.N., Singh M.P., "Readings in Agents," *Morgan Kaufmann Publishers*, (1997).
- [13] Huhns M.N., Stephens L.A., "Multiagent Systems and Societies of Agents," in *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, Weiss, G., ed., pp. 79-120, *MIT Press*, (1999).
- [14] Jennings N.R., Sycara K., Wooldridge M., "A Roadmap of Agent Research and Development," *Autonomous Agents and Multiagent Systems*, Vol. 1, pp. 7-38, (1998).
- [15] Jennings N.R., Wooldridge M., Edts., "Agent Technology, Foundations, Applications and Markets," *Springer-Verlag*, (1998).
- [16] "DRAFT Specification of the KQML Agent-Communication Language," *The DARPA Knowledge Sharing Initiative*, (1997).
- [17] Pylyshyn Z.W., "Computation and Cognition, Towards a Foundation of Cognitive Science," *MIT Press*, (1989).
- [18] Wahono R.S., Far B.H., "OOExpert: Distributed Expert System for Automatic Object-Oriented Software Design," *13th Annual Conference of Japanese Society for Artificial Intelligence*, JSAI' 99, pp. 456-457, (1999).
- [19] Sandholm T.W., "Distributed Rational Decision Making," in *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, Weiss, G., ed., pp. 201-258, *MIT Press*, (1999).
- [20] Soueina S.O., et al., "MALL: A Multi Agents Learning Language for Competitive and Uncertain Environment," *Transactions of IEICE*, Vol.E81-D No.12, pp.1339-1349, (1998).
- [21] Simon H.A., "Administrative Behavior", *Free Press*, New York, (1947).
- [22] Simon H.A., "Cognitive Science: The Newest Science of the Artificial", *Cognitive Science*, Vol. 4, pp. 33-46, (1980).
- [23] Smith R.G., "The Contact Net Protocol: High Level Communication and Control in a Distributed Problem Solver," *IEEE Trans. on Computers*, Vol. C-29, No. 12, pp. 1104-1113, (1980).
- [24] von Neumann J., Morgenstern O., "Theory of Games and Economic Behavior", *John Wiley and Sons*, (1944).