

Acquaintance Models for Integration-oriented Multi-agent Systems

Vladimír Mařík, Olga Štěpánková, Michal Pěchouček
The Gerstner Laboratory,
Czech Technical University,
Technická 2,
166 27 Prague 6, Czech Republic
Fax: +420-2-24357224
E-mail: {marik, step, pechouc}@labe.felk.cvut.cz

Abstract

The paper addresses problems of efficient knowledge representation and exploration of social knowledge enabling agents' integration into the multi-agent community. The proposed solutions are based on task decomposition, organization of negotiations, responsibility delegation and other aspects of social behavior in multi-agents systems aimed at integration of already existing software components. It is supposed that all the social knowledge is kept separated from the problem solving knowledge and agents' specific internal intelligence and that it is organized and administered in the acquaintance models located in the agents' wrappers. A specific tri-base acquaintance model is formalized and discussed throughout the paper. This model helps to optimize the communication traffic and to implement meta-reasoning processes. Practical applications of the acquaintance model in different fields are presented, acquired experience is discussed.

1. Introduction

The development in the recent decade has proven that the multi-agent paradigm represents a challenging framework for solving very complex tasks in a distributed way [9,11]. A multi-agent system (MAS) usually consists of a set of autonomous units capable of:

- independent operations aimed at meeting their **local goals**, and
- cooperative actions contributing jointly to the **global goal(s)** shared across the community.

There are various types of agent-oriented applications. Throughout this paper we will refer to a specific category of multi-agent systems that are designed and developed in order to allow intelligent, flexible and robust integration

of already existing software components. We will be addressing neither the issue of agents' specific internal intelligence, nor general reasoning machines, but we will be discussing agent's integration in the community and agents' social reasoning based on task decomposition, responsibility delegation, and corresponding organization of negotiations. The agents' abilities to communicate, mutually coordinate their actions, cooperate and share the global goals determine the level of their integration-oriented behavior. These abilities depend mainly on the extent and quality of knowledge available to the agents. In this paper, we will have in mind just the **knowledge-centered aspects** of functional integration in the communities of agents of diverse nature.

Knowledge – a true piece of evidence in which the agent believes [12] – can either:

- (1) guide agent's autonomous local decision making processes (aimed e.g. at providing an expertise or search in the agent's database) – this is what we call agent's (local) **problem solving knowledge**, or
- (2) express the other agent's behavioral patterns, their capabilities, load, experiences, commitments, knowledge describing conversations or negotiation scenarios – which we will refer to later as **social knowledge**¹.

Hereafter, when referring to knowledge we will primarily mean the agent's social knowledge.

Undoubtedly, the multi-agent systems should be equipped with a vast portion of knowledge to perform highly efficient cooperative behavior and to achieve global solutions. Such knowledge can be – in the extreme

¹ We will understand agent's knowledge about his own behavior, status and commitments, which is usually referred to as agent's **self-knowledge**, as a special instance of social knowledge. It is very often the case that the agent's self-knowledge is accessible to other agents in the community and it is a part of their social knowledge.

cases – stored either **centrally**, in a fully informed central unit, **or locally** owned by each of the agents. The latter case fits better the general visions how the multi-agents systems should be organized and implemented.

The main questions connected with the “local ownership” of the global knowledge are:

- What should be the reasonable extent of global knowledge administered locally, by individual agents?
- How much the agents should know about the global rules and knowledge ontologies?
- How much do they need to know about the particular cooperating “colleagues”?
- How to structure the locally stored knowledge to enable its efficient up-date and maintenance leading to reduction of the communication load in the multi-agent community?

Classical architecture of an agent separates his functional **body** containing the agent’s individual abilities from his **wrapper** that accounts for the inter-agent communication. In our integration domain we will place the encapsulated software application into the body together with the problem solving knowledge and specific inference algorithms. The body is assumed to have no awareness of the multi-agent community. The **wrapper** will contain knowledge structures and reasoning mechanisms required for communication, coordination, cooperation and integration with the rest of the community. We can further decompose the wrapper into two organizational layers (see Fig. 1):

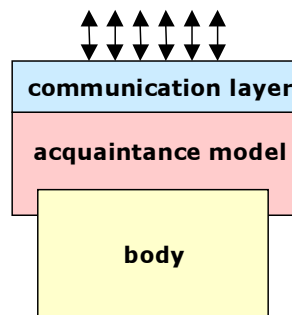


Figure 1 - An agent's body/wrapper architecture

- **communication** layer responsible for carrying out the required communication, it takes care for sending and receiving messages in a pre-determined format,
- **acquaintance** layer containing social knowledge and reasoning mechanisms needed for planning and organizing intelligent communication scenarios.

1.1. Acquaintance model

Models containing social knowledge are usually called the **acquaintance models**. They do represent an important category of instruments for efficient knowledge representation that supports the appropriate knowledge distribution across the multi-agent community.

An acquaintance model is a knowledge-based model of agent’s mutual awareness that collects agent’s knowledge about his collaborators and about suitable communication and negotiation scenarios. In the simplest possible form we can regard a *white-page* list as an instance of an acquaintance model. However we require some additional functionalities: an acquaintance model is supposed to maintain permanent, semi-permanent and non-permanent information about other agent’s services, knowledge, statuses, about potential negotiation scenarios, delegation principles etc. It is required that the acquaintance model will also contain certain knowledge about his own knowledge, status and intended activities. The corresponding part of this knowledge structure can be accessible to the collaborating agents and they maintain it in an identical form.

In our understanding, the agent’s body is not necessarily aware of the actual content of the acquaintance model: It is the acquaintance model as a part of the wrapper, which is responsible for exploring the social kind of information for delegating responsibilities and reasoning about other agents, etc. The real-life implementations of multi-agent systems usually don’t sharply distinguish where the social knowledge is located. Let’s consider all the social knowledge located outside of the body, more precisely let the acquaintance model be a part of the wrapper. This makes sense especially in the system integration oriented multi-agent systems.

In the communication or negotiation phase which should lead to certain degree of co-ordination or co-operation we can – in principle - distinguish among three simple brokering mechanisms, by means of which an agent can find the best suitable collaborating partner for delivering a required service. The mechanism based on the acquaintance model is one of them. These three mechanisms are based on:

- **broadcasting of requests** – in this case an agent sends requests for services to all members of the community and the best collaborator is selected from the subsequent replies,
- **facilitator** – where a request for service is sent to a certain central agent (facilitator) which administers all the data about the community members (yellow pages functionality) and forwards the request to an appropriate agent,
- **acquaintance model** – where each agent maintains certain amount of social knowledge

about the collaborating agents and thus he is aware of their actual capabilities – the agent to be contracted is then selected without any further communication.

While the first approach demands substantial communication traffic, the second approach lowers the amount of messages sent. On the other hand the second approach, unlike the former one, depends on a central agent and the community is therefore very fragile. By using the agents' acquaintance models we may design a compromise solution. Each agent maintains in his acquaintance model the part of the information administered by the facilitator (in the second approach) that he needs and that is accessible to him. Thus each agent takes over an appropriate part of the facilitator's knowledge.

Different types of acquaintance models have been used e.g. in the systems ARCHON [10], COVERAGE [3], PLEIADES [8] and others. Our approach was motivated mainly by the **twin-base** model presented in [1]. The twin-base model has been aimed at representing mainly the knowledge about capabilities, load, efficiency, reliability etc. of the cooperating agents. This are exactly the pieces of knowledge needed for creating integration architectures. The tri-base acquaintance model can be considered as a substantial extension of the twin-base model.

There are several additional questions that have to be considered when designing a quite particular implementation of an acquaintance model:

- What is the optimal extent of the collaborating environment (i.e. how many and which agents from the agent's neighborhood should be included into the model)?
- How detailed the social knowledge should be?
- How to keep separated various types of knowledge, namely *permanent* (addresses of the cooperating agents), *semi-permanent* (list of capabilities), *non-permanent* frequently changing properties (load, trust, reliability, etc.)?
- How to maintain knowledge to keep it as much up-to-date as possible?

Some of these questions are solved by a specific methodology for organization and administration of agent's mutual awareness developed in connection with the tri-base acquaintance model (3bA) [5]. This methodology extends the ideas used in the twin-base model and it basically includes:

- a) representation of permanent, semi-permanent and non-permanent knowledge in separate knowledge structures,
- b) an inference engine for exploring and processing the locally administered social knowledge,
- c) consideration of several types of social neighborhood,

- d) techniques for maintaining the knowledge up-to-date by revisions of the non-permanent part of knowledge (accomplished in the idle times of the system) leading to significant reduction of the communication traffic among the agents in the (usually time critical) phase of creation of collaborative scenarios.

In the following paragraph we will explain two key aspects of the tri-base model:

- **3bA model knowledge structures** – data collections that will appropriately represent an agent's collaboration space and
- **3bA model knowledge maintenance mechanism** – algorithms used to keep the acquaintance model as much up-to-date as possible.

2. Knowledge Structures of the 3bA Model

Prior to formalising the knowledge structures of the model let us introduce several primitives we will use throughout the course of explanation. Let Θ be the set of all agents within the community and S a set of all tasks the community members are able to take responsibility for. Results of our work will be illustrated on a very typical communication scenario – a *request for decomposition*. If an agent is requested to decompose a task it shall detect the best possible collaborators (based on his knowledge of decomposition) and contract these with parts of the original request. Hereafter we will talk about such agents. This is why we may refer to a set S as a collection of all tasks the agents are able to decompose.

For each agent $A \in \Theta$ let

- $\alpha(A) \subseteq \Theta$ be the *agent's total neighbourhood*, a set of agents the agent A is aware of,
- $\beta(A) \subseteq S$ be the set of all tasks the agent A is able to decompose,
- $\gamma(T)$, contains all possible plans for decomposing the task $T \in S$. Plan for the task T is in the form $\langle T, S, O, C \rangle$, where S is a set of subtasks which ensure completion of the task T provided that their processing meets precedence constraints O and applicability constraints C .
- $\omega(A, T) \subseteq \gamma(T)$ contains those plans for the task T an agent A knows about (if $T \notin \beta(A)$ then $\omega(A, T) = \emptyset$).

The following sets provide time dependent information. Let

- $\varepsilon^t(A) \subseteq \alpha(A)$ be the agent's current *cooperation neighbourhood*, a set of agent's A collaborators at the time instant t ,
- $\tau^t(A) \subseteq \beta(A)$ contain the tasks being solved by the agent A in a time instance t and

- $\pi^l(A) \subseteq \beta(A)$ be a collection of tasks an agent A is supposed to have pre-prepared in advance in time instance t

Within the tri-base model each agent maintains three knowledge bases where all the relevant information about the rest of the community is stored. We can distinguish among:

Co-operator Base (CB) maintains permanent information on co-operating agents (i.e.: their address, communication language, and their predefined responsibility). This type of knowledge is expected not to be changed very often. $CB(A)$ is then defined as

$$CB(A) \equiv \{\langle B, \text{Addr}(B), \text{Lang}(B), \beta(B) \rangle\}_{B \in \alpha(A)} \text{ where}$$

$\text{Addr}(B)$ specifies agent's the address, $\text{Lang}(B)$ language it communicates, as already mentioned $\beta(B)$ is a set of tasks the agent B accounts for and the set $\alpha(A)$ denotes members of the agent's A scope of the community.

Task Base (TB) stores in its *problem section* (PRS) general problem solving knowledge – (i) information on possible decompositions of the tasks to be solved by the agent and (ii) in its *plan section* (PLS) it maintains the actual and most up-to-date plans on how to carry out those tasks, which are the most frequently delegated to the agent - the owner of the task base, those denoted as $\pi^l(A)$. Formal definition of the $TB(A)$ is then

$$TB(A) \equiv \langle \text{PRS}(A), \text{PLS}(A) \rangle, \text{ where}$$

$$\text{PRS}(A) \equiv \{\omega(T, A)\}_{T \in \beta(A)} \text{ and}$$

$$\text{PLS}^l(A) \equiv \{\langle T, \langle \{ \langle s, B \rangle \}_{s \in S}, O, C, \text{Trust}(T) \rangle \rangle\}_{T \in \pi^l(A)},$$

where for any $\langle T, \langle \{ \langle s, B \rangle \}_{s \in S}, O, C, \text{Trust}(T) \rangle \rangle \in \text{PLS}^l(A)$ there exist O_I, C_I such that following constraints are met $\langle T, S, O_I, C_I \rangle \in \text{PRS}(A)$, $B \in \beta(A)$, $s \in \beta(B)$ and as C is a specialisation of C_I reflecting the considered allocation of the tasks $s \in S$, O is refinement of O_I and both O and C are valid.

State Base (SB) stores in its *agent section* (AS) all information on current load of co-operating agents. This part of the state base is updated frequently and informs the agent who is busy and who is available for collaboration. In the *task section* (TS) there is stored information on statuses of tasks the agent is currently solving. Formal description of the $SB(A)$ of agent A is thus

$$SB(A) \equiv \langle \text{AS}(A), \text{TS}(A) \rangle, \text{ where}$$

$$\text{AS}(A) \equiv \{\langle B, \text{Cap}(B), \text{Load}(B), \text{Trust}(B) \rangle\}_{B \in \beta(A)} \text{ and}$$

provided that agent's B capability has the form of $\text{Cap}(B) \equiv \{\langle T, \text{Cost}(T) \rangle\}_{T \in \beta(B)}$, overall agent load is $\text{Load}(B)$, and trust in this information in $\text{Trust}(B)$.

$\text{TS}(A)$ contains relevant information on all the tasks agent A agreed to supervise recently. This set is denoted by $\tau^l(A)$. Formally

$$\text{TS}(A) \equiv \{\langle T, \text{Dec}(T), \text{State}(T), \text{Trust}(T) \rangle\}_{T \in \tau^l(A)},$$

where decomposition $\text{Dec}(T)$ is taken from the $\text{PLS}^l_I(A)$ at the moment of contract (time t_1). $\text{State}(T)$ partitions subtasks from $\text{Dec}(T)$ into three parts: subtasks finished, actually processed, and the rest. The record is complemented with the trust value $\text{Trust}(T)$ denoting trust in the plan of the task T .

2.1. Knowledge Maintenance of the 3bA Model

Let us first comment how the knowledge is maintained in the **cooperator base**. As we have already mentioned, this base collects knowledge of rather permanent nature and we do not expect to update it very often besides the register phase. Once a new agent registers with a community (by means of contacting the **facilitator** agent), facilitator replies the newcomer by providing information about the community members. In addition to this, facilitator informs other agents about the newcomer and thus invokes an update of the CB (in the form of a record append).

The **state base**, which is supposed to model the actual state of the collaborating agents, is maintained by a simple **subscribe/advertise** mechanism. After parsing the problem solving knowledge (in PRS), each agent identifies possible collaborators and subscribes them for reporting on their statuses. Let us denote the subscribing agent **subscriber** and an agent who was subscribed as a **subscribed**. There are two ways how to maintain knowledge in the bases of all agents in the community. The subscriber can keep advertising its load, capabilities, task completion times and costs estimates either periodically or whenever either of these changes.

The subscribe/advertise mechanism facilitates the subscriber to make the best decision with no further communication.

The **task base** is kept up-to-date by **periodic revisions** of the pre-prepared plans in the PLS. Such a revision represents verification/modification of the plan by exploring the information kept up-dated in both the co-operator and state bases. The knowledge contained in the PRS can be maintained e.g. by the meta-agents.

Hereafter we will refer to a **contractor** as an agent who contracts another agent with a request. A **contractee** is an agent who was contracted by another agent with respect to the request. The **contractor** is supposed to select an optimal plan from the $\text{PLS}^l(A)$, where an appropriate amount of plans prepared in advance is stored. By this it does not need to contract peer agents in order to find out the most appropriate (optimal) offers for

further problem delegation. Knowledge stored in the PLS will help the agent to decide by itself. It is obvious that limiting the communication among agents will in its own way decrease the computational complexity of the entire problem. The price we have to pay for this, is a communication increase among agents when updating the entire model.

2.2. Knowledge Improvement

Besides the knowledge maintenance there is considered a special approach/technique aimed at permanent knowledge improvement. This approach is based upon meta-reasoning principles and supposes the introduction of a meta-agent. The meta-agent observe activities of the community, collect relevant pieces of information and, consequently, try to draw certain assumptions about the individual agent's behavior. They can learn a lot about how to enhance the community's functional efficiency and are able to provide advice to the agents. This advice has – as a rule – a form of proposed changes in the knowledge kept in 3bA models of the individual agents.

In principle, the meta-agents operate in three key phases, namely (i) data acquisition phase, (ii) inference phase (iii) community knowledge revision phase. During the last phase, the content of the knowledge bases in the 3bA models of the agents is modified by messages sent by the meta-agents.

The meta-knowledge owned by the meta-agents can be organized in the form of a special 3bA model as well. The body of a meta-agent contains mainly databases and machine learning algorithms.

There is one significant difference between a meta-agent and any kind of "central agents" like brokers, facilitators or mediators: The existence or non-existence of a meta-agent is not life-critical for the community. The community is, in principle, capable to operate without any meta-agent, but the knowledge applied (and, consequently the overall community performance achieved) cannot be improved.

2.3. Saving communication traffic by means of the 3bA

The prime motivation for investigating the concept of the 3bA model was saving communication traffic in a multi-agent system. Though it may look like the communication flow is not an issue, in many practical applications the amount of transmitted messages reflects the computational complexity of the problem, the system has been designed to solve.

Specific contribution to communication efficiency of the 3bA model has been tested. We have compared

required communication of two types of agents – (i) broadcasting agents that use classical contract-net-protocol and (ii) tri-base agent that uses the collection of the social knowledge [7].

The broadcasting agent will broadcast a "tender" for collaboration. Possible collaborators subsequently reply respective collaboration offers that will reflect their capability and current status (e.g. occupation). In the last phase the broadcasting agent will select the best (e.g. cheapest) collaborator and contracts him.

The tri-base agent will analyse its acquaintance model, where he stores permanent (offered services) and non-permanent (actual status) information about all the agents that belong to his social

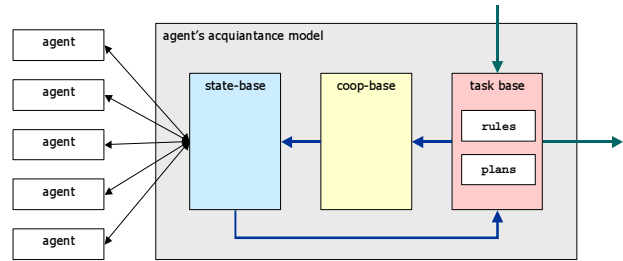


Figure 2 - Tri-base Acquaintance Model

neighbourhood. According to this information, he selects the best contractor without any other communication. The price one has to pay for this saving is substantial communication required for maintenance of the acquaintance model. When mediating whether it pays off to use the tri-base approach we have to find out how often do changes of agents' statuses occur, how much of communication we will save, etc.

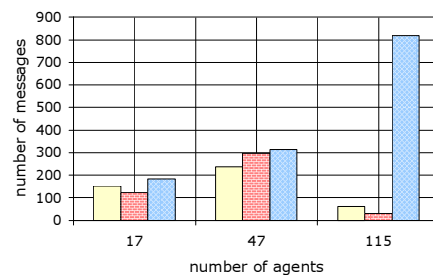


Figure 3 Tri-base Model Savings and Maintenance

The graph in the Figure 3 shows the average communication requirements for the identical request in three different communities (the x axis specifies number

of agents and the y axes draws number of messages required for solving the request). The white (left) bar shows number of messages for the tri-base agent solving the request and the light grey (right) bar illustrates the broadcasting agent. The dark grey (middle) bar stands for messages required for the tri-base model maintenance. The graph shows that tri-base reasoning brings substantial savings in multi-agent systems with complex communities.

3. Applications

The applicability of the tri-base methodology will be documented when several practical applications in various applications domains will be presented.

The first system has been the ProPlanT system [4] aimed at production planning in manufacturing facilities for project-oriented manufacturing (not a mass production facility). The production facility itself is represented by a three-layered agent structure consisting of production management agents (PMA) on the top, production planning agents (PPA) in the middle layer, and production agents (PA) at the bottom of the hierarchy. This internal, company structure communicates with external customer agents. The knowledge is permanently improved by applying the meat-agents. The ProPlanT system was implemented in C++ and explores the KQML communication standards.

The experience gained with the ProPlanT system have been explored in the design of the ExPlanTech system for the supply chain management tasks. ExPlanTech uses three types of agents: customer agents, decomposition agents (who are responsible for the task decomposition) and resource agents (representing material and transportation resources). ExPlanTech has been implemented in the open source platform JADE (<http://sharon.cse.it/projects/jade/>) and is fully FIPA-compliant.

The other application of the tri-base methodology represents the CPlanT system for automatic coalition formation in peace-keeping operations. After a detailed problem analysis, the following 4 types of agents have been designed, namely: in-need agents (representing the suffering entities, like villages, schools etc.), resource agents (representing material and transportation resources), humanitarian agents (playing the role of entities eager to help), and coordination agents ensuring efficient coordination of the humanitarian activities optimizing the exploration of available resources. The tri-base model has been found as convenient as it enables to separate chunks of knowledge which are public, or available just to several partners, or totally private. These highly specific demands required a certain "refinement" of the internal knowledge structures in the tri-base model.

The CPlanT system has been implemented in Allegro Common Lisp and is fully FIPA-compliant.

4. Conclusions

As it has been recognized, the 3bA model offers a general framework for representing knowledge in the agents' wrappers. This framework efficiently supports

- flexible internal structuring of the community (functional differentiation of agents, their grouping, hierarchical or heterarchical structuring of the agents or their groups etc.),
- reflective (and self-reflective) ways of reasoning provided the agents contain the information about their own capabilities, properties etc. in the same form as they do for other agents,
- meta-reasoning (using meta-agents equipped with the 3bA model structures as well), which proved important e.g. for solving of reconfiguration problems,
- utilization of machine learning algorithms, aimed at improvement of the system's functionality using its past or recent experience.

The 3bA methodology can be considered as a framework or guidelines for designing multi-agent systems on general level. At least, it helps to categorize and separate knowledge of diverse nature, to define scenarios for maintaining the knowledge and keeping it up-to-date as well as to analyze, classify and modularize the processes of exploring the social knowledge. The technology of maintaining the social knowledge efficiently, without any enormous communication requirements belongs to the strongest novel features of this approach.

There are many advantages of the 3bA approach. Let's summarize just some of them:

- Maintenance of knowledge can be carried out in a very efficient way, the knowledge can be up-dated and improved in the idle communication slots
- Meta-agents accomplishing meta-level reasoning and enabling certain degree of self-reflection can explore the same 3bA architecture as the "ordinary" agents with advantage
- The changes in knowledge in the case of community reconfiguration can be carried out in a very smooth and natural way

The analysis of practical applications in different fields shows that various global community functional architectures can be constructed by means of the 3bA acquaintance model, e.g. the multi-level hierarchical architecture has been used in production planning, totally flat architecture in the supply chain management area or hierarchic architecture when e.g. supply chain

management being directly linked to the production planning of the supplier.

In many application areas we can see that there is a group of customer agents negotiating and communicating with the group of resource agents through the group of mediating agents (Decker, 1997), (McGuire, 1993). These are called PMA and PPA agents in ProPlanT, decomposition agents in ExplanTech and coordination and humanitarian agents in the field of coalition formation. Existence of these three groups of agents and their comparatively “flat” and simple linkages are typical for many areas. These observations don’t exclude the possibility to structure the agents within a group in a much more complicated way (we can see e.g. the internal hierarchical structuring of the PMA and PPA agents in ProPlanT system).

The 3bA model should be considered as a very general acquaintance model which outlines the rough organization of knowledge into three separate knowledge bases as well as the general mechanisms for keeping the knowledge up-dated. The generality of the approach doesn’t exclude specific refinements of the general structures as we have documented e.g. in the case of the coalition formation problem. Such specific modifications for specific applications would represent the further trend.

5. References

- 1 Cao, W., Bian, C-G., and Hartvigsen, G.: Achieving Efficient Cooperation in a Multi-Agent System: The Twin-Base Modelling. In: *Co-operative Information Agents* (Kandzia, P., Klusch, M. eds.), LNAI No. 1202, Springer Verlag, Heidelberg, 1997, 210–221.
- 2 Decker, K., Sycara, K., and Williamson, M.: Middle Agents for Internet. In: *Proceedings of Int. Joint Conference on Artificial Intelligence 97*, Japan, 1997
- 3 Lamb, N., and Preece, A.: Verification of Multi-Agent Knowledge-Based Systems. In: *Proc. ECAI-96 Workshop on Validation, Verification and Refinement of KBS*, 1996.
- 4 Mařík, V., Pěchouček, M., Lažanský, J., and Roche, C.: PVS’98 Agents: Structures, Models and Production Planning Application. *Robotics and Autonomous Systems*, vol. 27, No. 1-2, Elsevier, 1999, 29–44.
- 5 Mařík, V., Pěchouček, M., Štěpánková, O., and Lažanský, J.: ProPlanT: Multi-Agent System for Production Planning. *Applied Artificial Intelligence*, vol. 14, No.7, 2000, 727–762.
- 6 McGuire, J., Kuokka, D., Weber, J., Tenebaum, J., Gruber, T., and Olsen G.: SHADE: Technology for Knowledge-Based Collaborative Engineering, *Concurrent Engineering: Research and Applications*, 1(3), 1993.
- 7 Pěchouček, M., Mařík, V., and Štěpánková, O.: Towards Reducing Communication Traffic In Multi-Agent Systems. *Journal of Applied Systems*, Cambridge International Science Publishing, vol. 2, No.1, 2001.
- 8 Sycara, K.: Intelligent Agents and Information Retrieval. Unicom Seminar on *Intelligent Agents and their Business Applications*, 8-9 November 1995, London, 143-157.
- 9 Weiss, G. (editor): *Multi-Agent Systems: Modern Approach to Distributed Artificial Intelligence*. The MIT Press, Cambridge, MA, 1999.
- 10 Wittig, T. (editor): *ARCHON: An Architecture for Multi-agent System*. Ellis Horwood, Chichester, 1992.
- 11 Wooldridge, M., and Jennings, N. R.: Intelligent agents: Theory and Practice. *The Knowledge Engineering Review*, 10(2), 1995, 115–152.
- 12 Wooldridge, M.: *Reasoning about Rational Agents*, The MIT Press, Cambridge, MA, 2000.