MULTI-AGENT SYSTEMS AS AN INTEGRATION ENVIRONMENT FOR CLASSICAL AND AI COMPONENTS

Lenka Lhotska
Gerstner Laboratory, Department of Cybernetics, Czech Technical University in Prague, Faculty of Electrical Engineering,
Technická 2, CZ-166 27 Prague 6, Czech Republic
Phone: +420-2-24353933, Fax: +420-2-24311081
E-mail:lhotska@fel.cvut.cz

Olga Stepankova
Gerstner Laboratory, Department of Cybernetics, Czech Technical University in Prague, Faculty of Electrical Engineering,
Technická 2, CZ-166 27 Prague 6, Czech Republic
Phone: +420-2-24357233, Fax: +420-2-24311081
E-mail: step@labe.felk.cvut.cz

ABSTRACT

The multi-agent technology has been recently considered to be much more suitable for creating open, flexible environment able to integrate software pieces of diverse nature written in different languages and running on different types of computers. It enables to design, develop and implement a comparatively open multi-agent environment suitable for efficient creating of complex knowledge-based or decision support systems. Such an environment is able to integrate geographically distributed knowledge sources or problem solving units. The task under consideration is located just on the borderline between Software Engineering and Artificial Intelligence. The idea of software integration based on efficient communication among parallel computational processes as well as that of the open architecture (enabling to add new elements without any change in the others) has been provided by the Software Engineering area. On the other hand, the multi-agent approach stemming from the theory of agency, from behavioral models of agents and methods of agentification of stand-alone programs can be considered as a contribution of Artificial Intelligence. Multi-agent systems have useful properties, such as parallelism, robustness, and scalability. Therefore they are applicable in many domains which cannot be handled by centralized systems, in particular, they are well suited for domains which require, for example, resolution of interest and goal conflicts, integration of multiple knowledge sources and other resources, time-bounded processing of very large data sets, or on-line interpretation of data arising in different geographical locations.

INTRODUCTION

Recently multi-agent systems have become one of the dominating topics of research in artificial intelligence. This part is gradually separating from distributed artificial intelligence as an independent discipline, based on research results both in other parts of artificial intelligence and in the area of computer science. From AI the multi-agent systems take over methods of knowledge representation and utilization, methods of formalization of knowledge models using expression tools of special logics and algorithms of machine learning. From computer science the multi-agent systems draw knowledge of communication tools, especially on lower levels. Great interest in multi-agent paradigm is quite natural because a number of software systems have reached such a degree of complexity that it is impossible to control and operate them as monolithic systems. Therefore the effort of decomposition of such systems is a natural procedure. The complex systems should be decomposed into natural functional units that solve partial tasks relatively autonomously and communicate in inevitably minimum range only with the aim to co-ordinate their activities with other units with which they share a global goal.

Most of nowadays existing multi-agent systems are systems developed ad hoc while they are making use of only the simplest, usually reactive, models of behaviour. Agent architectures and global architectures of multi-agent systems are designed more or less intuitively without utilizing deeper formalization that would enable to plan and correctly realize even more complex scenarios requiring exactly co-ordinated co-operation of a greater number of agents. Similar situation is in communication among agents. Agent communication languages only gradually start to utilize open communication standards. However using these standards is a very important requirement for mutual interoperability and reusability of multi-agent solutions, which is a substantial part of multi-agent system practical potential.

Many real world problems such as production planning, supply chain management, engineering design, intelligent search, medical diagnostics, robotics, etc. are naturally distributed. Hence multiagent systems (Weiss, 1999), (Wooldridge, 1995), (Huang, 1995) offer efficient problem solving platform. They eliminate limitations on the processing power of a single monolithic system. Distribution also brings inherent advantages of distributed systems, such as scalability, fault-tolerance, parallelism, robustness, etc. However, there are questions connected with mutual behaviour of the agents, of ownership of global knowledge, of structure and content of this knowledge. The applicability of agent architecture developed in the Gerstner Laboratory has been verified by applications in several areas, e.g. in the field of production planning and scheduling (ProPlanT system), supply chain management (ExPlanTech project), coalition formation (CPlanT system) (Marik, 2002), (Agents, 2002).

Most of the application domains are characterized by distributed data, information, knowledge, and competence. In addition, all three components (data, information, knowledge) may have different nature: descriptions in natural language, 2D images, measured signals, results of various tests or measurements (usually lists of numbers). They are stored on different media: sheets of paper, photographies, slides, electronic files, books (when considering "classical" knowledge), sometimes personal communications. Usually they are not available in a single place at a particular moment. This distribution represents a major problem when decisions have to be made in a timely fashion. Knowledge, decision making, planning, and actions are distributed functionally, geographically, and temporally as well. There exists a requirement of information flow among all participating subjects with the aim to satisfy the global goal – successful solution of the defined task. Of course, it is not usually predictable in extend and structure but it develops in time due to new knowledge and reactions. This requires high flexibility of supporting systems. To satisfy these requirements and provide adequate decision support, the use of intelligent software support is becoming increasingly desired. The agent technology offers an efficient and natural solution because it corresponds to main properties of most application domains, as for example industrial production or medical domain, namely distribution of information, problem-solving capabilities, resources, and responsibilities, decision-making with incomplete information, iterative refinement of plans.

MOTIVATION FOR APPLICATION OF MULTIAGENT SYSTEMS

In recent years there has been a growing interest in the application of agent-based systems in both industrial and service domains. Some domains in which agents have already been considered or even successfully implemented are the following: information retrieval from distributed information sources (Gomoluch, 2002); decision support systems for monitoring and diagnosis tasks (Larsson, 1998); distributed planning and scheduling (Říha, 2002); electronic business (Müller, 2002). This list serves as an illustrative example since the complete list would be much longer.

Considering the whole life-cycle nearly in any application domain, it is possible to identify (at least) five separate areas that can be computer supported, namely diagnostics, prediction, monitoring, information processing, workflow management and planning and scheduling. Let us describe briefly these areas and potential or existing utilization of computer support in them.

Diagnostics as a process of identifying a cause of a problem from its signs and symptoms is the most obvious area for application of computer support. The systems can help to focus the attention to the most probable causes, to suggest other special examinations or measurements, etc. At present, there exist various knowledge-based systems for decision support in diagnostics, in medical diagnostics see for example (http://medexpert.imc.akh-wien.ac.at/start.html). Many of them are used for differential diagnostics. Many of these systems are routinely used as stand-alone systems.

Prediction means in medicine inference regarding future disease development after application of certain treatment; in technical domain inference regarding future development of e.g. energy consumption with respect to several aspects (weekday, daytime, weather forecast, etc.). Prediction as well as diagnostics require vast amount of knowledge and experience on the user's side. Prediction may be supported by various tools as simulation systems or some of the machine learning methods (Klema, 2000).

Monitoring is life critical activity in intensive care units where delayed information can decide about patient's survival. Therefore it requires (at least partial) real-time data processing and evaluation. In this context, computer support represents significant time saving of personnel. Such a system can warn in time if there is suspicious development of one or more followed patient's parameters that would not lead to alarm in "classical" device and thus to initiate an appropriate action. The same description holds for monitoring of a technological process in an industrial plant where delayed information can cause high financial loss or lead to extreme damages (e.g. in chemical industry).

Information processing represents a big problem in nearly all application domains. Isolated “island” solutions are typical as many stand-alone systems were developed for particular tasks. Probably medicine is the most obvious example. There exist high degree of distribution; great extend of knowledge, and heterogeneity of information (findings, images, treatment protocols, laboratory results). If this information is to be used efficiently in diagnostics and treatment it must be easily accessible and must be as consistent as possible.

Workflow management and planning and scheduling requires high degree of co-operation and communication. There exist already practical examples of successfully implemented systems (Říha, 2002). Computer supported solutions promise increasing efficiency, and decreasing costs.

All these areas can benefit from new options offered by the modern information technologies: in diagnostics and prediction there is possible to involve agents who can search for similar cases appearing elsewhere. However, it is necessary in such a case that the agent is able to generate a proper query and these cases must be accessible via Internet. Another possibility is to use agents for preparation and evocation of electronic consilium.

SOFTWARE AGENTS

A multi-agent system is a collection of independent, autonomous agents that communicate, co-operate and co-ordinate their activities with the aim to reach solution of a complex task. Heterogeneity of individual agents and integration of legacy systems are further basic characteristics that are advantageous for many applications. An agent is usually defined as an autonomous software entity that receives inputs and interacts with its environment (including other agents), performing tasks in the pursuit of a set of goals. By an agent, we mean a software entity that exhibits the following properties (Wooldridge, 1995):

Autonomy: Agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state.

Social ability: Agents interact with other agents (and humans) via some kind of agent-communication language when they recognize necessity of such communication (usually with the aim to complete their own problem solving and to help others with their activities).

Reactivity (responsiveness): Agents perceive their environment (physical world, a user, a collection of agents, the Internet, or combination of all mentioned entities) and respond in a timely fashion to changes that occur in it.

Proactiveness: Agents do not simply act in response to their environment, they are able to exhibit goal-directed, opportunistic behavior and take the initiative when appropriate.

The strengths of Agent Technologies (mainly proactivity and autonomy) make such technologies well-suited for both technical and non-technical applications. We will summarize here only some of their arguments:

- the capability of agents to anticipate pro-actively the information needs of users;
- their support of synchronous and asynchronous communication among parties;
- their suitability to support distributed decision making;
- their ability to adapt to unpredicted situations;
- their capability to adapt the services to the user needs.

Although each agent has a different task in the system and therefore requires different kind of reasoning capabilities, all of them share a common basic structure. The characteristics of the environment have a great influence in the way agents have to be designed. Let us summarize the most important characteristics:

If the environment is persistent, then the agents should be constantly running without interruption.

If the environment is asynchronous, then messages can arrive at any time. An agent should be always ready to deal with new messages.

According to the application domain we can identify different types of tasks. However, in general we have to consider that the priority of tasks may change along time and that tasks may be cancelled as well. The priority of a task increases as long as the deadline to finish it is getting closer. It is important to take into account that how and when this priority should change is task dependent. This point and the previous one implies that an agent cannot be blocked waiting for an answer or spend a lot of time performing a single task. Periodically it has to check the environment (e.g. the message queue) to decide which is the most sensible thing to do next. To do that, agents use threads to parallelize task execution. Cancelling a task implies not only stopping its execution but also taking some „cleaning“ actions that will be different depending on the current progress of the task.

There exist many different practical solutions and architectures of multi-agent systems. Typically we can identify three large groups: multi-agent systems in which no agent has any knowledge of other community members and has to use broadcasting whenever it wants to send any information; multi-agent systems using a central agent (facilitator) that serves as a mediator, although this approach is frequently used it has many disadvantages of a central element; multi-agent systems using social models that represent a compromise solution between the first two alternatives. Each agent maintains social model of its environment, models of behaviour of cooperating agents, their load and readiness to cooperate. Representatives of social models are twin-base model (Cao, 1996, 1997) and tri-base model (Pěchouček, 2000).

TRI-BASE ACQUAINTANCE MODEL

The basic architecture of an agent in the system consists of a functional body (usually a stand-alone program with a well-defined functionality) and a wrapper (which is responsible for involvement of the agent into the community of agents) – see figure 1.

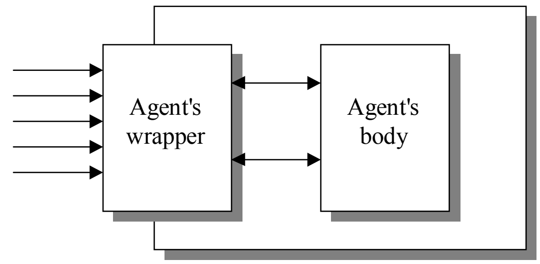

Figure 1. Structure of an agent

The tri-base acquaintance (3bA) models are encoded in agents' wrappers (see figure 2). The 3bA models have several important purposes:

- to limit explosive communication in multiagent system;
- to ensure immediate reply in time-critical situation;
- to generate and maintain databases of information sources.

Within the 3bA model each agent maintains three knowledge bases where all the relevant information about the rest of the community is stored, namely the Co-operator Base, the Task Base, and the State Base.

The Co-operator Base (CB) maintains permanent information on co-operating agents, i.e. their address, communication language, and their predefined responsibility (including information about required form of input data for agent's body). This may speed up process of selection of a proper agent that is able to perform required task since in all areas of the life-cycle there is usually used large volume of input data of different nature and form. This type of knowledge is expected not to be changed very often.

The State Base (SB) contains the information about collaborating agents, i.e. about their current state. The SB stores in its agent section (AS) all information on current load of cooperating agents. This part of the state base is updated frequently and informs the agent which of the collaborating agents are busy and which of them are available for collaboration. In the task section (TS) there is stored information on statuses of tasks the agent is currently solving. For example, if the agent is to process data that must be pre-processed by another agent, the former agent must know whether the latter has already started/finished the pre-processing.

The Task Base (TB) has two sections: problem section and plan section. In the plan section (PLS) it maintains the actual and most up-to-date plans on how to carry out those tasks,

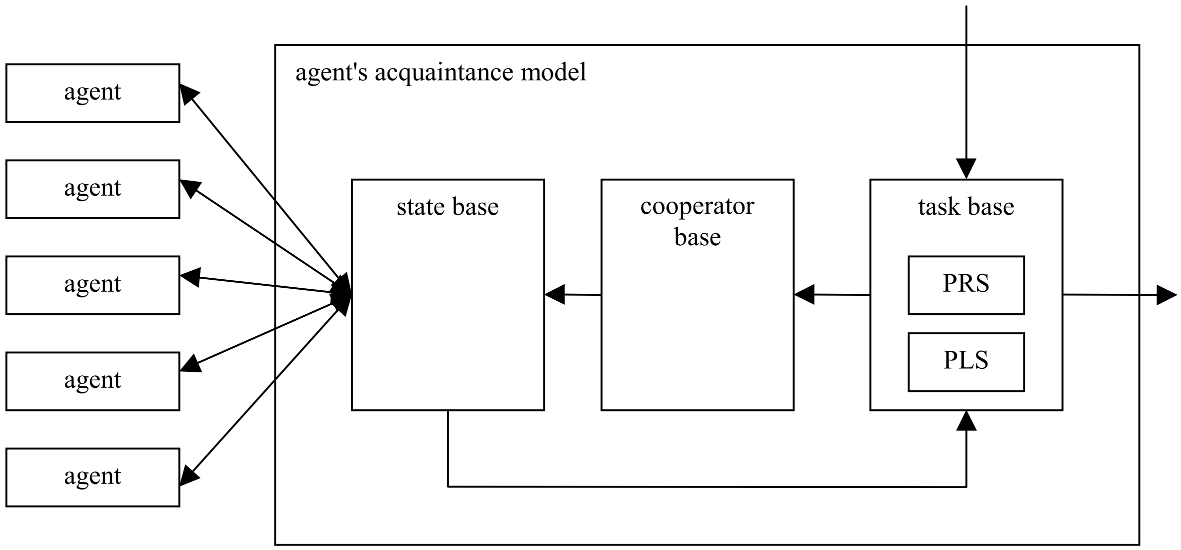

Figure 2. Tri-base acquaintance model

which are the most frequently delegated to the agent. In addition, it maintains information about the most suitable chains of agents that can perform certain tasks from collecting, pre-processing to evaluation. This information is updated by the metaagent that learns from successful and failed cases in the past (see below). The Task Base stores in its problem section (PRS) general problem solving knowledge on possible decision making with respect to input data type and expected output. In case of an agent responsible for certain data pre-processing the task base will contain knowledge about data types that can be pre-processed by the agent's body and about the results, which can be obtained in this way. It may contain knowledge about possible outputs of the agent's activity, namely whether the output represents intermediate results that should be sent for further processing or whether it represents final results that should be sent to the user. In case of an agent responsible for evaluation of pre-processed data its task base will contain information about data types that may arrive at its input and about procedure for checking data consistency. Considering time critical applications, response time of agents is important as well. Therefore information about average, maximum and minimum response time of the agent is attached to description of tasks the agent's body is able to execute. If the response is required in shorter time than the agent is able to deliver then another agent must be found that is able to deliver response in required time. Depending on type of the task, suboptimal solution delivered in shorter time can be preferred to late optimal solution. For example, classification using neural network or decision tree is usually faster than using case-based reasoning.

How is the knowledge maintained and updated in individual bases? As we have already mentioned, the co-operator 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 a central agent – facilitator that administers all the data about the community members), the facilitator replies the newcomer by providing information about the community members. In addition, it informs other agents about the newcomer.

The state base, which models 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. 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 updated in both the co-operator and state bases – this update can be done whenever the agent finds the time for that (idle time activity). The knowledge contained in the PRS can be maintained e.g. by the meta-agents.

Content of all three bases is dependent on the tasks the agents are supposed to solve. In case of planning agents the dominant role is played by knowledge of task decomposition and responsibility delegation. In case of configuration agents the knowledge stored in the task base can be used to lead the communication scenarios. The diagnostic agents contain the social knowledge about data sources, about the process of finding appropriate data and about the current progress in required data processing by the other agents.

Instead of communicating with all the involved agents in order to find out certain information about the community, an agent equipped with the acquaintance model consults this social knowledge stored in its wrapper instead. This feature is very closely linked with the second one. If we require immediate reply to an input or stimulation there is usually not much time for communication with collaborating agents. The agent must react without any delay and therefore it must have relevant information at hand, e.g. which agent should execute the task. Using negotiation in such cases is not acceptable. The agents can browse the Internet and search for relevant information. If such information is later used the source is included on the list of potential sources of information for further use.

Now let us briefly describe two applications of this model, namely diagnostics and planning.

CASE STUDIES

PLANNING

The ProPlanT (Mařík, 2000) system was developed for planning of the project-oriented production. The traditional production planning activity is substituted by agent driven service negotiations, intelligent decomposition and distributed decision-making. The system is a collection of agents, which reflect the information and organizational structure of the industrial enterprise and models the scheduling and planning process from product configuration phase to resource allocation. Agents may be divided into two fundamental super-classes: intra-enterprise agents (IAE) and inter-enterprise agents (IEE) (Říha, 2002). The following basic classes of agents in ProPlanT (see Figure 3) represent the IAE category: PRODUCTION PLANNING AGENT (PPA) is in charge of project planning. Its aim is to construct an exhaustive, partially ordered set of tasks that need to be carried out in order to accomplish the given project. It builds product configuration and contracts PMA agents. PRODUCTION MANAGEMENT AGENT (PMA) performs project management in terms of contracting the best possible PA agents (in terms of operational costs, the delivery time and current capacity availability). PMA delegates its responsibility either to another PMA or it controls work of a group of PA agents contracted for the considered task. PRODUCTION AGENT (PA) represents the lowest level production units that simulates or encapsulates shop floor production processes on the IAE. PA carries out the parallel-machinery scheduling of given tasks and manages resources allocation via special type of database agents. DATABASE AGENT (DBA) can be classified as both IAE and IEE. It maintains local database. This database can be used as agents knowledge backup or as representation of external resources availability. Another IEE agent is a CUSTOMER AGENT (CA). CA agent is the actor that may trigger the course of production planning. It negotiates with the PPA agent in order to specify the production requirement and both deadline and other production constraints. The role of the remaining IEE agents depends on specific business case. They correspond to customers and partners outside of the system and they can be represented e.g. by a special instance of PA or PMA.

The described system models a multi-level managing structure and performs intelligent reasoning about the enterprise resources with the aim to produce an estimate of project's deadline and costs as accurate as possible. The agent's knowledge structures providing information necessary for agent's efficient reasoning processes have been carefully studied and experimentally verified. The resulting transparent tri-base acquaintance (3ba) model (Mařík, 2002) is general enough to be applied for solving tasks in diverse context. It successfully covers the product specification and configuration phase, product flow configuration and resource allocation phases and partially the integration requirements of the production planning process. Moreover, it has been successfully used for coalition planning in humanitarian relief operations (Pěchouček, 2002) and it proved useful in the design of MAS system for medical diagnosis – see below.

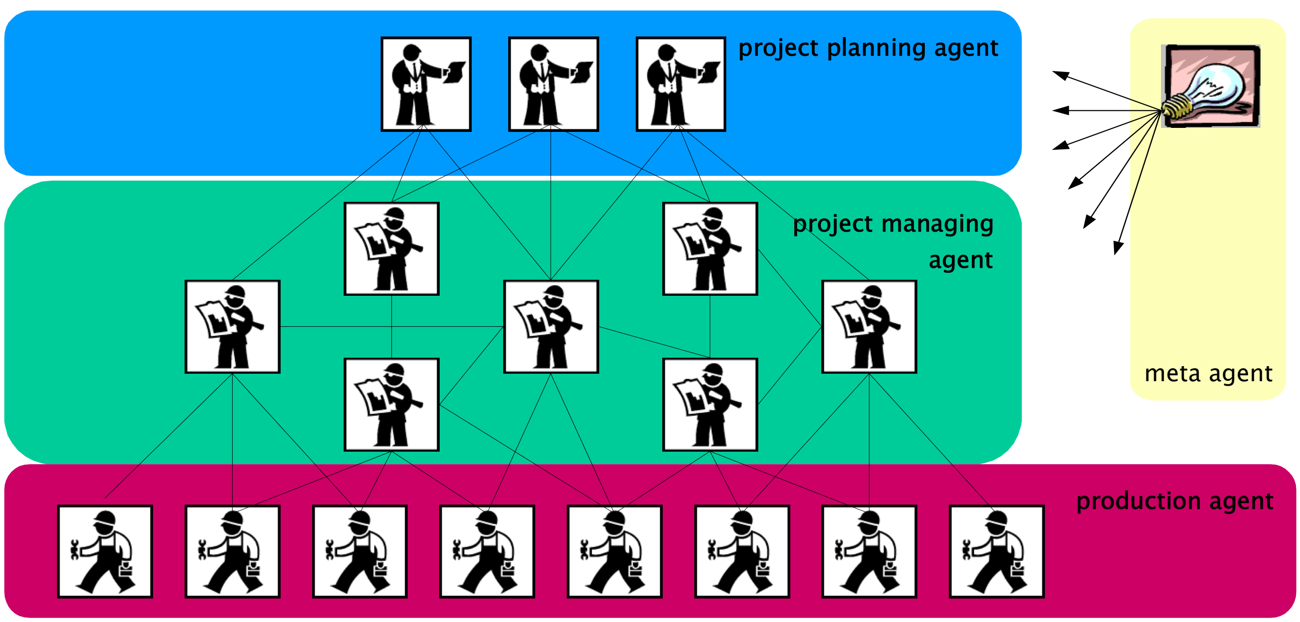

Figure 3 ProPlanT architecture

In ProPlanT, there is included a special monitoring agent - META AGENT (MA). It ensures analysis of behaviour of the agents' community as a whole and it offers advise how to improve system's efficiency (including material and work flow visualisation). Meta-agent can be viewed as one of the means of

MAS adaptation. One of its main advantages is that it does not undermine robustness of the whole system: the community of agents can survive even when meta-agent is switched-off or destroyed. "Ordinary" agents are able to communicate in peer-to-peer manner, but the meta-agent is able to induce a specific efficiency consideration from observation of the community workflow.

DIAGNOSTICS

Medical diagnostics in general is a complex process requiring vast amount of specialised knowledge and experience. Depending on the patient's symptoms, the general practitioner GP is able to determine a probable diagnosis more or less precisely. If the symptoms are the same for more diseases he/she has to perform further examinations (locally or send the patient to a specialised clinics). Some examinations are based on measurement of signals or parameters (e.g. blood pressure blood analysis). Most of these data need interpretation (= explanation of the semantic content).

Traditionally, vast quantities of measured data (EEG, ECG signals, etc.) have been interpreted by human experts with only minimal software assistance. However, such manual interpretation is a painstakingly slow and tedious process (imagine 24hour ECG record). Moreover, since interpretation involves subjective judgments and each interpreter has different scientific knowledge and experience, formulation of an effective interpretation often requires the co-operation of several such experts. Therefore it would be very useful to have a software system in which individual interpretations can be generated automatically and then refined through the use of cooperative reasoning and information sharing. Of course, it must be stressed that in any case the final decision is done by the medical doctor and not by a computer system. The computer system is always considered as a decision support tool.

Medical diagnosis was one of the application domains of expert systems in 1970s. Since that time many problem-oriented systems have been developed. However, most of them have had a narrow focus of expertise. They have taken the form of a single software methodology or technique (Shapiro, 1990), (Stefik, 1995), (Lhotská, 2001). First step to distributed organisation was introduction of blackboard architecture (Englemore, 1989). Logical continuation is the multiagent systems (Weiss, 1999) in which a number of interrelated tasks are performed by a network of cooperating agents. They may be heterogeneous utilizing different methods or techniques for their problem solving. Then their results can be combined or the best solution can be used.

When we look at medical data and information more closely we find out that there is no generally applicable best method or technique for evaluation of particular data. Each one has its relative strengths and weaknesses. Some can only produce an approximate solution, but do so comparatively quickly; others are more accurate, but relatively slow. Furthermore, a given technique's performance is often dependent on the nature of the data set (some work well with noisy data, others do not; some work well with data that has a high signal strength, others work comparatively better with a low signal strength, some can cope with missing data, others do not). However, the things are even more complicated. It is often impossible to determine a priori which technique is the most appropriate for a given data set or its part. There are several reasons for that, namely if the data set is too large the user is usually not able to evaluate the quality of data manually, the user may not be very experienced, the user may skip important part of data, etc. From that basic requirement on the system being developed follows: the system needs to be responsive to its problem-solving context.

To overcome the problems associated with selecting a single technique, there can be developed a system that allows multiple methods to co-exist. However, as examples from other domains show (e.g. image processing), such systems or tools typically place a significant burden on the user. For each technique, the user is expected to know its problem solving characteristics, be able to judge when, where, and how to apply it, and to determine how best to integrate and fuse the results it produces. It would be too demanding to solve this problem using a single monolithic system (e.g. expert system) because it does not allow integration of different techniques and evaluation of partial results reached by these techniques. Therefore we have decided to design and develop an open system that will provide a wide range of uncoupled base techniques (represented by separate modules) and allow the software system to determine at run-time which of them are appropriate in which circumstances. The interchange of partial and final results between individual modules will be directly supported at the software level.

Each data and signal pre-processing and evaluation technique can be regarded to be an autonomous software agent that cooperates, communicates and coordinates, if necessary, with other agents to try to satisfy the global goal. Components of the ADIA system (Agents for DIAgnostics) can be divided into three main layers, namely agents for data collection, agents for pre-processing and processing (determination of diagnosis), and agent for final evaluation. In addition to these agents, we propose a meta-agent that is an independent agent observing the community. It has two roles: passive role (visualisation of community structure, distributed solutions, user interface, etc.) and active role (it affects community operation - invokes operation sequences, learns from observations and tries to improve behavior of the whole community). The architecture is illustrated in figure 4. It is assumed that the agents are running on different machines and connected via Internet. The system architecture uses the same basic ideas as the ProPlanT system.

A very important question is how the agents should recognize which one is to start the pre-processing or directly processing (without pre-processing step) of accepted input data. As it has been mentioned above, it is advantageous to use different methods for different data types, depending on the nature of the data, presence of noise, etc. In some cases, e.g. processing of anamnestic data or numerical values from laboratory tests, there is usually not necessary to apply any pre-processing and processing step can be initiated.

Whenever data (measurements) enter the multi-agent system, their rough evaluation has to be done first with intention to distinguish life critical situations: multi-agent system has to work in completely different mode in such a case (Hernando, 2002).

As it has been already mentioned and following this analysis we have decided to introduce a meta-agent into the system architecture that would be able to learn from experience and thus to control co-operation of agents more efficiently.

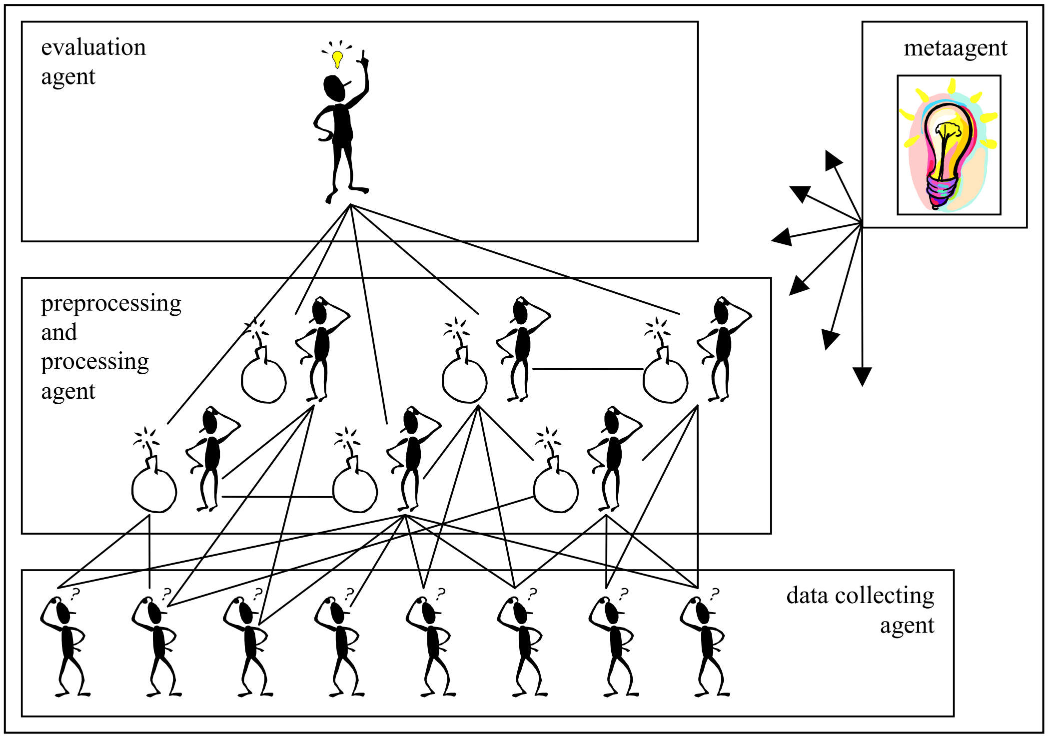

Figure 4. ADIA system architecture

Since there is certain knowledge about advantages and disadvantages of application of individual methods to various data processing it is possible to formulate a set of initial rules (prior knowledge) for such a meta-agent. If previously processed data type arrives and the meta-agent informs the processing agents about it, they will know which sequence of pre-processing methods and successive processing and evaluation to use (because such sequence was successful in the past case). If unknown (unrecognised) data type arrives, then all agents should be given the opportunity to try to solve the problem. However, in this case we get a number of different results. The question is how to decide which of them are relevant, and which are the best. The meta-agent serves as user interface as well. It enables visualization of the decision-making process and interaction with the user. If the suggested solution does not seem appropriate to the user, he/she can enter the decision making process and invoke certain agents manually.

Although the original 3bA model does not support proactive information search and retrieval, it is possible to equip the agents with this ability (adding relevant knowledge in the knowledge bases in agents' wrappers). This ability enables to locate and connect the ultimate service provider with the ultimate requesters in open environments. This is the case of connecting a human user with required processing agent. Since the user communicates with the agents using user interface agent we can equip this agent with necessary knowledge. Then it can ensure all functions that are realised, for example, by middle agents in the RETSINA system (Sycara, 2001). Let us describe one such scenario. The user has received ECG data from another source, thus he/she does not need any of the data collecting agents, but the services of pre-processing, processing and evaluating agents are required. The user interface agent starts communication with relevant agents and sends them data. When the whole processing and evaluation process is finished the user receives results. The user may ask, for example, for additional information about suggested diagnosis. The user interface agent starts communication with agents responsible for information search and retrieval that are able to find relevant information on the Internet and pass it to the user.

There are described in literature and WWW other multiagent systems and their applications in various domains - see for example (www.multiagent.com, www.agentlink.org). It is possible to identify several common features besides the basic agent characteristics, as heterogeneity of agents, integration of legacy systems, effort to add ability to learn to the agents, search for evaluation criteria, use of metaknowledge, etc.

CONCLUSIONS AND FUTURE WORK

In this paper we have illustrated strengths of the multi-agent approach on the examples of planning and diagnostics. Co-operating agents provide a very natural means of automating pre-processing and (at least partially) evaluation of vast amount of data utilizing all available knowledge. The multi-agent systems integrate very efficiently existing software systems by the agentification process. The agentified software system is encapsulated within the agent wrapper that administers agent-to-agent communication and collects the agents´ social knowledge. Such a software system becomes an agent – a fully-fledged member of the multi-agent system. That means that it is not necessary to replace the entire operational information or knowledge-based systems by a new technology. Instead, it is possible to make a best use of a combination of the existing software infrastructure and the novel, agent-based technology.

There are a number of issues that require further investigation. First, a more comprehensive set of pre-processing and processing techniques is required. Second, the agents should be able to adapt and learn from the social interactions they experience. Agents should learn which acquaintances give reliable results in which circumstances. Based on this knowledge they should be able to adapt their selection appropriately. The 3bA model represents general acquaintance model that allows construction of various global community functional architectures. As practical applications in different fields show (Marik, 2002), the 3bA architecture can be used by meta-agents accomplishing meta-level reasoning as well.

ACKNOWLEDGMENTS

The research described in the paper has been supported by the research program No. MSM 210000012 "Transdisciplinary Biomedical Engineering Research" of the CTU Prague, sponsored by the Ministry of Education, Youth and Sports of the Czech Republic and by MIRACLE - Machine Intelligent Research and Application Centre for Learning Excellence, EU project "Centre of Excellence", ICA1-CT-2000-70002.

REFERENCES

Agents, 2002: http://agents.felk.cvut.cz/main/index.php?home

Cao W., Bian C.-G., Hartvigsen G., 1996: Cooperator Base and Task Base for Agent Modeling: The Virtual Secretary Approach. In Proceedings of AAAI-96 Workshop on Agent Modeling, AAAI Press

Cao W., Bian C.-G., Hartvigsen G., 1997: Achieving Efficient Cooperation in a Multi-Agent System: The Twin-Base Modeling. In Kandzia P. et al., editors, Co-operative Information Agents, Heidelberg, Springer-Verlag LNAI 1202

Englemore R., Morgan T., editors, 1989: Blackboard Systems. Addison-Wesley, Reading, MA

Gomoluch J., Schroeder M., 2002: Flexible Load Balancing in Distributed Information Agent Systems. In Mařík V. et al., editors, Multi-Agent Systems and Applications II, Heidelberg, Springer-Verlag LNAI 2322

Hernando M.E., Gomez E.J., Alvarez P. and del Pozo F., 2002: Intelligent Alarms Integrated in a Multi-Agent Architecture for Diabetes Management. In Intelligent E-Health Applications in Medicine and Other Medical Applications of Adaptive/Hybrid Intelligent Systems. Aegean : University of the Aegean, 2002, vol. 1

Huang J., Jennings N.R., Fox J., 1995: An Agent-Based Approach to Health Care Management. Applied Artificial Intelligence 9(4)

Kléma J.; Lhotská L.; Štěpánková O.; Palouš J., 2000: Instance-Based Modelling in Medical Systems. In Cybernetics and Systems 2000. Vol. 2. Vienna: Austrian Society for Cybernetics Studies

Larsson J.E., Hayes-Roth B., 1998: Guardian: an intelligent autonomous agent for medical monitoring and diagnosis. IEEE Intelligent Systems, January-February 1998

Lhotská L., Mařík V., Vlček T., 2001: Medical applications of enhanced rule-based expert systems. International Journal of Medical Informatics 63

Mařík V., Pěchouček M., Štěpánková O., 2002: Organization of social knowledge in multi-agent systems. Integrated Computer-Aided Engineering 9

Mařík V., Pěchouček M., Štěpánková O., Lažanský J., 2000: ProPlanT: Multi-Agent System for Production Planning. Applied Artificial Intelligence 14(7)

Müller J.P., Bauer B., Berger M., 2002: Software Agents for Electronic Business: Opportunities and Challenges. In Mařík V. et al., editors, Multi-Agent Systems and Applications II, Heidelberg, Springer-Verlag LNAI 2322

Pěchouček M., Mařík V., Štěpánková O., 2000: Role of Acquaintance Models in Agent-Based Production Planning Systems. In Klusch M., Kerschberg L., editors, Co-operative Information Agents IV, Heidelberg, Springer-Verlag LNAI 1860

Říha A., Pěchouček M., Vokřínek J., Mařík V., 2002: From Intra-Enterprise towards Extra-Enterprise Production Planning. In Mařík V. et al., editors, Knowledge and Technology Integration in Production and Services, Kluwer Academic Publishers, Boston

Shapiro S.C., editor, 1990: Encyclopedia of Artificial Intelligence. John Willey & Sons Publ. New York

Stefik M., 1995: Introduction to Knowledge Systems. San Francisco. Morgan Kaufmann

Sycara K., 2001: Multiagent infrastructure, agent discovery, middle agents for web services and interoperation. In Luck M. et al., editors, ACAI 2001, Heidelberg, Springer-Verlag LNAI 2086

Weiss G., editor, 1999: Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. Cambridge, Mass. The MIT Press

Wooldridge M., Jennings N.R., 1995: Agent Theories, Architectures, and Languages: A Survey. In Wooldridge M., Jennings N.R., editors, Intelligent Agents. Heidelberg, Springer-Verlag LNAI 890