Asian Journal of Information Technology 3 (2): 107-114, 2004
© Grace Publications Network, 2004

A Practical Framework for Agent-based Hybrid Intelligent Systems

[1]Chunsheng Li, [1]Li Liu and [2]Qingfeng Song
[1]Faculty of Information Technology, University of Technology, Sydney P. O Box 123, Broadway, NSW 2007, Australia; [2]Integrated Development Department, Venus Info Tech Inc. Hai Dian District, Beijing 100081, China

Abstract: The design and development of hybrid intelligent systems (HIS) are difficult because there are many interactions between various components. Existing software development techniques cannot manage those complex interactions efficiently as those interactions may occur at unpredictable times, for unpredictable reasons, between unpredictable components. In this study, we contribute a multi-agent framework and a ring-based architectural model to organize those components and interactions. The framework consists of user interface, decision making, knowledge discovering, information management facilitators, and distributed heterogeneous data resources (DHDR). We employ middle agent concept to match task requesters with specific intelligent agents. A ring-based architectural model to organize the middle agents in HIS is developed. We demonstrate the potentials of the framework by case study and present theoretical and empirical evidence that our framework is available and robust.

Key words: Architectural model, coordination mechanism, hybrid intelligent system, intelligent agent, organizational structure

Introduction

Hybrid intelligent systems (HIS) combining genetic algorithms, fuzzy logic, neural networks and expert systems are proving their effectiveness in a wide variety of real-world problems. However, the design and development of HIS are difficult because they contain a large number of components with many interactions. Existing software development techniques cannot manage those complex interactions efficiently as those interactions may occur at unpredictable times, for unpredictable reasons, between unpredictable components. For this reason, it is futile to try and predict or analyze all the possibilities at design-time. At this stage, agent technology is very promising to take on the responsibilities. An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives (Khosla and Dillon, 1997). Agent-oriented techniques have the potential to improve current practice significantly in software engineering and to extend the range of applications that can be feasibly tackled. From the multi-agent perspective, entities in multi-agent systems are autonomous and engage in flexible, high-level interactions. Considering the autonomous nature of agents, autonomy means that the agents have their own persistent thread of control and that they decide which actions they should perform at what time for themselves. The fact that agents are active means that they know when they should be acting and when they should update their states for themselves. The flexible nature of interactions means that agents make decisions about the nature and scope of interactions at run-time rather than design time. Thus the multi-agent perspective is suitable for modeling, designing, and constructing HIS.

In an agent-based HIS, different types of intelligent agents are usually designed or employed for solving a specific complex problem. Those intelligent agents may be of the basic knowledge of fuzzy systems, neural networks, genetic algorithms, and other intelligent techniques. For improving the performance of agent-based HIS, two problems must be solved. The first one is how to hybridize and manage those intelligent agents so as to complete a task. The second one is how to organize those intelligent agents and necessary facilitators with suitable organizational structure and coordination mechanism. Organizational structure presents the interrelationship among agents, and coordination mechanism is knowledge level protocol to control the sequences of interactions and manage conflicts among agents.

In this study, we propose an overall framework for agent-based HIS. Based on the framework, we employ the middle agent concept to match sub-task requesters with specific intelligent agents. At the same time, we contribute a ring-based architectural model to organize the middle agents in HIS. Middle agent is a kind of facilitators. The ring-based architectural model is based on logical ring organizational structure and token-based coordination mechanism. About the organizational structure, the facilitators are divided into hosts and duplicates according to their roles in the system. About the coordination mechanism, elected coordinator mechanism is employed to build the logical ring, to manage token, and to proliferate or cancel facilitators based on sub-task agent type concept. For verifying the performance of the ring-based architectural model, an type-based HIS about description of crude oil and natural gas reservoirs is implemented. We develop a couple of facilitators with ring-based architectural model to match sub-task agent with two kinds of intelligent agents (neural network agent and genetic algorithm's pattern clustering agent). Here, facilitator acts on the role of matchmaker. The facilitator is of the features of proliferation and self-cancellation according to the sensory input from its environment. Two kinds of facilitators (host and duplicate) are implemented for promoting the scalability and robustness of the system.

Review of related work: Thus far there are some hybrid intelligent systems that adopted agent concepts. One of such attempts is the MIX multi-agent platform (Iglesias *et al.*, 1996). The focus of MIX is the development of strategies and tools for integrating neural and symbolic technologies. Another such attempt is the predictor system (Schererm and Schlageter, 1995). It used distributed artificial intelligence approach to combining neural networks and expert systems. In (Khosla and Dillon, 1997), Khosla and Dillon introduce a computational architecture called imahda (Intelligent Multi-Agent Hybrid Distributed Architecture). The imahda can be regarded as being constructed from generic software agents, generic intelligent agents, and problem solving agents. By analyzing these agent-based hybrid intelligent systems, we find out that the way for integrating intelligent techniques into multi-agent systems in those systems is to embed the intelligent techniques in each individual agent, and did not use any middle agents. Such integration approaches have two limitations. Firstly, it is impossible to embed many intelligent techniques within a single agent. Secondly, it is not flexible to add more intelligent techniques to or delete some unwanted ones from the multi-agent systems. Because of those limitations of the current paradigms, a novel approach needs to be developed.
In practical agent-based systems, the organizational structures have been modeled into Peer-to-Peer, Tree, and Grouping with Facilitator. Coordination mechanisms have been classified into direct search, matchmaker, broker, and contract-net. According to the research of Lee and Kim, seven architectural models for agent-based systems have been proposed by combination of the organizational structures and coordination mechanisms. Following this idea, many practical frameworks or architectures have been developed for solving some specified problems (Lertpalangsunti and Chan, 1998; Li, 2000 and Srikanth, 1999). However, the models are not enough to cover the gamut of work on agent-based system construction, especially, on the agent-based hybrid intelligent systems. All above work, to certain extent, promoted our research.

Multi-agent framework for HIS

Framework: For many complex problems, they involve many different components or sub-tasks, each of which is based on adequate information, rich knowledge, and appropriate skills by using relevant knowledge and information. In order to solve those complex problems, we propose an agent-based framework for HIS. The framework mainly consists of three parts, namely, decision making, knowledge discovering, and distributed heterogeneous data resources (DHDR) as shown in Fig. 1.
Decision making part uses available knowledge and appropriate information effectively to make reasonable decisions. Under this part, there are three main types of agents: decision making agents, intelligent agents, and serving agents. Here, decision making agents are agents without embedding intelligent techniques. They are at the front end of a multi-agent system (MAS). Intelligent technique agents are at the back end of the MAS. They provide decision making agents with intelligent technique capabilities. The serving agent is a matchmaker – one kind of middle agents. The ontology is the foundation for agent communication. All agents in the society interpret the content of received messages based on the ontology. The architectural model between agents in this part is one of the main tasks of this system because the agents are the main components of HIS.
Knowledge discovering part is of three aspects of function. The first aspect is information gathering, which gathers relevant information from Internet websites, databases, or data files according to other agents' requirements. The second aspect is data mining, which is in charge of discovering knowledge from retrieved information as well as other relevant databases or data files. The third aspect is information constructing, which is in charge of constructing necessary information and storing them into databases or data files. DHDR part consists of two sub-parts, namely, information management facilitators and data resources. Information management facilitator sub-part consists of three types of agents, each of which wraps one kind of the DHDR and makes the non-agent system exhibit agent characteristics in agent-based environment. Data resource sub-part consists of three types of information resources, that is, websites, databases, and data files.
A scenario goes as follows: the user queries or maintains the system through the user interface module of a

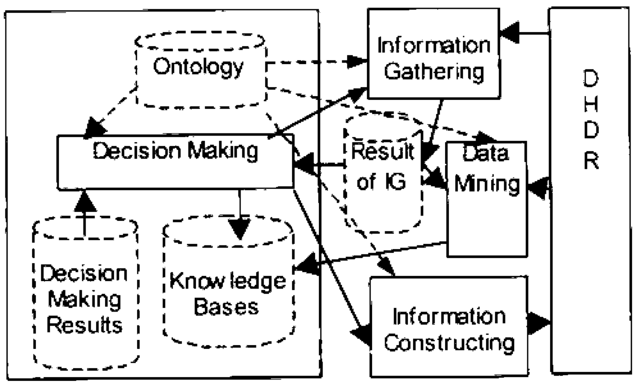

Fig. 1: Agent-based frame work for HIS

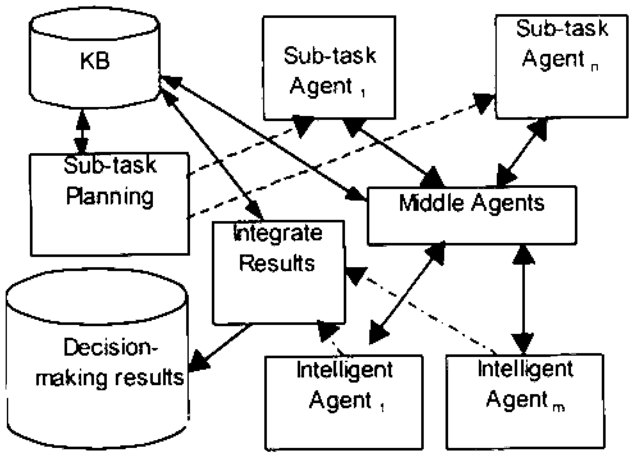

Fig. 2: Decision making society

user agent. The user interface module converts the user's inquiry to some kind of internal representation. The decision making agents then may ask information gathering agents to collect some information according to the user's inquiry on the Internet. The retrieved information is stored in a database, which may be located in DHDR. Data mining agents extract some knowledge from the retrieved information as well as other databases, and add the knowledge into the knowledge base of decision making agents. Data mining agents work on-line or off-line. The decision making agents make decisions according to the user's inquiry, the collected information, and their own domain knowledge. The final alternative decisions are provided to the user by the result visualization module of user agents. If it is necessary, some information may be organized and stored into databases or data files by means of information management facilitators.

Knowledge-based decision making society: How to incorporate intelligent techniques into decision making agents efficiently and effectively is of paramount importance. It is vital to seek novel framework to facilitate the construction of HIS for complex decision making. The framework must meet the following requirements. Firstly, any components with novel intelligent techniques or their combinations can be added to the system dynamically. Secondly, any decision making component in the system can access any one of the intelligent technique components which are available in the system. Thirdly, if one specific intelligent technique component disappears, decision making components can find other intelligent technique component with the same or similar capabilities. Finally, different decisions can be fused when necessary. A knowledge-based decision making society to accomplish the integration task is proposed. Fig. 2 shows the organization of the decision making society.

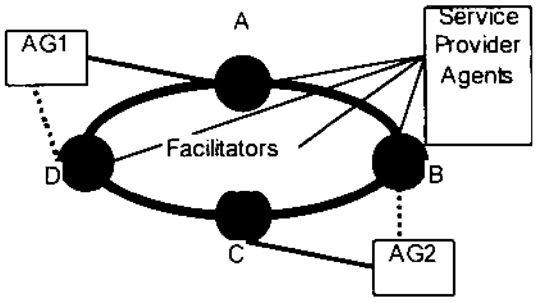

Fig. 3: Ring organizational structure

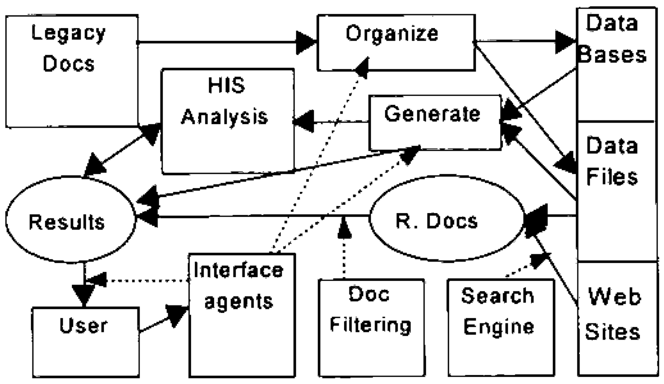

Fig. 4: Reservior description system

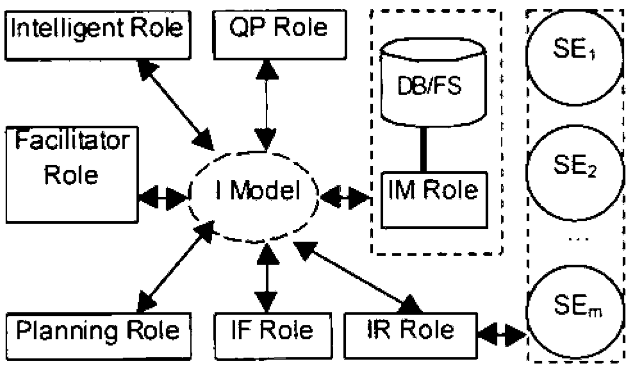

Fig. 5: Relationships between the roles

In this society, any soft computing technique adds to or deletes from the society dynamically as needed. This increases the flexibility and robustness of the corresponding hybrid systems significantly. Such a society has the ability to generate sub-tasks and match those sub-tasks with relevant intelligent agents. It is its responsibility that the result integration of each intelligent agent achieves the proposed task. The society integrates not only soft computing techniques but also traditional hard computing techniques such as expert systems.

When user submits a task to the society, the task will be proposed by using suitable knowledge-based planning approaches, where the knowledge is stored in the knowledge base (KB). According to the task, some sub-task agents (sub-task agent$_1$, sub-task agent$_2$, ... sub-task agent$_n$) will be proliferated for completing the user's task. The middle agents will match those sub-task agents with suitable intelligent agents (intelligent agent$_1$, intelligent agent$_2$, ... intelligent agent$_m$). When those intelligent agents finish the sub-tasks, their results will be integrated.

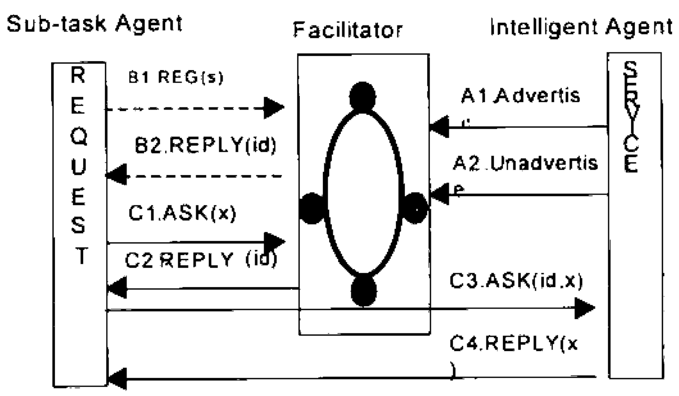

Fig. 6: Interactions between agents

The integrated decision making results will be sent to the user. In order to construct an agent-based hybrid intelligent system, each intelligent agent is not only developed, but the agents must be organized with an available architectural model which is convenient to employ those intelligent agents. The design of the architectural model is crucial for a hybrid intelligent system construction.

Architectural model for middle agents: A key issue concerning agent-based systems with middle agents is how to organize requester agents, middle agents, and service provider agents so that the requester agents can receive appropriate services quickly and efficiently. We contribute a self-organizing ring-based architectural model to organize the middle agents in agent-based system. Architectural model for middle agents is based on organizational structure and coordination mechanism. Ring-based architectural model is powerful for promoting the scalability and robustness of the key components (e.g., middle agents) of agent-based system. The ring-based architectural model is based on logical ring organizational structure and token-based coordination mechanism. The logical ring organizational structure is showed in Fig. 3.
The facilitators A, B, C, and D serve for requester agents, that is, agent group 1 (AG1) and agent group 2 (AG2). Each agent group includes a set of requester agents that are of similar feature or close cooperate to achieve a chief goal, which can be regarded as a type of sub-task agents using the same intelligent agent. The requester agents in an agent group may be organized as peer-to-peer or tree structure. A requester agent in an agent group cannot directly contact with one in another agent group. However, a requester agent can belong to more than one agent group. All information of service provider agents is stored in every facilitator by coordination mechanism. The service provider agent advertises its information to the coordinator of the ring. The information is transmitted to all facilitators by token ring coordination mechanism. Each facilitator stores meta necessary information (like yellow page) about its supervised child agents to control the interaction among those child agents.
The facilitators are organized as logical ring structure. That is, facilitator A can only contact directly with facilitator B and D, but C. If facilitator A wants to contact with C, the message must be transmitted by B or D. For simplifying control mechanism, the logical ring is designed to be of the feature of one way in transmitting message. For example, facilitator A can only directly transmit message to facilitator B and receive message from D, but it is not true in contrast. The facilitators are divided into two categories, namely, host and duplicate, according to their roles in the system. The host facilitators, for example A and C, are the superior agents of agent group 1 and agent group 2, respectively, in normal case. If the host facilitator A or C crashed, the ring would be automatically reorganized. At the same time, its duplicate facilitator D or B will act as a host and automatically proliferate a new duplicate facilitator. The meta information in host facilitator and its duplicate must be synchronized in time.

Coordination mechanism: Coordination mechanism is another factor related to the interaction and organization among agents and facilitators. Without coordination, agents will not work in order because of the conflicts between them and cannot correctly response to service requests. The token ring coordination mechanism is designed for solving above problem. The token ring coordination mechanism consists of two algorithms, namely,

logical ring establishing algorithm and token control algorithm.
For efficiently establishing the logical ring of facilitators, there is a coordinator among facilitators at any time to establish a logical ring (system beginning, token lost or the ring broken down). In general, it does not matter which facilitator takes on this special responsibility, but one of them has to do it. For selecting a facilitator as coordinator, each facilitator is assigned a unique number (0 to 255). In general, election algorithm attempts to locate the facilitator with the highest facilitator number and designate it as coordinator. Furthermore, it is assumed that every facilitator knows the facilitator number of every other facilitator. What the facilitators do not know is which ones are currently up and which ones are currently down. The goal of election algorithm is to ensure that when an election starts, it concludes with all facilitators agreeing on who the new coordinator is to be. In our research, we employed ring election algorithm (Tanenbaum, 1995). This algorithm is based on the use of a ring, but without a token. It is assumed that the facilitators are logically ordered, so that each facilitator knows whom its successor is. When any facilitator notices that the coordinator is not functioning (the token interval is overtime), it builds an election message containing its own facilitator number and sends the message to its successor. If the successor is down, the sender skips over the successor and goes to the next member along the ring, or the one after that, until a running facilitator is located. At each step, the sender adds its own facilitator number to the list in the message. Eventually, the message gets back to the facilitator that started it all. That facilitator recognizes this event when it receives an incoming message containing its own facilitator number. At that point, the message type is changed to coordinator and circulated once again, this time to inform everyone else who the coordinator is (the list member with the highest number) and who the members of the new ring are. When this messge has circulated once, it is removed and coordinator starts to manage the ring. The coordinator is in charge of managing the token, receiving service provider agent's advertisement, receiving new application-based system (agent group) request for proliferating a pair of new facilitators (host and its duplicate) for this new system, receiving a application-based system removing request for canceling the related facilitators (host and its duplicate) and removing them from the ring, maintaining he logical ring (control to proliferate new facilitators or remove old ones), etc.
When the ring is initialized, the coordinator will produce a token. The token circulates around the ring. It is passed from facilitator k to facilitator k+1 (modulo the ring size, 256) in point-to-point messages. When a facilitator acquires the token from its neighbor, it has three aspects of tasks to do. Firstly, facilitator checks the token if there is a service provider agent that has just advertised it to the coordinator. If yes, the facilitator registers service provider agent information. Secondly, if there is any maintaining message (new system information, cancelled system information, applied facilitator numbers, etc.) on the token, the facilitator updates related items or statuses. Thirdly, if the facilitator is a host, it checks if its duplicate is normal (If not, it proliferates a duplicate and makes it on the ring). If the matchmaker is a duplicate, it checks if its host is normal (If not, it acts as the host, proliferates a duplicate and makes it on the ring). When the coordinator holds the token, it will clear the token and reload it again according to the environment.

Case study

Framework of reservoir description system: An agent-based oil reservoir description system, which is used in petroleum industry, is developed by using the framework. The system consists of five aspects of function, namely, DHDR environment, non-agent-based legacy document processing, well-log information generating, petroleum information gathering, and HIS-based reservoir analysis. Fig. 4 presents the relationships between those modules and related agents.
DHDR consist of databases, data files, and websites. Legacy documents (e.g. well-log graphs) are processed and then stored into databases or data files. Those data can be regenerated according to user's requirements. User can query any petroleum information from data files or websites according to user's query statement. The rough relevant documents (R. Docs) can be got by employing specific search engines. User can get more exact relevant documents by means of document filtering agent (Doc filtering). By using hybrid intelligent approaches, reservoir can be described based on above results.

Role model of the agent-based system: We analyze a model with seven roles, namely, Intelligent role, Facilitator role, Planning role, Query Processing role (QP), Information Filtering role (IF), Information Retrieval role (IR), and Information Management role (IM). There are inevitably dependencies and relationships between the seven roles in the system. Indeed, such interplay is central to the way in which the system functions. The relationships between the roles are shown in Fig. 5.
Intelligent role, Facilitator role, and Planning role carry out the HIS-based reservoir analysis. Other agent roles

are designed to carry out the legacy document processing and regenerating, data management, and petroleum information retrieval. The IM role controls to create, to read, to update, or to delete information stored in database (DB) or file server (FS). The IR role informs selected search engines ($SE_1 - SE_m$) to browse the rough documents according to the browsing keywords or expressions, and saves those documents to file server by means of interaction with IM role. Intelligent role consists of two complicated lithology identification agents. One intelligent agent is based on Partheno-Genetic Algorithm's pattern clustering, and another one is based on artificial neural networks. Facilitator role consists of four middle agents that are organized with ring-based architectural model. Planning role consists of a sub-task planning agent and some dynamically produced sub-task agents. QP role consists of an information input agent and a data query agent. The information input agent, which scans, preprocesses, compresses, and digitizes images to data files or curve data, is developed with the image processing techniques. IF role includes an information filtering agent. The information filtering agent, which filters the relevant documents from the results of browsing agent and search engines, is developed with information retrieval techniques. IR role consists of a browsing agent. IM role consists of a data file management agent and a middleware agent. The data file management agent to store and access data in files is developed with the idea of distributed file system. The middleware agent, which is run in the same server with database management system, is developed with legacy software wrapping view to manipulate heterogeneous databases in uniform database-based agent communication language (DBACL).
The links between roles are presented in the interaction model (I Model). In fact, 'I Model' consists of a set of protocol definitions, one for each type of inter-role interaction. The protocols between agent and user consist of the interactive interfaces and disciplines for user to use the system.

Interaction between agents by facilitators: Facilitators automatically construct themselves to satisfy the environment of the intelligent agents and current types of sub-tasks (neural networks based sub-tasks or genetic algorithm based sub-tasks, which are produced by the planning agent). At the same time, they coordinate the interactions between sub-task agent and intelligent agent. The interactions can be divided into three categories: managing intelligent agent registration, managing current types of sub-task registration, and matching sub-task agent with intelligent agent. Fig. 6 presents the interaction patterns.
When an intelligent agent wants to register for providing service to sub-task agents, it advertises its capability and feature to the facilitator which is acting as the coordinator (A1 in Fig. 6). When the coordinator holds the token, it registers the intelligent agent information into Intelligent Agent Table, puts the information on the token, and informs all other facilitators to register the intelligent agent information as it did. When the token comes back, coordinator clears this information from token. When an intelligent agent doesn't want to provide service to sub-task agents, it sends 'unadvertise' to the coordinator (A2 in Fig. 6). When the coordinator holds the token, it removes the intelligent agent information from Intelligent Agent Table, puts the information on the token, and informs all other facilitators to remove the intelligent agent information as it did. When the token comes back, coordinator clears this information from the token. If a facilitator detects a crashed intelligent agent, it will ask coordinator to do the similar work as 'unadvertise'.
When a new type of sub-task wants to register for constructing its facilitators (host and duplicate), it sends its information to the coordinator (B1 in Fig. 6). When the coordinator holds the token, it registers the new type of sub-task's information into Current Type List, proliferates two facilitators for the type and makes them on the ring, puts the information on the token, and informs all other facilitators to register the type information as it did. When the token comes back, coordinator clears this information from token and replies the new type with related facilitator's access port number (B2 in Fig. 6). The type informs all its sub-task agents of the facilitator's port number. When a type doesn't want to run forever, it sends 'remove' to the coordinator. When the coordinator holds the token, it removes the related facilitators from the ring (if there are only two facilitators on the ring, this step will be overleaped), removes the type information from Current Type List, puts the information on the token, and informs all other facilitators to remove the type information as it did. When the token comes back, coordinator clears this information from the token.
For matching sub-task agent with intelligent agent, the steps are as follows.
1) Sub-task agent asks its related facilitator to answer to its request (C1 in Fig. 6).
2) Facilitator searches all intelligent agents in Intelligent Agent Table, selects an intelligent agent which is able to answer to the request, and replies the intelligent agent's information to the sub-task agent (C2 in Fig. 6).
3) Since sub-task agent knows which intelligent agent is able to solve its request, it directly asks the intelligent agent to answer the request (C3 in Fig. 6).

4) The intelligent agent completes the request and replies the result to the sub-task agent (C4 in Fig. 6).

After matching sub-task agent with intelligent agent, facilitator does not intervene any interaction between sub-task agent and intelligent agent, that is, Step 3 and 4 can be repeated for many times for other similar tasks. The proposed system was implemented by using C and Socket techniques. The well-log graph image file is preprocessed as the standard pattern file and then compressed to characteristic data files, which are saved to data file servers by data file management agent. The curves implied in those data files can be digitized to curve data. Transmit curve data to middleware agent by means of DBACL to store the data into Oracle database. The agents worked well by processing test with thousands of well-log graphs. We chose 118 stratum samples from the complicated block of an oil field in China for processing practically. According to geological characteristic, 8 measure-well parameters were chose to represent the pattern. By using HIS-based agents to distinguish the stratum lithology, the total accordance rate by comparing to practical geological data is 88.3 percent. Especially in dividing intergrade lithology, accordance rate reaches 86.9 percent.

Conclusion

Current software techniques for hybrid intelligent system development cannot meet the growing demand in applications, while agent technology can greatly increase our ability in modeling, design, and building hybrid intelligent systems. By combining them and exploiting the unique advantages of each, many complex problems such as reservoir description and financial investment planning can be solved in a shorter time frame and resulting in higher quality solutions. The main contributions in this study are as follows.

- The framework that overcomes the technical impediments and facilitates hybrid intelligent system construction is built.
- We develop an agent-based system for reservoir description of petroleum industry for verifying the framework.

References

Iglesias, C., J. Gonzales and J. Velasco, 1996. MIX: a general purpose multiagent architecture, in: M. Wooldridge, J. Muller, and M. Tambe (Eds.), Intelligent Agents II (ATAL'95), LNCS 1037, Springer, pp:251-266.

Khosla R. and T. Dillon, 1997. Engineering intelligent hybrid multi-agent systems, Kluwer Academic Publishers, USA.

Lertpalangsunti, N. and C.W. Chan, 1998. An architectural framework for the construction of hybrid intelligent forecasting systems: application for electricity demand prediction, Engineering Application of Artificial Intelligence, Elsevier Science Ltd. 11, pp: 549-565.

Li, S., 2000. The development of a hybrid intelligent system for developing marketing strategy. Decision Support Systems, Elsevier Sci., Ltd. 27, pp: 395-409.

Schererm A. and G. Schlageter, 1995. Aulti-agent approach for the integration of neural networks and expert systems, in: S. Goonatilake and S. Khebbal (Eds.), Intelligent Hybrid Systems, Wiley, pp: 153-173.

Srikanth, V. , 1999. Intelligent trading system: a multi-agent hybrid architecture, Proceedings of the IEEE/IAFE 1999 Conference on Computational Intelligence for Financial Engineering, pp: 64-73.

Tanenbaum, A. S., 1995. Distributed operating systems, Prentice-Hall, Inc. New Jersey, USA.