Information and Knowledge Exchange in a Multi-Agent System for Stock Trading

Yuan Luo
School of Computing, Staffordshire University, Stafford, ST18 0DG, UK. Y.Luo@staffs.ac.uk

Darryl N. Davis
Neural, Emergent and Agent Technologies Group, Department of Computer Science, University of Hull, HU6 7RX, UK. D.N.Davis@dec.hull.ac.uk

Kecheng Liu
School of Computing, Staffordshire University, Stafford, ST18 0DG, UK. K.Liu@staffs.ac.uk

Abstract: A distributed problem solving system can be characterised as a group of individual agents running and co-operating with other agents to solve a problem. As dynamic domains such as stock trading are continuing to grow in complexity, it becomes more difficult to control the behaviour of agents in the domains where unexpected events can occur. This paper presented an information and knowledge exchange framework to support the distributed problem solving in the stock trading domain. It addresses two important issues: (1) How individual agents should be interconnected so that their capacities are efficiently used and their goals are accomplished effectively and efficiently; (2) How the information and knowledge transfer should take place among agents to allow them to respond successfully to user requests and unexpected situations in the outside world. The focus of this paper is dynamic knowledge exchange among MASST agents. The co-ordinator agent together with a decision enabling warehouse acting as a dynamic blackboard plus direct intercommunication among the agents enable facts, commands, and rules to be transferred between MASST agents. Knowledge can be exchanged among the agents by using a combination of facts, rules and commands transfers.

Keywords: Distributed Problem Solving, Multi-Agent System, Knowledge Exchange, Stock Trading.

1. Introduction

All applications covered by Distributed Problem Solving (DPS) assume that it is possible to carry out a complex task by calling upon an assembly of specialists possessing complementary skills. When the problem is so wide and complex that one person cannot possess all skills to solve the problem, it is necessary to call upon several specialists, who must work together in pursuing a common objective. These specialists co-operate with one another to solve a common problem such as a medical diagnosis [Tu *et al.*, (1995)], the design of an industrial product [Iffenecker and Ferber, (1992)], the acquisition of knowledge, fault finding in nets [Jennings *et al.*, (1995)], the recognition of shapes [Demazeau *et al.*, (1994)], or the understanding of natural language [Sabah, (1990)].

A DPS system can be characterised as a group of individual agents running and co-operating with other agents to solve a problem. As dynamic domains such as stock trading are continuing to grow in complexity, it becomes more difficult to control the behaviour of agents in domains where unexpected events occur. In recent years, there has been considerable growth of interest in the design of intelligent agent architectures for dynamic and unpredictable domains. Most of today's intelligent agent architectures are limited to performing pre-programmed or human assisted tasks. In a multi-agent system that consists of several agents the agents should be able to interact with each other and with their environment in an adaptable manner. Each agent has a local view of the environment, generally has specific goals and alone is unable to solve the system devoted global task. The global characteristics of such a system thus emerges from the co-operation of its component parts. This co-operation, in turn, impinges on the interactions between agents and subtly modifies the properties of the system [Gleizes *et al.*, (2000)]. In order to be more useful in complex real world domains, agents need to be more flexible. They need to learn how to respond promptly to unexpected events while simultaneously carrying out their pre-programmed tasks in response to subtly modified triggers.

It is highly possible that an agent with a responsibility in a dynamic environment faces unexpected events. In order to be responsive, the agents should have enough knowledge to deal with unexpected events. If an agent is not able to deal with a particular event on its own, it can take the following actions: 1) Learn how to solve the problem by experimenting with different solution strategies. 2) Let some other knowledgeable agent solve the problem and then use the results. 3) Learn how to solve the particular problem by acquiring the necessary knowledge from other agents capable of solving the problem. 4) Ignore the unexpected event [Cengeloglu *et al.*, (1994)]. For a real time application domain such as the stock trading, action 1 may not be suitable because it may take a long time to obtain the solution. Action 2 is a natural way for co-operating agents to solve the problem. However, it is not suitable for scenarios where there are large volumes of data involved in either the event or result. For action 3, there may be no huge volume of network transferred data, however, the agents themselves must be quite sophisticated. They need temporarily (maybe permanently) to keep the knowledge acquired from other agents and then to learn how to use this knowledge to solve the particular problem.

For the domain of stock trading, we proposed a multi-agent framework for stock trading (MASST) [Liu *et al.*, (2000)]. MASST is a closely collaborating agent system in which every agent has its own specialised capabilities and knowledge, and no one agent has the whole knowledge about the world. All the task specific MASST agents are situated on the same machine. Hence we do not need to worry about huge volume data transfers over a network. Based on the discussion above, the MASST agents will take actions 2 and 4 listed above to deal with a particular event. The objective of this paper is to investigate and recommend a framework to support distributed problem solving for action 2.

In this paper, we address two important issues:

1) How individual agents should be interconnected so that their capacities are efficiently used and their goals are accomplished effectively and efficiently.
2) How the information and knowledge transfer should take place among agents to allow them to respond successfully to user's request and unexpected situations in outside world.

The breakdown of the rest of this paper is as follows: Section 2 gives some background about the applications of agents in stock trading domain. Section 3 briefly describes the MASST framework. Section 4 discusses in detail the information and knowledge exchange framework used in the MASST agents. Section 5 gives the conclusion for this paper.

2. Background

Intelligent agents are software that act on behalf of their human users in order to carry out arduous information gathering and processing tasks, such as locating and accessing information from various on-line information sources, resolving inconsistencies in the retrieved information, filtering away irrelevant or unwanted information, integrating information from heterogeneous information sources and adapting over time to their human user's information needs. Intelligent agents work by allowing people to delegate works that they could have done, to the agent software. Agents can automate repetitive tasks, remember things you forget, intelligently summarize complex data, learn from you, and even make recommendations to you.

The agent technology is especially suitable to address those issues concerning the portfolio management domain. Sycara et al. (1998) analysed the task of the portfolio management domain. They pointed that this is the task of providing an integrated financial picture for managing an investment portfolio over time, using the information resources already available over the Internet. This task environment has many interesting features, including: (1) the enormous amount of continually changing, and generally unorganised information available, (2) the variety of kinds of information that can and should be brought to bear on the task (market data, financial report data, technical models, analysts' reports, breaking news, etc.), (3) the many sources of uncertainty and dynamic change in the environment. The overall task in the portfolio management domain is to provide the best possible rate of return for a specified level of risk, or conversely, to achieve a specified rate of return with the lowest possible risks. A multi-agent system approach is natural for portfolio monitoring because the multiple control threads in such a computational model are a natural match for the distributed and ever-changing nature of the underlying sources of data and news that affect higher-level decision-making process. A multi-agent system can more easily manage the detection and response to important time-critical information that could appear suddenly at any of a large number of different information sources. A multi-agent system provides a natural mapping of the multiple types of expertise to be brought to bear during any portfolio management decision-making.

Rus and Subramanian (1997) presented a customisable architecture for software agents that capture and access information in large, heterogeneous, distributed electronic repositories. The key idea is to exploit underlying structure at various levels of granularity to build high-level indices with task-specific interpretations. Information agents construct such indices and are configured as a network of reusable modules called structure detectors and segmenters. They illustrate their architecture with the design and implementation of smart information filters in two contexts: retrieving stock market data from Internet newsgroups and retrieving technical reports from Internet FTP sites.

Benos and Tzafestas (1997) presented a methodology of studying the complex phenomena emerging in stock markets. Their methodology is based on the use of distributed multi-agent models with minimal knowledge representation and reasoning capabilities that have proven to be a powerful modelling tool for complex biological systems. Unlike neural models, they reported that their models allow a comparative and incremental evaluation of validity and relevance to the observed phenomena. The possibility of their application to the modelling and study of stock market phenomena was demonstrated on a simple example of a central agency that regulates the behaviour of the investors.

Delgado et al. (1999) investigated a hybrid learning system that combines different fuzzy modelling techniques. In order to implement the different methods, they proposed the use of intelligent agents, which collaborate by means of a multi-agent architecture. This approach, involving agents that embody the different problem solving methods, is a potentially useful strategy for enhancing the power of fuzzy modelling systems. Working with stock markets requires constant monitoring of the continuously changing stock information, and the ability to take decision instantaneously based on certain rules as the changes occur. Garcia et al. (2000) recently reported on a framework for implementing a deliberative multi-agent system for this domain. This system can be used as a proactive tool for expressing and putting to work high-level stock trading strategies. In the framework, agents are able to monitor and extract the stock market information via the World Wide Web and, using the domain knowledge provided in the form of defeasible rules, can reason in order to achieve the established goals. The overall system is integrated using Jinni (Java INference engine and Networked Interactor), which provides a platform fro building intelligent autonomous agents (Tarau, 1999), and Defeasible Logic Programming (DeLP), which provides the agents with the capability of reasoning using defeasible rules in a dynamic and changing domain.

Even though there are several agent-based approaches reported in literature that address the issues in the financial trading domain, most of the current agent-based approaches focus on how to get the information from the distributed resource (the Internet). The use of intelligent agents to support decisions has not been thoroughly explored and merits serious consideration. In current practice, portfolio management is carried out by investment houses that employ teams of specialists for finding, filtering and evaluating relevant information. Based on their evaluation and on predictions of the economic future, the specialists make suggestions about buying or selling various financial instruments. The current practice, as well as software engineering considerations, motivates our research in multiple agent systems for the stock management. A multiple agent system approach is natural for portfolio management because of the multiplicity of information sources and the different expertise that must be brought to bear to produce a good recommendation for a stock buy or sell decision.

3. MASST Framework

MASST is a middle layer agent system between the demand side of information (i.e. investors in stock market) and the supply side of information (i.e. the Internet). Figure 1 shows the MASST scope and its context. The major functions of MASST include:

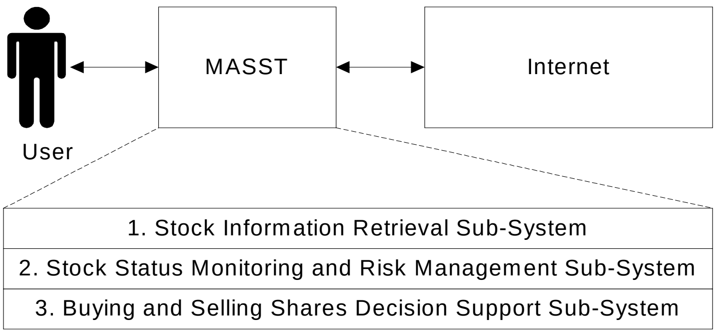

Figure 1. *MASST Scoped Organisational Context*

- Stock Information Retrieval – this function is not a unique function of our system. Nowadays many online brokers and commercial stock analysis software can provide this function. Thus our research does not focus on this field. Our system will provide this function because it is a basic requirement of investors and users. MASST will provide four categories of information retrieval: (a) trading history and current quotations, such as the information of Opening Price, Highest Price, Lowest Price, Current Price, Trading Volume for the selected stock on given trading date; (b) browsing stock technical analysis charts, such as price movement chart (for example Candlestick Chart), trading volume chart, and various technical indicators chart; (c) listed company's fundamental data and financial health information retrieval, such as total amount of stock trading volume, after-tax profits, earnings per share, annual earnings increases, number of common shares, new products or services, new management, and the market news; (d) market statistics information retrieval, such as the list of top ten shares of maximum trading volume in the given trading day(s), the list of top ten shares of maximum price upward (or downward), the list of top ten of the lowest Price-Earning Ratios.

- Stock Status Monitoring and Risk Management – MASST will automatically monitor the market status of the shares that the user holds and is interested. The share's market status includes the listed company's fundamental financial status and status of the share's technical indicators. Based on the share's market status, MASST will automatically and promptly report any abnormal status to users. Indicators of abnormal status include: (a) price fluctuation abnormal; (b) trading volume abnormal; (c) technical indicator's status abnormal, (d) price chart pattern abnormal, and (e) some break news relating to the given shares. Furthermore, MASST will provide the profits and risks management including calculation of profits/risk ratio based on shares' market status and user's investment, and reminders of stop-loss level for holding shares according to the user's profile.
- Buying and Selling Shares Decision Support – From the investors' perspective, the most important and concerned issues for investment in stock market are buying share issue and selling share issue. Which share is the best one to buy? What time is the right time to buy the share? What time is the right time to sell your holding shares? It is difficult (maybe impossible) to find a simple and accurate answer for these kinds of questions. Every investor has his/her own buying and selling share strategies and rules. MASST will provide buying and selling decision support based on behaviour rules defined by the investors themselves. Through a combination of human and machine knowledge, using agent and AI technologies, MASST aims to reduce investors' work overload in the process of stock analysis and investment decision-making.

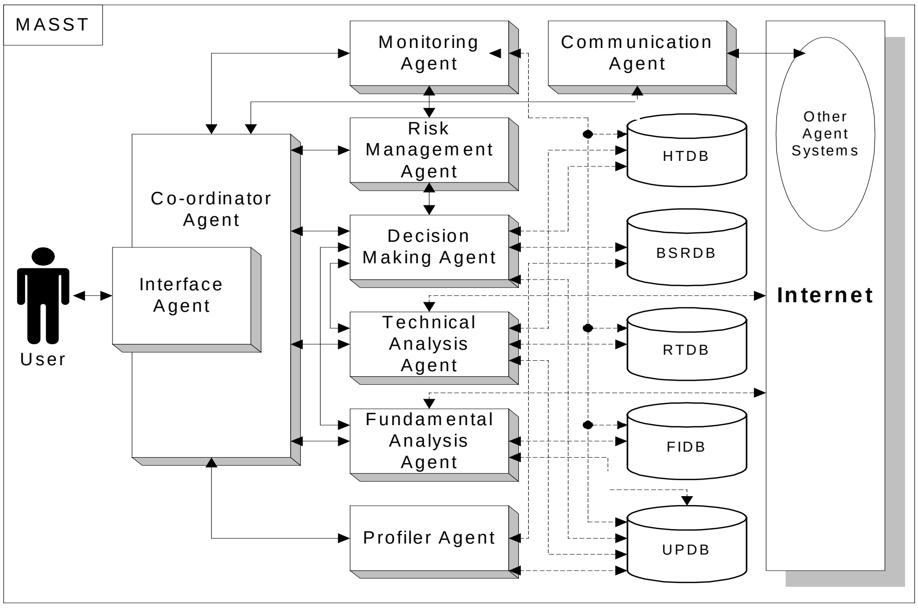

Figure 2. *MASST Framework*

We are implementing the MASST framework as shown in Figure 2. MASST provides a unified environment in which several agents are integrated. These intelligent agents inter-operate to collect, filter, and fuse information from distributed, network-based information sources and to make buying and selling decision suggestions for investors in their daily stock trading. The framework aims to provide a secure and private environment for registered users. There are three levels of privacy of the information held in a user profile:

- Public: the public information is available for all registered users.
- Restricted: the user can specify which individuals or groups are allowed to know about the restricted information. The user can exchange the information or opinions with specified users or groups in their common interest.
- Private: the private information is only available to the user's agents and will not be disclosed to other users' personal agents.

One of the key components in this framework is the User Profile Database (UPDB), which is dynamic, changing and shared amongst agents within the system. Each user has his or her own personalised interface agent, an individual user profile and Buying/Selling Rules Database (BSRDB) which is the user's private trading strategies, while other agents in the system are shared by all users. The UPDB includes information such as the username, password, the group that the user belongs to, the stock list that the user possesses, the stock list that the user is interesting in, monitoring instructions, planned tasks, preferences and privacy settings. These agents are assigned to an individual user and must be able to learn a user's interests and behaviour autonomously and adapt to the changing needs of the user over time. The profile is centrally available to all the user's agents.

The History Trading Database (HTBD), Real-time Trading Database (RTDB), and Fundamental Information Database (FIDB) combine to form a volatile local warehouse which is the internal data resource and is continuously updated with relevant external information by the Technical Analysis Agent and the Fundamental Analysis Agent. The functions and relationships among the agents in MASST are as follows:

- Interface agent interacts with the user, receiving user tasks and specifications and delivering results. Interface agent pass user's tasks to and get returns from co-ordinator agent.
- Co-ordinator agent is responsible for task decomposition and planning. The co-ordinator agent maintains a set of beliefs about the capabilities of all agents in MASST. It can decompose a given task into a number of subtasks and dispatch the subtasks to relevant agents to perform, in order to achieve its goals.
- Profiler agent provides the mechanism by which a user's profile and BSRDB are generated and maintained. The profiler agent interacts with the co-ordinator agent to receive information from the user and the environment to determine the interests of the user.
- Monitoring agent monitors the status of the given stocks on behalf of users according to the user's profile. This agent reports on the technical indicators' status of the given stocks and notifies any abnormal change in trading volume and price.
- Communication agent allows the framework to interact or communicate with other agents or programmes developed by other developers. This is a reserved interface to other systems.
- Risk management agent, on the basis of the user profile, interacts with the monitoring agent and decision-making agent to analyse the risk levels of user's share holdings, report the profit status and suggest a stop-loss level for the holding shares.
- Decision-making agent combines the outcomes of the technical analysis agent and the fundamental analysis agent, according to the investment strategies selected through the user's BSRDB. The decision agent will have two main functions: (1) to give a list of stocks advised for the next trading day to buy; (2) to give suggestions for users holding shares to hold or sell.
- Technical analysis agent retrieves and processes the raw stock trading data from the Internet, store the raw data to relevant database (HTDB, RTDB), calculates various technical indicators, identifies various price and trading volume patterns, and gives the output to decision agent.
- Fundamental analysis agent gathers the macroeconomics data, fundamental financial status of the listed companies, opinions of the market commentators or experts, and some relative news, and puts this information into FIDB. The fundamental analysis agent integrates the information and makes recommendations to the decision agent.

4. Information and Knowledge Exchange in the MASST

4.1 Overview of Inter-communication architecture in the MASST

Artificial Intelligence approaches to decision-making applications have been becoming more attractive to the commercial domain. An intelligent agent operating in a dynamic environment will find its reasoning and other actions constrained by limitation of time, information, and other critical resources. These agents have to interact with dynamically changing and partially unknown environments. Nonetheless, these agents must be able to respond to unanticipated changes and events occurring in their operational environment. In such situations, agents must know what actions need to be taken. If the agents are not able to deal with the unexpected situation, they should seek help from other agents in the system to resolve the problem as discussed in section 1.

Figure 3 shows the inter-communication architecture among MASST agents. The interface agent accepts tasks from and delivers information and solutions to the users. It passes the user's tasks or commands to co-ordinator agent. According the content of the command, co-ordinator agent distributes and decomposes the task to corresponding task specific agents such as monitoring agent, risk management agent, decision-making agent, profiler agent, technical analysis agent, and fundamental analysis agent. Co-ordinator agent must have access to knowledge that models the functions and task capabilities of the other agents. Information, relevant to the system's current task, is held by co-ordinator agent that exchanges it with all the other agents. Co-ordinator agent is also responsible for ensuring that the data used within the framework is reliable and available as integrated information held in the Decision Enabling Warehouse (DEW). From the perspectives offered by blackboard system design [Corkill, (1991); Carver and Lesser, (1994); Sadeh *et al.*, (1998)], the DEW together with the co-ordinator agent acts as a dynamic blackboard. The blackboard is a data structure that can be shared by all intelligent agents simultaneously, and co-ordinator agent is similar to the "Control" in traditional blackboard systems. The blackboard is organised by co-ordinator agent. The DEW has three different sections: (1) Blackboard Status - a counter incremented after each update of the conversation; (2) Blackboard Private Message Area, used by co-ordinator agent to send task related messages to other MASST agents and used by other MASST agents to control their conversation session; when a task or a conversation is complete, the relevant messages in this area are deleted; and (3) Blackboard Knowledge Area which holds rules, facts and raw data to be used by the MASST agents. The knowledge area is composed of HTDB, RTDB, BSRDB, FIDB, and UPDB.

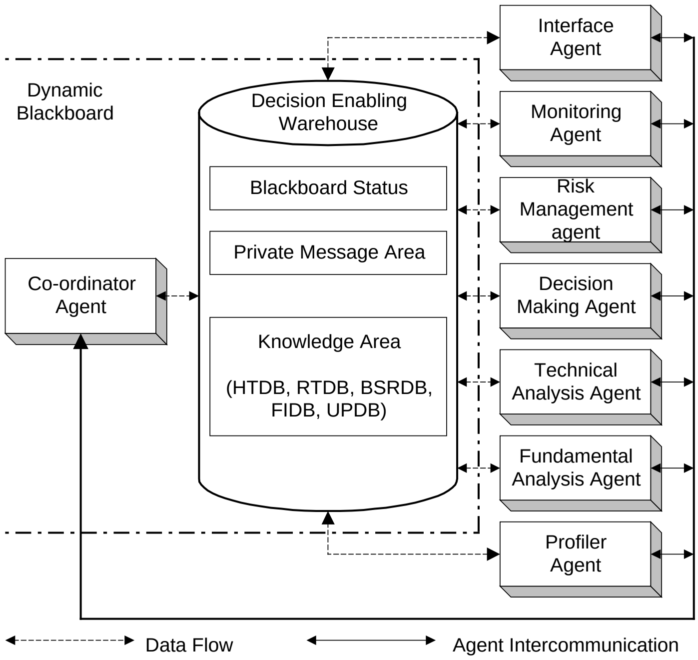

Figure 3. *Inter-communication Architecture among MASST Agents*

There are four communication categories in the MASST framework: (1) Internal Agent – Internal Database; (2) Internal Agent – Internal Agent; (3) Internal Agent – External Data; and (4) Internal Agent – External Agent. The communication protocol depends upon the type of agents involved and the information and knowledge being exchanged. The purpose of Internal Agent–External Data is to collect and filter the necessary information from distributed data resources (for example, the Internet). Much research has been done in this area [Maes, (1994); Sycara *et al.*, (1996); Etzioni, (1996); Chen and Sycara, (1998); Caglayan, (1997)] and is not repeated here. The focus of this work is on how to use the information for decision-making rather than how to collection it. The purpose of the Internal Agent-External Agent category is to make the MASST an open system, whereby it can extend its functions. The technical issues involved are ontology, agent communication language, security and so on.

Agent to database communication is ODBC based using SQL requests and commands. This protocol is relatively straightforward. Agent connects to an active database via ODBC, and phrases information requests using SQL. In MASST framework, the internal agents use this to obtain and place information on the decision-enabling warehouse. Internal agents in the framework communicate with each other using the MASST Agent Communication Language (MASST-ACL) which is the combination of FIPA ACL [FIPA 97 Specification, Part 2] and XML [XML v. 1.0, (1998)]. The details about MASST-ACL are beyond the scope of this paper. According to the description in Section 2, the major functions of MASST include: (1) stock information retrieval, (2) stock status monitoring and risk management, and (3) buying and selling shares decision support. In the following subsections, the agent interactions will be discussed in detail for these processes.

4.2 Agent Interactions in Stock Information Retrieval Process

MASST can provide seven kinds of stock information retrieval functions based on the Search Actions (SA) the user delegated. These include: SA1 – Query all share's quotation of current day; SA2 – Query a given share's quotation of current day; SA3 – Query a given share's real-time trading chart; SA4 – Query a given share's history price chart over a period; SA5 – Query a given share's price and technical indicator chart over a period; SA6 – Query a given share's fundamental analysis data; and

SA7 – Query the market statistic information over a period. Figure 4 shows how the agents interact and communicate in response to a request of SA1. User delegates the task to interface agent, which then passes the request to co-ordinator agent using MASST-ACL message by packaging the function ID (such as SA1) and the relevant parameters (such as Market-Code and Current-Date). Co-ordinator agent will distribute the task to relevant task specific agent based on the function ID. In this example, it is the technical analysis agent. Technical analysis agent then interacts with decision enabling warehouse through ODBC-SQL to obtain the database records. Then technical analysis agent sends a MASST-ACL message containing database records to co-ordinator agent. Co-ordinator agent then passes the message to interface agent, in the meantime, co-ordinator agent will decrease the Blackboard Status counter and delete the relative messages associated with this conversation session in the Private Message Area of the blackboard, as this conversation (or task) is complete. Finally, interface agent presents the results in a readily understandable format to the user.

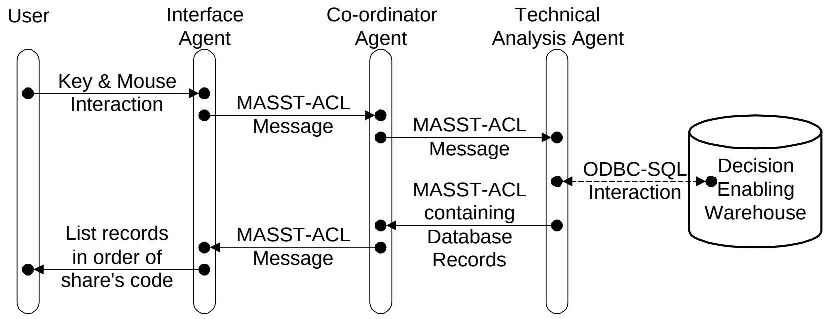

Figure 4. *Agent interaction through ODBC-SQL and MASST-ACL communication in response to the request of SA1*

For the other stock information retrieval functions (i.e. SA2 to SA7), similar agent interaction scenarios exist. Just the task specific agents and the information formats presented to the user by interface agent differ. For example, if the search condition is SA4 which is to access the given stock's price and trading volume charts, the interface agent needs to present the requested charts in a given time period.

4.3 Agent Interactions in Stock Status Monitoring and Risk Management Process

For the stock status monitoring and risk management process, the scenario of agent interactions is much more sophisticated than the one shown in Figure 4. MASST will automatically monitor the market status of the shares that are held or of interest to the user. The share's market status includes the listed company's fundamental financial status, status of price and trading volume movement, as well as status of the share's technical indicators. Based on the share's market status and the Monitoring Actions (MA) given by users, MASST will also automatically report any abnormal status in time to users. The Monitoring Actions include: MA1 – Monitoring price fluctuation abnormal; MA2 – Monitoring trading volume abnormal; MA3 – Monitoring technical indicator's status abnormal; MA4 – Monitoring price chart pattern abnormal, and MA5 – Monitoring the Break News relating to the given shares. Furthermore, MASST will provide profits and risks management, which includes calculation of profits/risk ratio based on shares' market status and user's investment, and reminders of the stop-loss level for user's holding shares according to the user's profile.

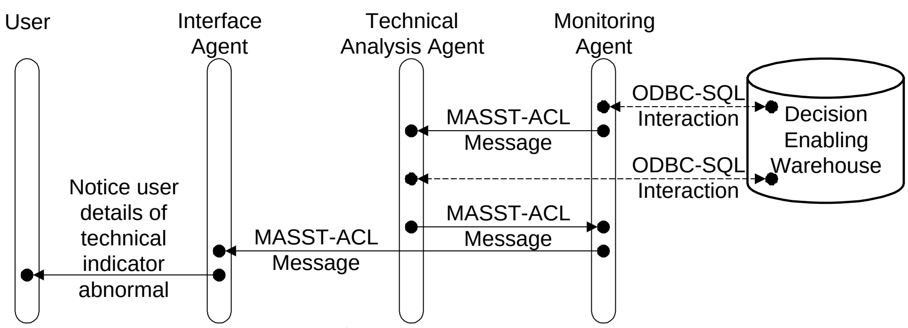

Fig 5. *Agent interactions for monitoring technical indicator abnormal*

As we discussed in the beginning of the section 3.1, the decision-enabling warehouse together with co-ordinator agent acts as a dynamic blackboard in the system. Every MASST agent can watch the information change (or an event) on the blackboard and an agent will react to the event or just ignore the event according to the agent's responsibility. Figure 5 shows the MASST agent interactions for monitoring of the market technical indicator abnormal. Monitoring agent requests the user profile database in decision-enabling warehouse to get the share codes needed for monitoring and user's monitoring instruction (i.e. MA4). According to the monitoring instruction, monitoring agent will ask technical analysis agent's help to find out the current value of the given shares' technical indicator by sending a MASST-ACL message with function ID (e.g. MA4) and parameters (e.g. share codes, technical indicators). Based on the function ID, technical analysis agent interacts with HTDB through ODBC-SQL and calculates the value of the technical indicators on behalf of monitoring agent, then reply the results to monitoring agent. And then monitoring agent compares the value obtained from technical analysis agent with the predefined appraisal thresholds of abnormal technical indicators that are obtained from the user profile database. If an abnormal event comes out, monitoring agent will immediately send a MASST-ACL message to interface agent, to notify the user of abnormal indicators.

4.4 Agent Interactions in Buying and Selling Share Decision Support Process

For the Buying and Selling Share Decision Support Process, the scenario of agent interaction is variable in the MASST. It largely depends on the buying and selling rules that defined by users themselves. For example, if the buying and selling rules are only one simple rule like:

IF short-term moving average price goes up through long-term moving average price **THEN** buying
IF short-term moving average price goes down through long-term moving average price **THEN** selling

The agents' interaction is quite simple. However, one cannot expect the system to give a good quality decision with such a simple rule. Normally it needs a set of rules which take into consideration share price, price pattern, price trend, trading volume, investment risk, technical indicators' status, the correlation between the individual share and market trend, the company's financial health and so on. MASST provides user an interface to set up the BSRDB in which holds the user's buying and selling rules (strategies). Users can change them at will by modifying the BSRBD.

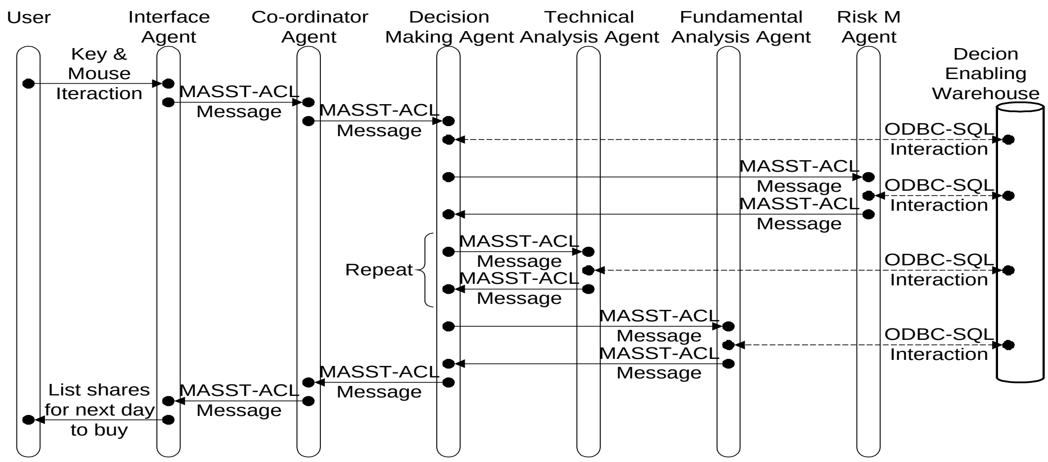

Fig 6. A*n example scenario of agent interactions for decision support of buying shares*

Figure 6 shows an example scenario of MASST agent interactions for this process. Suppose user delegates a task to the MASST like this: "tell me what is the best share(s) to buy for next trading day?" User delegates the task to interface agent through key and mouse interaction. Interface agent passes the task to co-ordinator agent with a function ID (for example, Buy-Decision-1) and parameters (for example, Market-Code and Current-Date) together. According to the function ID, co-ordinator agent decides to assign the task to decision-making agent. Decision-making agent needs to request the BSRBD to get the buying rules through ODBC-SQL. The records in BSRDB are not only the buying and selling strategies but also the commands to be executed by the MASST agents. Therefore, the decision-enabling warehouse together with co-ordinator agent, acting as a dynamic blackboard, plus direct communication between MASST agents make the dynamic exchange of facts, knowledge, and commands flexible and transparent in our framework. Suppose the buying rules in BSRDB lead the decision-making agent to walk along the following steps:

- Step 1: Decision-making agent (DMA) asks risk management agent to carry out risk evaluation for all shares. Risk management agent evaluates the trade-off between risk and return based on the portfolio theory [Markowitz, 1952)] or based just on the SAR algorithm [Wilder, (1978)]. DMA throws out the relatively high-risk shares in order to keep the profit/risk ratio under control.
- Step 2: DMA asks technical analysis agent (TAA) to carry out trade assessment. TAA measures the average number of trades that reveals the popularity of the share. DMA now rejects those

shares whose average number of trades is too small as shares of this kind are easily controlled by institutional investors.

- Step 3: DMA asks TAA to assess the correlation between the individual share and the market trend. TAA takes into account the relative stock trend as compared with the weighted stock price index. DMA rejects those shares whose price moves against the weighted stock price index in a given trading period as shares of this kind have probably been dominated by institutional investors.
- Step 4: DMA asks TAA to assess the share price trend and assumes the trend will continue. TAA evaluates the share price trend through the slope of moving average line and the relationship between short-term moving average and long-term moving average. DMA further throws out those shares whose price trend is in on a downtrend.
- Step 5: DMA asks TAA to assess the share's trading volume trend. TAA assess the share's trading volume trend by measuring the OBV indicator [Achelis, (1995)]. DMA further throws out those shares whose trading volume trend is in a downtrend while the price trend is in an up-trend.
- Step 6: DMA ask fundamental analysis agent (FAA) to evaluate the value of the share. FAA evaluates the value of the share by taking into consideration of the following factors: (1) Net Profit Margin; which indicates how much profit the company is able to make for each dollar of sales. (2) P/E Ratio (i.e., Price/Earning Ratio); the lower P/E ratio, the better value the share. (3) Book Value Per Share; if the share's current price is far below its book value, it may be an indication that the share is under-priced. FAA can obtain fundamental analysis data from the company's financial reports or statements that have already been collected in the decision-enabling warehouse. After evaluating the condition of the company, FAA can determine if the company's share is overvalued, undervalued, or correctly valued. According the suggestion, DMA further rejects the overvalued shares to get the final list of shares for next trading to buy.

DMA sends the final list of shares to co-ordinator agent, which passes the message to interface agent through MASST-ACL message, and finally the result is delivered to the user.

5. Discussion and Conclusion

In this paper, we proposed a communication architecture for the dynamic exchange of information and knowledge. The coordinator agent plays a vital role in maintaining the appropriate communication protocol. It decomposes system-level tasks to subtasks and distributes the subtasks to related task specific agents. The task specific agents are relatively simple. We recognise that a limitation of our MASST framework is the availability of the coordinator agent. The coordinator agent is the control locus of this framework. If it fails on its task, the whole system cannot work properly. Some generalised control heuristics will allow the system to recover, but remain unable to perform the precipitating task. An alternative is to change the design to that of distributed control. This can be done by making the task specific agents hold on the beliefs about the address and abilities of other agents, and giving the task specific agents the ability to decompose the system-level tasks to subtasks. This approach can improve the reliability of system but at the expense of increased complexity of design for each task specific agent. Future computational experiments will determine whether this is necessary.

The focus of this paper is dynamic knowledge exchange among MASST agents. We have introduced a framework in which MASST agents can exchange knowledge in a dynamic environment. The co-ordinator agent together with decision enabling warehouse acting as a dynamic blackboard plus direct intercommunication among the agents enable the transfer of facts, commands, and rules among MASST agents. Knowledge can be exchanged among the agents by using combination of facts, rules and commands transfers. We believe that dynamic knowledge exchange is an important feature for any application in which unanticipated conditions or events occur. Using the proposed dynamic knowledge exchange capability, co-operative problem solving sessions can be initiated where each agent can share its problem relevant knowledge with other agents to solve the problem. An obvious advantage of this capability is the elimination of redundant knowledge and hence the improved utilisation of the system memory capacity.

References

Achelis, S. B., (1995) Technical Analysis From A to Z, McGraw-Hill, 1995.

Benos, A. and Tzafestas, E. (1997), "Alternative distributed models for the comparative study of the stock market phenomena", *Information Sciences*, vol. 99, no. 3-4, pp. 137-157.

Caglayan, A., Snorrason, M., Mazzu, J., Jacoby, J., Jones, R. and Kumar, K. (1997) "Open sesame – a learning agent engine". *Applied Artificial Intelligence*, vol. 11(5) pp. 393–412, 1997.

Carver, N. and Lesser, V. (1994). "The Evolution of Blackboard Control Architectures", *Expert Systems with Applications, Special Issue on The Blackboard Paradigm and Its Applications, vol. 7, no. 1, 1-30.*

Cengeloglu, Y., Khajenoori, S. and Linton, D. (1994). "A Framework for Dynamic Knowledge Exchange among Intelligent Agents", *American Association for Artificial Intelligence (AAAI) Symposium; Control of the Physical World by Intelligent Agents,* New Orleans,LA, USA, November 1994.

Chen, L. and Sycara, K. (1998) "Webmate : A personal agent for browsing and searching". In *Proceedings of the Second International Conference on Autonomous Agents (Agents 98*), Minneapolis/St Paul, MN, May 1998.

Corkill, D. D. (1991). "Blackboard Systems", *AI Expert 6(9): 40-47*, September 1991.

Delgado, M., GomezSkarmeta, A.F., MarinBlazquez, J.G., and Barbera, H.M. (1999), "A Multi-Agent Architecture for Fuzzy Modelling", *International Journal of Intelligent Systems,* Vol. 14, No. 3, pp. 305-329.

Demazeau, Y., Boissier, O. and Koning, J. L. (1994) "Using Interaction Protocols to Control Vision Systems". In *IEEE International Conference on Systems, Man and Cybernetics,* San Antonio.

Etzioni, O. (1996) "Moving up the information food chain: Deploying softbots on the world-wide web". In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96*), Portland, OR, 1996.

FIPA 97 Specification, Part 2, Agent Communication Language, Foundation For Intelligent Physical Agents Fipa 97 Specification, Publication date: 10th October, Geneva, Switzerland, 1997.

Garcia, A., Gollapally, D., Tarau, P. and Simari, G. (2000), "Deliberative Stock Market Agents using Jinni and Defeasible Logic Programming", *Proc. Of the 14th European Conference on Artificial Intelligence, Workshop on Engineering Societies in the Agents' World (ESAW'00),* August, Berlin.

Gleizes, M. P., Glize, P. and Leger, A., (2000) "Abros: An Adaptive Brokerage Tool Based on Multi-Agent Framework", *IEEE Computer Communications,* January.

Iffenecker, C. and Ferber, J. (1992) "Using Multi-Agent Architecture for Designing Electromechanical Product". In *Proceedings of Avignon '92 Conference on Expert Systems and their Applications,* Avignon, EC2.

Jennings, N., Corera, J. M. and Laresgoiti, I. (1995) "Developing Industrial Multi-Agent Systems". In *First International Conference on Multi-Agent Systems,* San Francisco, V. Lesser (Ed.), MIT Press.

Liu, K., Luo, Y. and Davis, D. N. (2000). "A Multi-Agent System for Stock Trading", *Proceedings of Conference on Intelligent Information Processing, 16th world Computer Congress 2000,* August 21-25, 2000, Beijing, China.

Maes, P. (1994) "Agents that reduce work and Information overload", *Communications of the ACM,* 37, No. 7

Markowitz, H., "Portfolio selection", *Journal of Finance, March 1952.*

Rus, D. and Subramanian, D. (1997), "Customizing Information Capture and Access", *ACM Transactions on Information Systems,* vol.15, no.1, pp.67-101.

Sabah, G. (1990) "CARAMEL: A Computational Model of Natural Language Understanding using Parallel Implementation". In *Proceedings of the Ninth European Conference on Artificial Intelligence (ECAI 90),* Stockholm.

Sadeh, N. M., Hildum, D. W., Laliberty, T. J., McA'Nulty, J., Kjenstad, D. and Tseng, A. (1998). "A Blackboard Architecture for Integrating Process Planning and Production Scheduling", *Concurrent Engineering: Research and Applications, Vol. 6, No. 2,* June 1998.

Sycara, K., Decker, K., Pannu, A., Williamson, M. and Zeng, D. (1996) "Distributed Intelligent Agents". *IEEE Expert*, vol. 11(6), 1996.

Sycara, K., Decker, K. and Zeng, D. (1998), "Intelligent Agents in Portfolio Management", in Jennings, N.R. and Wooldridge, M. (Eds.), *Agent Technology: Foundations, Applications, and Markets,* Springer, pp.267-282.

Tarau, P. (1999), "Inference and Computation Mobility with Jinni", in Apt, K.R., Marek, V.W. and Truszczynski, M. (Eds.), *The Logic Programming Paradigm: a 25 Year Perspective,* pp. 33-48, Springer.

Tu, S. W., Eriksson, H., Gennari, J., Shahar, Y., and Musen, M. A. (1995) "Ontology-based configuration of problem-solving methods and generation of knowledge-acquisition tools: Application of PROTÉGÉ-II to protocol-based decision support", *Artificial Intelligence in Medicine 7(3), 257-289.*

Wilder, J. W. (1978). *New Concepts in Technical Trading Systems.* Greensboro, NC: Trend Research.

XML v. 1.0, (1998), eXtensible Markup Language (XML) 1.0, *XML v. 1.0 Model and Syntax, W3C Recommendation, February 10, 1998, REC-xml-19980210,* http://www.w3.org/TR/REC-xml