

# Combining KADS with ZEUS to Develop a Multi-Agent E-Commerce Application

Darryl N. Davis, Yuan Luo<sup>1</sup> and Kecheng Liu<sup>2</sup>

*Department of Computer Science, University of Hull, Hull, HU6 7RX, UK.*

[D.N.Davis@dcs.hull.ac.uk](mailto:D.N.Davis@dcs.hull.ac.uk)

<sup>1</sup> *School of Computing Science, Middlesex University, Bounds Green, London, N11 2NQ, UK.*

[y.luo@mdx.ac.uk](mailto:y.luo@mdx.ac.uk)

<sup>2</sup> *Department of Computer Science, The University of Reading, PO Box 225, Reading, RG6 6AY, UK*

[K.Liu@reading.ac.uk](mailto:K.Liu@reading.ac.uk)

A KADS based requirement analysis for the management of stock trading portfolios is presented. This provides a theoretical foundation for a stock trading system. This system is designed around portfolio management tasks that include eliciting user profiles, collecting information on the user's portfolio position, monitoring the environment on behalf of the user, and making decision suggestions to meet the user's investment goals. The requirement analysis defines a framework for a Multi-Agent System for Stock Trading (MASST). Experiments in task decomposition and agent interaction using a partially implemented system are described.

**Keywords:** Multi-agent system, stock trading, decision support, KADS, ZEUS

## 1. Introduction

Electronic commerce (e-commerce) will be the bedrock of the emerging information society. This combination of networked applications and commerce protocols needs to provide an information technology infrastructure that will support future business processes and the exchange of goods, resources and services.

Agent technology is one of the most vibrant and fastest growing areas of information technology – new agent based products, applications, and services are being announced on an almost daily basis. The reason for this intense interest is that the metaphor of autonomous problem solving entities cooperating and coordinating to achieve their desired objectives is an intuitive and natural way of conceptualising problem solving. Moreover, the conceptual apparatus of agent technology provides a powerful and useful set of computational structures and processes for designing and building complex software applications.

The development of the Internet, especially the emergence of electronic commerce, is making a dramatic impact on online stock trading. More and more people have an interest in the Stock Market and take part in online stock trading. This is due not only to the expected high profits in stock market, but also the development of the Internet, Web and other information technologies, which makes investment easier, faster, more effective and convenient. Taking into account the impact of the Internet and e-commerce on the Stock Market, what are the issues of concern to investors in the Stock Market? Generally speaking, any answer to this question should include the following two aspects: (1) how to retrieve investment-related information from a distributed resource (for example, the Internet); and (2) how to use the information in making investment decisions. Most agent based approaches in the financial trading domain, focus on how to fetch information from a distributed resource. The use of intelligent agents for decision support in stock trading has not been thoroughly explored and merits a serious consideration. This motivates our research in

investigating the multi-agent approach to decision support in stock trading. This paper presents a multiple agent (knowledge-based) framework for stock trading that can assist investors in investment decision-making based on requirements specified by the investor. Our research focuses on how to use the information for decision support rather than how to get the information from the Internet. This places a boundary on our research.

In the rest of this paper, we present a requirement analysis for the task domain of stock management. Based on a requirement analysis performed using the KADS methodology [1], we identify the overall organisational process that the system needs to support, and present the KADS process model that consists of process decomposition, process distribution, and process glossary. Arising from the process model is the proposal to use a framework comprising of multiple task-specific agents that together will be capable of performing decision support in the stock trading domain. We present an analysis of how the agents exchange information and knowledge, and the processes the agents must support. Based on the KADS analysis, we have partially implemented the framework proposed in this paper with the help of the ZEUS Agent Builder Toolkit [2] and Java servlet technology. The results of experiments with this framework are presented.

## 2. Background

The stock market is one of the most popular investment places because of its expected high profit. The Internet, Web, and other information technologies have brought and will continue to have a dramatic effect on the stock market. Various types of financial information and stock trading have become available to investors over the Internet. However, the information available from the Internet is disorganised and distributed over many sites. The variety of data sources is increasing dramatically and constantly changing. Therefore, information becomes increasingly difficult for an individual investor to collect, filter, evaluate, and use for decision-making. As a result, the problem of locating information sources, accessing, filtering, and integrating information in support of decision making has become a critical task.

In their analysis of the portfolio management domain, Sycara, Zeng and Decker [3] point out that the fundamental task is that of providing an integrated financial picture for the management of an investment portfolio over time using the information resources already available over the Internet. Central to this task is the provision of 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 risk.

Agent technology is especially suited to the issues that need to be addressed in designing computational systems for portfolio management. Rus and Subramanian [4] present a customisable architecture for software agents that capture and access information in large, heterogeneous, distributed electronic repositories. The key idea is to exploit the underlying domain 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 sites.

Benos and Tzafestas [5] present a methodology for studying the complex phenomena emerging in stock markets. Their methodology is based on the use of distributed multi-agent models with limited knowledge representation and reasoning capabilities. Such models have proven to be a powerful tool for modelling complex biological systems. Unlike neural net models, they report that these models allow a comparative and incremental evaluation of validity and relevance to observed phenomena. The feasibility of their application to the modelling and study of stock market phenomena was demonstrated with a simple example of a central agency that regulates the behaviour of investors.

Bui and Lee [6] propose a framework for building decision support systems using software agent technology to support organisations characterised as physically distributed, enterprise-wide, heterogeneous information systems. Intelligent agents offer tremendous potential in supporting well-defined tasks, for example information filtering, data mining and data conversion, in such an ill-defined environment. They propose the generation of a taxonomy of agent characteristics that can be used to help identify the type of agents needed to support different types of decision tasks. They advocate the use of a goal-directed, behaviour-based architecture for agent-based co-operative decision support systems.

Delgado *et al.* [7] investigated a hybrid learning system that combines different fuzzy modelling techniques. In order to implement the different methods, they proposed the use of an architecture built of multiple, collaborating, intelligent agents. This approach, involving agents that embody different problem solving methods, is a potentially useful strategy for enhancing the power of fuzzy modelling systems.

The use of intelligent agents to support decisions in the financial trading domain has not been so 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 stock management. A multiple agent system approach is natural for portfolio management because of the multiplicity of information sources and the different types of expertise that must be brought to bear to produce a good recommendation for a stock buy or sell decision.

### 3. KADS and ZEUS Methodologies

In the last few years, several attempts have been made to develop agent-oriented modelling techniques and methodologies. These include Agent-Oriented Analysis and Design [8]; the Agent Modelling Technique for Systems of BDI agents [9]; and the CoMoMAS methodology [10]. However, none of these techniques can yet be regarded as a comprehensive methodology for the analysis and design of multi-agent systems. In this section, the combination of the KADS analysis and design methodology with the ZEUS application toolkit employed in this research is presented. The successor to KADS (CommonKADS) is becoming the de facto European knowledge engineering standard. We do not claim this methodology provides a comprehensive framework for the analysis, design and development of all types of multi-agent systems, but it has sufficient power for analysing our problem domain and producing a framework suitable for implementation. A variation on CommonKADS, the MAS-CommonKADS methodology of Iglesias *et al.* [11], has been used to successfully model multi-agent systems.

KADS (Knowledge Acquisition and Document Structure) is a knowledge engineering methodology in which the main concern is with the two phases of analysis and design. The central theme is that of modelling the behaviour of the intended knowledge-based system, the context within which it will work and the framework for more general requirements and constraints. KADS is a structured approach to the development of knowledge based systems and as such is to be seen in contrast to unstructured approaches such as rapid prototyping. KADS does not require a commitment to any specific implementation paradigm. According to KADS, the development of a knowledge-based system is to be seen as a modelling process, during which models of acquired knowledge at different levels of abstraction are developed. The KADS user can determine if and how to model the results of a KADS analysis according to the requirements of the computational infrastructure within which the final system needs to sit.

KADS identifies three levels of models. At the process level, the process model identifies the tasks involved in the domain, the nature of data flows and stores, and the assignment of ownership of tasks and data stores to agents. At the system level, the co-operation model describes in detail the interactions between the system and external agents, and how the internal agents interact. The co-operation model is used to separate the user task model from the system task model, and allows the knowledge wholly internal to the system to be readily identified. The expertise level corresponds to an expertise model. This divides the task of describing domain and expert knowledge and its use within the system into a number of supportive tasks. The four-layer framework for expertise consists of the domain, inference, task and strategic layers. The domain layer is comprised of static or slowly changing knowledge describing concepts, relations, and structures in the domain. The inference layer reformulates the domain layer in terms of the different types of inferences that can be made. The task layer defines knowledge about how to apply the knowledge in these two layers to problem-solving activities in the domain. The strategic layer defines how to select appropriate problem-solving capabilities for specific types of task.

Within this framework knowledge engineering becomes a structured search for appropriate task, inference and knowledge models. This is performed through the use of analysis (the process of generating abstract descriptions of the inference patterns) and design and coding (a process of implementing the details of those patterns). The influence of KADS is most pervasive during the knowledge acquisition and analysis

stages of knowledge based system development. However, the overall KADS methodology provides considerably less guidance during design and implementation, except for recommending that these stages should strive to maintain the structure developed during the previous stages. Thus the ZEUS methodology [2] is employed for the design and implementation phase of this research system.

ZEUS is an integrated environment for the rapid building of collaborative agent applications. It was designed from the start to support distributed and Internet applications based on an agent-oriented programming paradigm. The main characteristic of ZEUS is the complete integration of all development stages from design to deployment. It provides theoretical and practical tools, uses software engineering techniques and provides an extensive library of methodological documents to facilitate software development.

The main activities associated with agent design in ZEUS include (1) a domain study leading to candidate agent identification and domain ontology specification, (2) agent definition, (3) task definition, (4) the specification of agent co-ordination protocols, (5) agent organisation, and (6) domain-specific problem solving code production. Steps 1-5 are followed iteratively until the developer is satisfied that the final suite of agents accurately captures the details of the problem being modelled.

The combination of KADS with ZEUS is a viable approach for the rapid analysis, design and implementation of a multi-agent based application. The KADS analysis clearly identifies the potential agents, the initial resources available to the agents and the tasks to be performed by the agents. It identifies the problem solving or 'expert' behaviour required of the system (i.e. how the agents are to perform their tasks), the relationship and interactions between the agents, and the domain ontology (known as Domain Structures in KADS). All of these results from the KADS analysis can be directly used as input to the ZEUS Agent Builder Toolkit. ZEUS can efficiently compensate the shortcomings of the KADS methodology during the design and implementation phases of system development.

## 4. Requirement Analysis for Stock Management Systems

The overall task in portfolio management, as stated by modern portfolio theory [12], is to provide the best possible rate of return for a specified level of risk. Sycara *et al.* [3] point out that portfolio management has several components. These include eliciting (or learning) user profile information, collecting information on the user's portfolio position, and suggesting and monitoring stock allocation to meet the user's current profile and investment goals.

The stock market is a complex system. Stock price movements are affected by many financial and human factors. Two common approaches to identifying future stock price movements are fundamental analysis and technical analysis. A fundamental analysis relies on the statistics of macroeconomics data to arrive at an estimate of future business conditions. This includes factors such as interest rates, money supply, inflationary rates and foreign exchange rates, as well as the basic financial status of firms and the daily news. Taking all these into account, the analyst buys in those stocks priced below his appraisal threshold. In contrast, a technical analyst pays more attention to financial time-series data. Predictions are made by exploiting implications hidden in past trading activities, and by analysing patterns and trends shown in price and volume charts. A technical analyst does not deal with what a firm sells or manufactures, or how it is capitalised. Both fundamental analysis and technical analysis can interpret stock price movements well. The former is usually adopted to predict the long-term stock trend, while the latter is better suited for short-term stock price movements.

Besides these analytical approaches, artificial intelligence is widely used to develop new methodologies for time-series predication. Advanced trading systems employ soft computing techniques such as neural networks, fuzzy logic, genetic algorithms, and expert systems. These have been widely used in finance for stock selection, stock forecasting, as well as profit and risk management with proven performance [13; 14; 15; 16].

Based on the literature and discussion above and by consulting with experienced traders, the basic requirements for a stock trading management system are the ability to perform the following tasks:

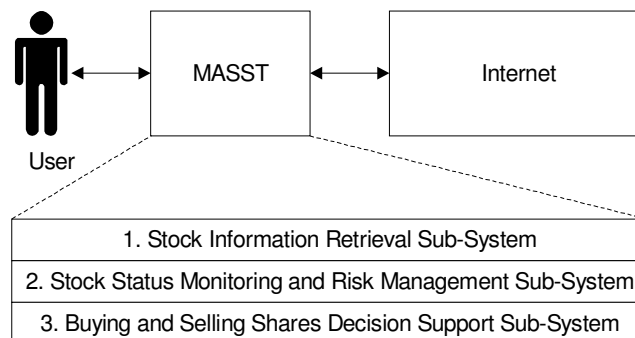
- Collecting raw stock trading data – the daily opening price, highest price, lowest price, closing price, trading volume, and number of trades.
- Providing technical indicators such as charts analysis, Japanese candlesticks philosophy, and Dow theory. Technical indicators are a group of mathematical equations with simple trading algorithms

as identified by [17]. Charts analysis is aimed at determining patterns and trends, for example reversal patterns hidden in price and volume charts [18]. Japanese candlesticks can reflect the mass psychology in a stock market [19]. Dow theory explains the financial market behaviour by means of primary, secondary and minor trends [20].

- Collecting and updating the fundamental information concerning the listed companies – such as stock trading volume, after-tax profits, earnings per share, the percentage increase in earnings per share, number of common shares, new products or services, and new management.
- Finding, filtering and evaluating relevant news, reports, and analyst’s comments from the environment (typically the Internet).
- Providing decision support for stock trading - combining technical analysis, fundamental analysis, and artificial intelligence technology, processing the input data, filtering out stocks of no interest, and advising on a list of stocks for buying, selling or holding.
- Identifying the investment behaviours of the institutional investors. Although the number of the institutional investors in the stock market is small, they have a marked influence on the state of the stock market. For individual investors, one of the most essential conditions for success is to understand the investment behaviours of institutional investors [21].
- Monitoring the status of given stocks on behalf of users, reporting the technical indicators’ status of the given stocks, notifying the user of any abnormal change in trading volume and price according to the user’s profile.
- Providing profits and risks management - calculating profits based on the user’s investment and reminding the user of the stop-loss for held shares according to the user’s profile.

## 5. MASST Scoped Organisational Context

This research, of which the project reported in this paper is but one part, proposes the use of a Multi-Agent System framework for Stock Trading (MASST). We have used such frameworks for decision support in other domains, for example the maintenance of water supply infrastructure [22], and for reasons given above consider them applicable to the current domain. This section outlines the organisation-level tasks of the MASST framework.



**Figure 1. MASST Scoped Organisational Context.**

MASST is a middle layer agent system situated between the demand side of information (i.e. investors in stock market) and the supply side of information (i.e. the Internet), as shown in figure 1. The major functions of MASST include:

- Stock Information Retrieval – this function is not unique to our system. Nowadays most online brokers and commercial stock analysis software can provide this function. While the research described in this paper does not focus on this, 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) Daily trading history and current quotations such as the opening price, highest price, lowest price, current price and trading volume for the selected stock on given trading date;

- (b) Stock technical analysis charts such as price movement chart (for example Candlestick Chart), trading volume chart, and various technical indicator charts;
  - (c) Listed company's fundamental data and financial health information, 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, such as the top ten maximum trading volume shares of the given trading day(s), the top ten shares displaying maximum price upward (or downward), the top ten list of stocks displaying the lowest price-earning ratios etc.
- Stock Status Monitoring and Risk Management – MASST will automatically monitor the market status of the shares that the user holds and/or are of interest. A 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 activity to the users. Indicators of such activities include: (a) abnormal price fluctuations; (b) abnormal trading volume; (c) technical indicator's status abnormal, (d) price chart pattern abnormal, and (e) some categories of breaking news related to the given shares. Furthermore, MASST will provide profit and risk management based on the users' profile including calculation of profits/risk ratio based on a shares' market status and user's investment, and reminders of stop-loss level for shares held.
  - Buying and Selling Shares Decision Support – From the investors' perspective, the most important issue for investment in the stock market is the buying and selling of 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 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. In this prototype we provided nine rules for buying shares or selling shares. Users can set their own trading strategy by any logic combinations of these nine rules. Through a combination of human and machine knowledge, using agent and artificial intelligence technologies, MASST aims to reduce investors' work load in performing stock analysis and investment decision-making.

## 6. Process Decomposition and Distribution

The previous section identified the system level functions or tasks of the MASST framework. This section reports on a more detailed analysis of the three major MASST functions in which the tasks and data stores associated with the decomposed process are assigned to different agents.

### 6.1 Stock information retrieval process

In MASST the stock information retrieval process (see figure 2) is split into seven sub-processes according to the Search Condition (SC) submitted by the user. SC1 - query quotations for all shares for the current day. SC2 - query a given share's quotation on a current day. SC3 - query a given share's real-time chart. SC4 - query a given share's history price chart over a given period. SC5 - query a given share's price and indicator chart over a given period. SC6 - query a given share's fundamental analysis data. SC7 - query the market statistic information over a given period. Figure 2 shows the augmented form of the stock information retrieval process, with tasks and data stores separately assigned to the Interface Agent, the Technical Analysis Agent, and the Fundamental Analysis Agent.

The History Trading Database (HTDB) holds the day-trading raw data for every share over a time period specified in the user profile. The data fields for every share include opening price, highest price, lowest price, closing price and trading volume. The Real-time Trading Database (RTDB) holds every share's real-time trading data over a trading-day. The data fields include opening price, highest price, lowest price, current price, current trading volume, accumulating trading volume, current bid price and volume, current ask price and volume. The Fundamental Information Database (FIDB) holds a listed company's fundamental data and financial health information, such as the 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, etc. The FIDB consists of text files that contain this information for the listed companies. There are several decades of Technical Indicators (TIs) which are used in stock technical analysis, such as MACD (Moving Average Convergence/Divergence), RSI (Relative Strength Index) and MFI (Money Flow Index). MASST can provide users with most of the technical indicators in common use.

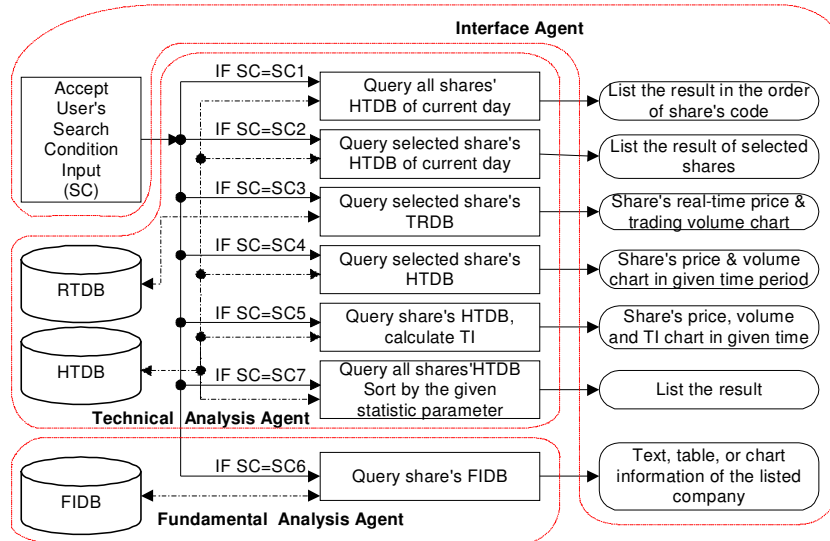


Figure 2. Stock information retrieval process decomposition.

## 6.2 Stock status monitoring and risk management process

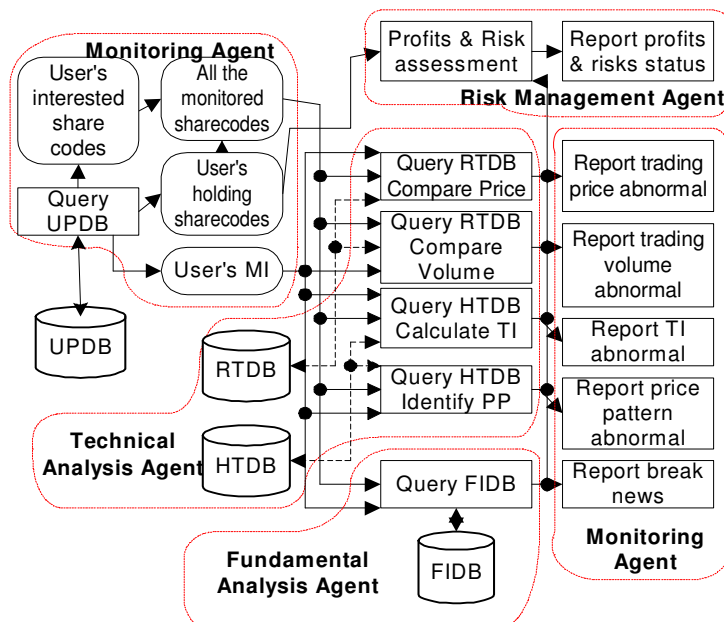


Figure 3. Stock status monitoring and risk management process decomposition.

The stock status monitoring and risk management process is split into six sub-processes according to the Monitoring Instructions (MI) delegated to MASST by the user. (1) Monitoring price abnormality for given shares. (2) Monitoring trading volume abnormality for given shares. (3) Monitoring technical indicator abnormality for given shares. (4) Monitoring price pattern abnormality for given shares. (5) Monitoring the breaking news related to given shares. (6) Reporting profit and risk status for given shares. Figure 3 shows the decomposition of the stock status monitoring and risk management process with tasks and data stores assigned to agents. Both user held share codes and user interested (but not held) share codes are stored in the User Profile Database (UPDB). The Monitoring Agent retrieves all the monitored share codes by sending a request to the UPDB every trading day. MASST provides the five kinds of abnormal status' monitoring functions described above, but the user must define what abnormal status is for each of these processes. For example, if the user wants MASST to monitor the price abnormal status for a particular share, the user should define what price level is abnormal (the appraisal threshold) for that share. The user can define monitoring tasks by giving the Monitoring Instructions that are any combinations of these five functions and related abnormal status definitions. The user's Monitoring Instructions are also stored in the UPDB. The Monitoring Agent will interact with the Technical Analysis Agent and the Fundamental Analysis Agent in order to complete the monitoring tasks assigned by the user. Any reports of abnormal status will be presented to the user through the Interface Agent. The Risk Management Agent will continuously assess the profits status and risk level of user's holding shares by interacting with the Monitoring Agent.

### 6.3 Buying and selling shares decision support process

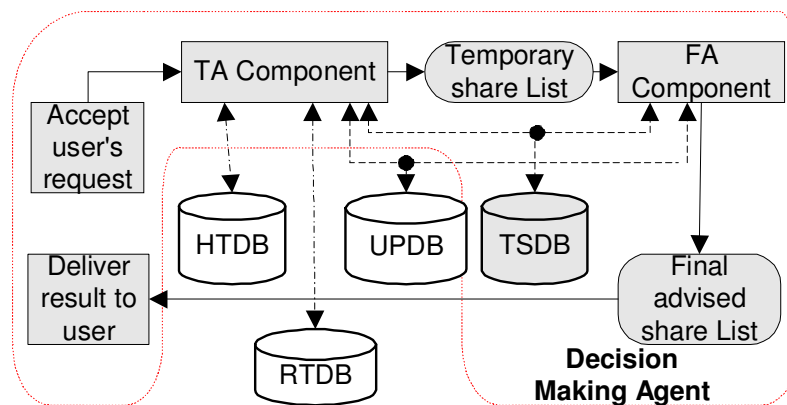


Figure 4. Buying and selling share decision support process decomposition.

The main functions in the buying and selling shares decision support process are (1) to evaluate the investment value of given shares; (2) to suggest the buying of given shares; and (3) to suggest the selling of given shares. Figure 4 shows the process decomposition for this function. This process is composed of an intelligent stock selection sub-process and an intelligent stock assessment decision support sub-process. The stock market is a complex system. Stock price movements are affected by many financial and human factors. Two common analytical approaches are fundamental analysis and technical analysis. A fundamental analysis relies on the statistics associated with macroeconomics data to arrive at an estimate of future business conditions. This includes factors such as interest rates, money supply, inflationary rates and foreign exchange rates, as well as the basic financial status of firms and the daily news. Taking all these into account, the analyst buys in those stocks priced below his appraisal threshold. In contrast, a technical analyst pays more attention to historical financial time-series data. Predictions are made by exploiting implications hidden in past trading activities, and by analysing patterns and trends shown in price and volume charts. Both fundamental analysis and technical analysis can interpret stock price movements well. The former is usually adopted to predict the long-term stock trend, and the latter is better suited for the short-term stock price movements.

There are two main components to this process. A Technical Analysis (TA) component is responsible for pre-processing the market raw trading data, filtering out shares of little or no interest and advising on a list

of shares for buying or selling. The second component is the Fundamental Analysis (FA) component, which plays the role of an experienced trader to refine the advised share list produced by the TA component. These activities can be considered to be sequential. In the first stage, the TA component pre-processes the raw trading data of each given stock and produces a temporary share list with the highest potential. In the second stage, the FA component makes a judgement of the market based on financial data and combines its recommendation with the output of the TA component to provide a list of shares for trading.

To support this process, the Decision Making Agent will need to interact with Technical Analysis Agent and Fundamental Analysis Agent in order to supply decision support to the users. The activities in this process are closely related to the user's trading strategies that are stored in the Trading Strategies Database (TSDB). Because of the complexity of the stock market itself and the complexity of the investors' investment behaviours, there is no unique, indubitable trading strategy guaranteed to win in the stock market. The investors should have their own trading strategies for buying and selling shares. The records in TSDB are the user's instructions, which provide guidance to the system in its decision support activities.

## 7. MASST Framework

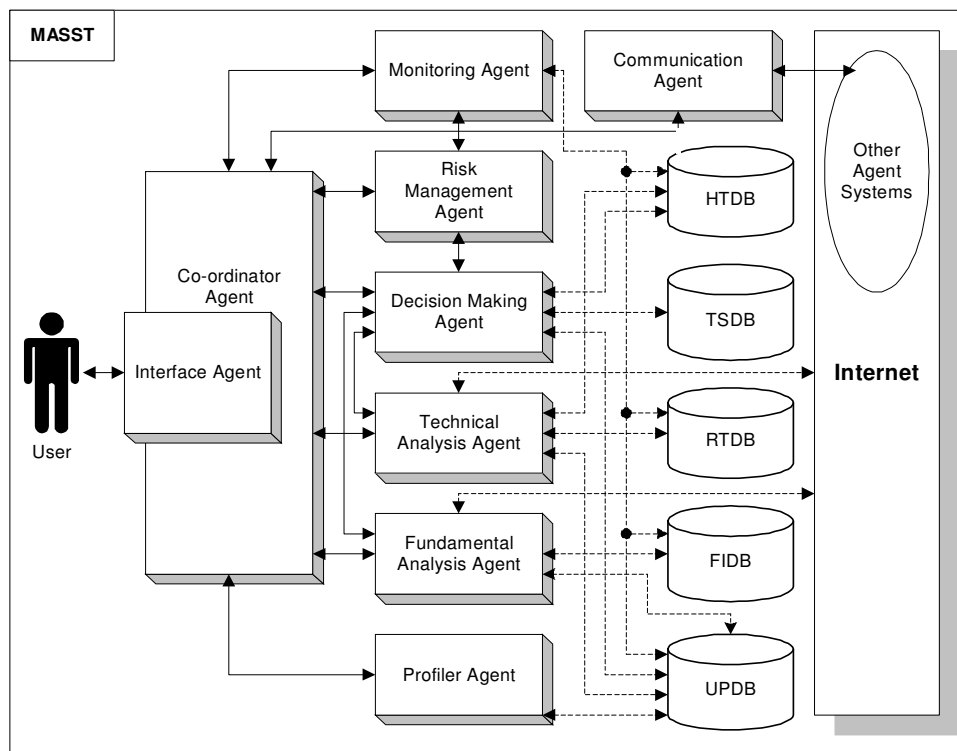


Figure 5. The MASST Framework

Based on the process analysis reported above, the MASST framework is being implemented as shown in figure 5. MASST provides a unified environment in which several agents are integrated. Further details are available elsewhere [23]. These intelligent agents co-operate to collect, filter, and fuse information from distributed, network-based information sources and to make buying and selling 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:** this information is available to all registered users.

- Restricted: the user can specify which individuals or groups are allowed to know about specific information. The user can exchange information or opinions with specified users or groups who share a common interest.
- Private: this information is only available to the user's agents and will not be disclosed to other users or their personal agents.

One of the key components in this framework is the User Profile Database (UPDB). This is modelled as a dynamic data store shared amongst agents within the system. Each user has his or her own personalised interface agent, an individual user profile and Trading Strategy Database (TSDB) defining the user's private trading strategies. Other databases in the system are shared by all users, all of whom are served by all of the remaining agents. 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 interested 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 History Trading Database (HTDB), Real-time Trading Database (RTDB), and Fundamental Information Database (FIDB) combine to form a volatile local data warehouse that acts as the main internal data resource. This is continuously updated with relevant external information supplied by the Technical Analysis Agent and the Fundamental Analysis Agent. The functions and relationships among the nine main agents in MASST are as follows:

- The Interface Agent interacts with the user, receiving user tasks and specifications, and delivering results. This agent passes user's tasks to and receives solutions from the Co-ordinator Agent.
- The Co-ordinator Agent is responsible for task decomposition and planning. This 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.
- The Profiler Agent provides the mechanism by which a user's profile and TSDB are generated and maintained. This agent interacts with the Co-ordinator Agent to receive information from the user and the environment to further the interests of the user.
- The 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 indicator status of given stocks and notifies the user of any abnormal change in trading volume and price.
- The 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.
- The 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 the user's share holdings, report the profit status and suggest a stop-loss level for the holding shares.
- The 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 TSDB. The Decision Making Agent has two main functions: (1) to give a list of stocks advised for buying on the next trading day; (2) to give suggestions for users holding shares to hold or sell.
- The Technical Analysis Agent retrieves and processes the raw stock trading data from the Internet, stores the raw data in the relevant database (HTDB or RTDB), calculates various technical indicators, identifies various price and trading volume patterns, and provides results for the Decision Making Agent to act on.
- The Fundamental Analysis Agent gathers macroeconomics data, fundamental financial status of the listed companies, opinions of the market commentators or experts, and relevant news. This information is placed in the FIDB. The Fundamental Analysis Agent integrates this information and makes recommendations to the Decision Making Agent.

## 8. Implementation

Although a full KADS analysis has been performed, this research work is still ongoing. At the time of writing, the framework has been implemented as an experimental platform (figures 6 to 9). Further work is

currently addressing the detailed nature of the user profile and how that relates to the type of data mining the system performs across the Internet to provide summary information supportive to decision making in MASST.

Users delegate their tasks through a web browser running on the MASST home page. Java servlets, running on the WWW server, accept users' requests and pass them to the MASST. These requests define the tasks the MASST agents will complete on behalf of the users. Agent resources (for example, the raw trading data) are kept in a Microsoft Access database. MASST agents interact with this database through JDBC-ODBC connections. Java servlets produce HTML pages as MASST responses to the client's requests. These experimental MASST agents are constructed by using the ZEUS Agent Builder Toolkit [2].

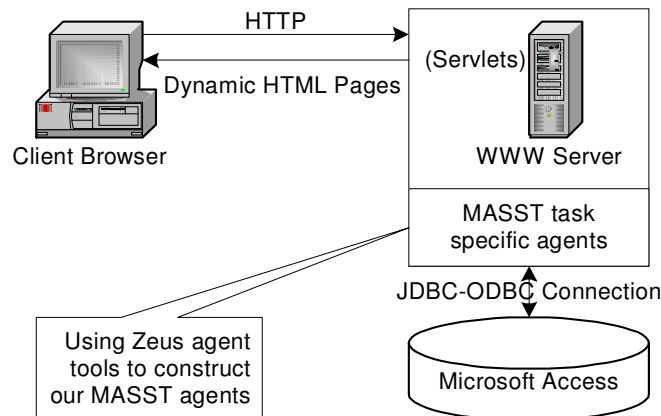


Figure 6. One experimental platform based on aspects of MASST.

In the MASST framework, the Interface Agent accepts tasks from and delivers information and solutions to the users. The Interface Agent passes the user's tasks or commands to the Co-ordinator Agent. According to the content of the command, the Co-ordinator Agent distributes and decomposes the task to corresponding task specific agents such as the Monitoring Agent, the Decision Making Agent, or the Technical Analysis Agent. The Co-ordinator Agent must have knowledge that models the functions and task capabilities of the other agents. This is one potential bottleneck in the current design. Other design options require greater framework sophistication.

Information, relevant to the system's current task, is held by the Co-ordinator Agent. This information is exchanged with other agents in defining their tasks. The Co-ordinator Agent is also responsible for ensuring that the data used within the framework is reliable and available as the integrated information held in the Decision-Enabling Warehouse (DEW). The DEW together with the Co-ordinator Agent acts as a dynamic blackboard. The blackboard is a data structure that can be simultaneously shared by all the agents within the systems. The Co-ordinator Agent is similar to the "Control" module in traditional blackboard systems, and is responsible for organising the blackboard. The DEW has three different sections. (1) Blackboard Status is a counter that is incremented after each update of the conversation. (2) Blackboard Private Message Area is used by the Co-ordinator Agent to send task related messages to other MASST agents and by other MASST agents to control their conversation sessions. When a task or a conversation is completed, the relevant messages in this area are deleted. (3) Blackboard Knowledge Area holds the rules, facts and raw data used by the MASST agents.

In this implementation, the Blackboard Status is a database table called TaskList with records: TaskID, Parameters, Status, and DMDate. TaskID and Parameters are based on our ontology definition, a direct result of the KADS analysis. Status is the representation of the current status of the current task, and DMDate is the decision-making date. TaskID is used to denote the tasks dispatched to the task specific agents by the Co-ordinator Agent. The task specific agents can change the record Status of table TaskList (with one of "waiting", "processing", "dispatched", "monitoring", "finish", or "delivered"). Every agent can watch the status changes in the Blackboard Status Area by starting a Java thread to monitor Status. The Blackboard Private Message Area is a Java vector object in our implementation. Each agent has a vector object that holds the interaction message for a given task. When the task is completed, the messages are removed from the vector. The Blackboard Knowledge Area holds the raw data, user profile settings and

trading strategies. All of these are stored as database tables across five databases (HTDB, RTDB, TSDB, FIDB, and UPDB in figure 5). MASST agents can access these tables through a JDBC-ODBC connection.

## 9. Some experimental results

For the purpose of validating our framework prototype, we use real trading data collected from the China Stock Market (Shengzhen Stock Exchange). This trading data, used to test the functioning of our prototype, covers the period 14/06/2000 to 14/02/2001. A number of experiments have been made [23]. Only two of our experiments can be shown here because of space limitations.

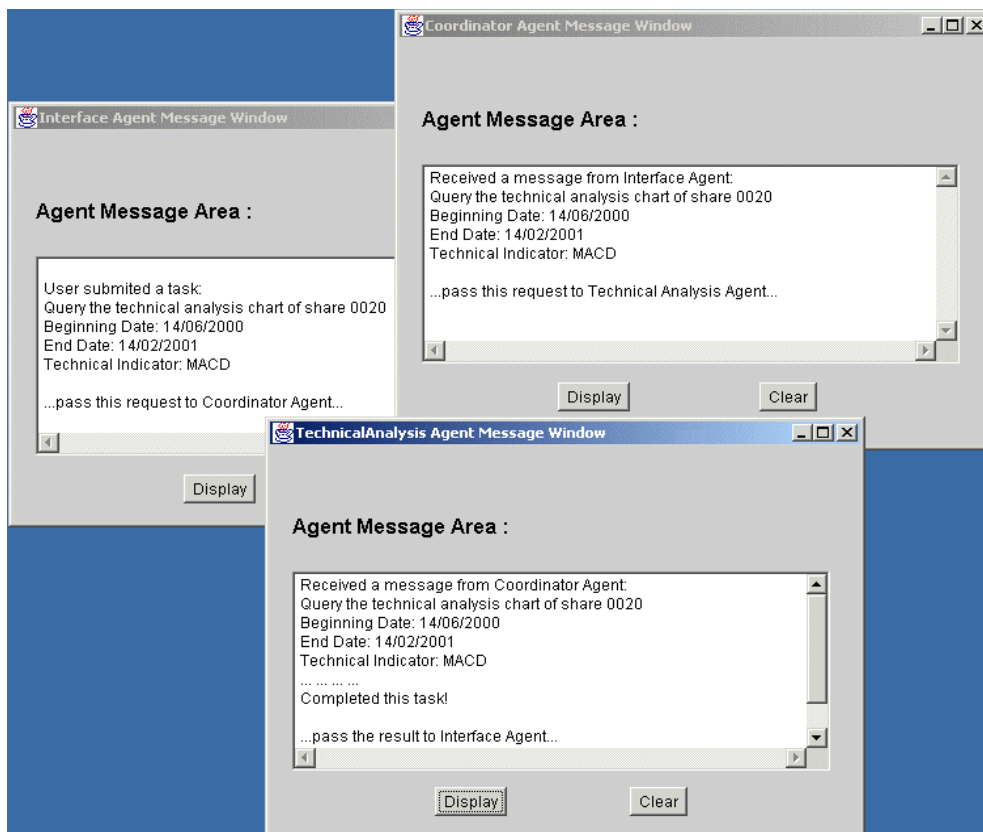


Figure 7. The MASST agents' interactions for the SearchAction5 request.

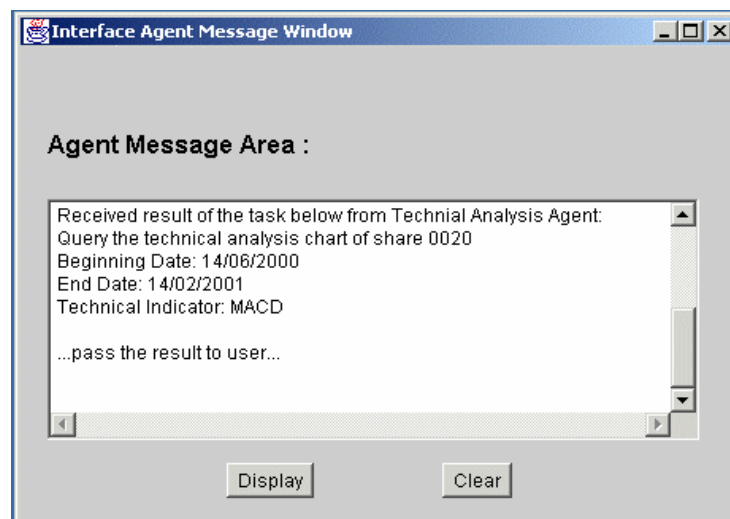
### 9.1 Example 1: information retrieval

This example shows how the system reacts to a SearchAction5 task (*'Query a given share's price and indicator charts over a time period'*). For this task, the user must have provided four parameters: StockCode, Beginning Date, End Date, and the name of the Technical Indicator. An example of such a user request is, *"Query share 0020's MACD (Moving Average Convergence/Divergence) chart from 14/06/2000 to 14/02/2001"*. The servlet (SC5Response in this prototype) collects these parameters and passes them to the MASST agents. The MASST agents interact with the system data source (a MS-Access database in this prototype) and create the related charts (Day Price-Chart, Day Volume-Chart, and Day MACD-Chart). The MASST agents return the result to the servlet SC5Response. Finally, the servlet sends the result as a HTML page to the user's browser.

Users interact with the system through the client's browser. In order to demonstrate how the agents interact with each other, we developed an external interface for each agent to display their interaction messages in a human understandable form. Thus the interactions between the MASST agents are transparent to the user. Figures 7 and 8 show the interactions between the MASST agents for the above task (SearchAction5). The servlet SC5Response passes the user delegated task to the Interface Agent and sets the Status field in the Blackboard Status Area to "waiting". The Interface Agent then passes the request to the Co-ordinator Agent by packaging the function ID (SA5 in this case) and the relevant parameters (i.e. Stock-Code, Beginning Date, End Date, and the name of the Technical Indicator). In the meantime, it sets the Blackboard Status to "processing". The Co-ordinator Agent then distributes the task to relevant task specific agent based on the function ID, in this case the Technical Analysis Agent. It then sets the Blackboard Status to "dispatched". The Technical Analysis Agent interacts with the Decision-Enabling Warehouse through a JDBC-ODBC connection and SQL to obtain the database records. It calculates the value of the given indicator and creates related technical analysis charts as the result. The Technical Analysis Agent sends the result back to the Interface Agent, and sets the Blackboard Status to "finish". Finally, the Interface Agent presents the results to the user (using the servlet SC5Response to send a HTML page containing the related charts), and sets the Blackboard Status to "delivered". This means the whole conversation is complete and the Co-ordinator Agent can delete the record from the Blackboard Status Area.

## 9.2 Example 2: decision support for buying/selling shares

From the investors' perspective, a vital concern when investing in the Stock Market is the buying and selling of share issues. Which share is the best one to buy? What time is the right time to buy the share and 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 provides buying and selling decision support based on the investment behaviour rules defined by investors themselves. MASST also provides a mechanism to let users define their own trading strategies (through 'User Profile Settings' on the MASST home page).



**Figure 8. Response from the MASST agents for the SearchAction5 request.**

For the task of decision support for buying/selling a given share, the user must provide two parameters: stock code and the date of decision-making. Before submitting the task to the MASST agents, the user should go to the page "User Profile Settings" to define the strategies for buying/selling a stated share. The MASST agents can subsequently give the user a suggestion to buy or sell the given share on the given date. In this experiment, the share's code is 0020, and the decision-making period runs from 14/06/2000 to 14/02/2001. MASST records its recommendation (i.e. "Buy" or "Sell" signals) for every day over this period. The results are shown in Table 1. In order to show the effectiveness of this experiment, we have

overlaid these “Buy” and “Sell” signals on the price chart of the share 0020 (shown in figure 9). It is clear from this figure that the MASST generates “Buy” signals around the lowest or relatively lower points and “Sell” signals around the highest or relatively high points.

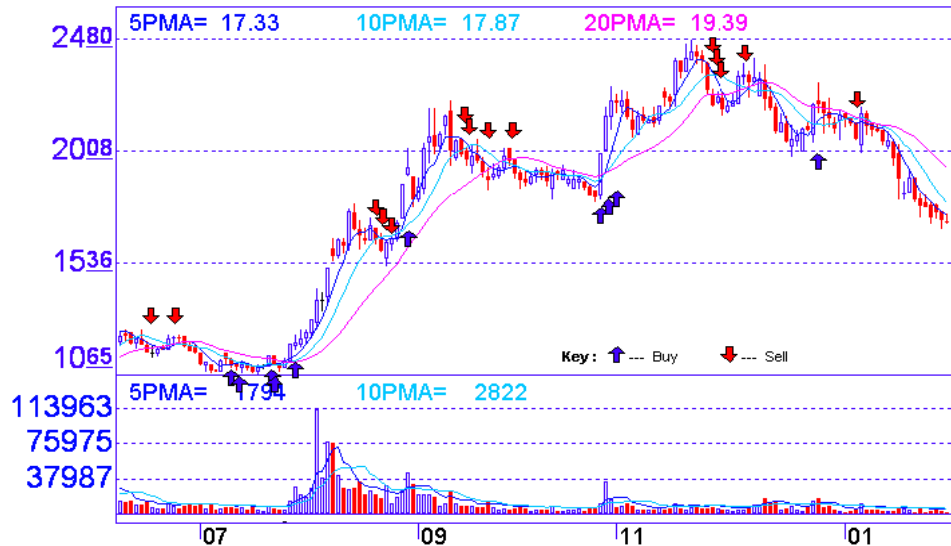


Figure 9. An experiment in decision support for the buying/selling of shares.

Table 1. The results of the Decision Making Agent for buying/selling the share 0020.

Date	“Buy” or “Sell” Signals
19/06/2000, 26/06/2000	Sell
11/07/2000, 13/07/2000, 20/07/2000, 21/07/2000, 27/07/2000	Buy
21/08/2000, 22/08/2000, 24/08/2000	Sell
28/08/2000	Buy
13/09/2000, 14/09/2000, 19/09/2000, 26/09/2000	Sell
27/10/2000, 30/10/2000, 31/10/2000	Buy
24/11/2000, 27/11/2000, 28/11/2000, 05/12/2000	Sell
22/12/2000	Buy
04/01/2001	Sell

## 10. Conclusion and Future Work

This paper has described the development of a multiple agent (knowledge-based) framework for stock trading that can assist investors in the decision-making associated with making investments in the stock market. The nature of the decision support is driven by the requirements of the users’ investment profile. The MASST framework does not guarantee the users that they will make money from the Stock Market. It aims to help the investors to make better buying or selling decisions in their daily investment activities and reduce the workload of the investors in analysing investment-related information. Whether or not a user makes a profit from the Stock Market depends largely upon the investors’ trading strategies.

The design and development of this framework has been guided by a combination of the KADS and ZEUS methodologies. It has been shown that the contribution of KADS is most influential and pervasive during the knowledge acquisition and analysis stages. We consider KADS to be a valuable tool in the hands of KBS developers, and its use is appropriate to the tasks associated with this domain. Its greatest strengths lie in the provision of a structure, which can be used as a starting point for system implementation. ZEUS is a powerful and integrated environment for the rapid building of collaborative agent applications. The

combination of KADS with ZEUS is a viable approach for the rapid analysis, design and implementation of a multi-agent based e-commerce applications.

This paper presented the multi-agent framework for the management of stock trading in terms of information access, filtering and integration. Descriptions have been provided for the various agent types that are necessary for supporting decision support and seamlessly integrating information gathering from distributed internet-based information sources. We believe that such a flexible multi-agent architecture, consisting of reusable agent components, will be able to fit the requirements for systems to be used in stock trading. These requirements include locating, accessing, filtering and integrating information from disparate information sources, monitoring the financial environment, notifying the investors about events of particular interest, performing user-designated tasks, and incorporating retrieved information into decision support tasks.

There are many online trading systems that provide basic technical analysis including charts analysis and trading data display in real-time. However, they do not provide intelligent decision-making support. The final analysis is actually carried out by the users. They do not adopt the agent-oriented approach and the programs cannot be executed autonomously. However, these existing systems are a useful information source for the raw trading data, relevant news reports, and the comments from the market analysts that MASST requires. It is feasible for the MASST system to interact with these existing systems. One current project looks to extend the MASST framework with text mining agents to facilitate such information gathering processes.

Further work involves the full implementation of the MASST framework. Traditional intelligent decision support methods and soft computing techniques, such as the neural network, fuzzy logic, and/or genetic algorithms, can be used in the full implementation of each task specific agent to improve the performance of the domain-level problem solving. For example, we may derive a set of optimal technical indicator parameters for each share by using neural network techniques or we can give the users more reasonable suggestions for decision support by using fuzzy logic techniques. As any of these enhancements could be another research project on its own right, we leave them for future research.

## References

- [1] Schreiber, G., B. Weilinga, and J. Breuker. (1993). *KADS: A Principled Approach to Knowledge-based Systems*. Academic Press, London.
- [2] Nwana, H., D. Ndumu, L. Lee, and J. Collis. (1999). "ZEUS: A toolkit for building distributed multi-agent systems", *Applied Artificial Intelligence Journal* 13(1), 129-186.
- [3] Sycara, K., D. Zeng, and K. Decker. (1998). "Intelligent Agent in Portfolio Management", In Jennings, N.R. and Wooldridge M.J. (Eds.), *Agent Technology: Foundations, Applications, and Markets*, Springer, 267-281.
- [4] Rus, D. and D. Subramanian. (1997). "Customizing Information Capture and Access", *ACM Transactions on Information Systems* 15(1), 67-101.
- [5] Benos, A. and E. Tzafestas. (1997) "Alternative distributed models for the comparative study of stock market phenomenon", *Information Sciences* 99(3-4), 137-157.
- [6] Bui, T. and J. Lee. (1999). "An Agent-Based Framework for Building Decision Support Systems", *Decision Support Systems* 25(3), 225-237.
- [7] Delgado, M., A.F. GomezSkarmeta, J.G. MarinBlazquez, and H.M. Barbera. (1999). "A Multi-Agent Architecture for Fuzzy Modeling", *International Journal of Intelligent Systems* 14(3), 305-329.
- [8] Burmeister, B. (1996). "Models and methodology for agent-oriented analysis and design", in Fischer, K. (Ed), *Working Notes of the KI'96 Workshop on Agent-Oriented Programming and Distributed Systems*, DFKI Document D-96-06.
- [9] Kinny, D., M.P. Georgeff, and A.S. Rao. (1996). "A methodology and modelling technique for systems of BDI agents" in W. Van De Velde and J.W. Perram, (Eds.), *Agents Breaking Away – 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (WAAMAW'96)*, volume 1038 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, 56-71.
- [10] Glaser, N. (1996). *Contribution to Knowledge Modelling in a Multi-Agent Framework (the CoMoMAS Approach)*. PhD thesis, L'Universit e Henri Poincar e, Nancy I, France.

- [11] Iglesias, C.A., M. Garijo, J.C. Gonzalez and J.R. Velasco. (1998). "Analysis and design of multi-agent systems using MAS-CommonKADS". in INTELLIGENT AGENTS IV: Agent Theories, Architectures, and Languages, LNAI 1365, Springer Verlag, 313-326.
- [12] Markowitz, H. (1991). Portfolio selection: efficient diversification of investment. Cambridge, MA: B. Blackwell, second edition.
- [13] Kuo, R.J. (1998). "A Decision Support System for the Stock Market Through Integration of Fuzzy Neural Networks and Fuzzy Delphi", Applied Artificial Intelligence 12, 501-520.
- [14] Saad, E., D. Prokhorov and D. Wunsch. (1998). "Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks", IEEE Transactions on Neural Networks 9(6), 1456-1469.
- [15] Chou, S.T., H. Hsu, C. Yang, and F. Lai. (1997). "A stock selection DSS combining AI and technical analysis", Annals of Operations Research 75, 335-353.
- [16] Lee, K.C. and W.C. Kim. (1995). "Integration of human knowledge and machine knowledge by using fuzzy post adjustment: its performance in stock market timing prediction", Expert System 12(4), 331-337.
- [17] Achelis, S.B. (1995). Technical Analysis From A to Z, Second Printing, New York, NY: McGraw-Hill.
- [18] Qing, M. (1997). Essential of Patterns Analysis, Lingnan Art Press, China.
- [19] Nison, S. (1991). Japanese Candlestick Charting Techniques. New York, NY: New York Institute of Finance.
- [20] Edwards, R.D. and J. Magee. (1974). Technical Analysis of Stock Trends, John Magee, Inc.
- [21] Baldwin, N.S. and R.E. Rice. (1997). "Information-seeking behavior of securities analysts: Individual and Institutional influences, information sources and channels, and outcomes", Journal of The American Society for Information Science 48(8), 674-693.
- [22] Davis, D.N. (2000). "Agent-Based Decision Support Framework for Water Supply Infrastructure and Development" International Journal of Computers, Environment and Urban Systems, 24, 173-190.
- [23] Luo, Y. (2001). Multi-Agent System Approach in E-Commerce -- A MASST Framework for Decision Support in Stock Trading. Ph.D. Thesis, University of Stafford, UK.