USING ONTOLOGY IN DESINGING A MULTI-AGENT SYSTEM

Chiung-Hui Leon Lee and Alan Liu

Department of Electrical Engineering,
National Chung Cheng University, Chiayi, Taiwan, R.O.C.
Email: leon@ai.ee.ccu.edu.tw aliu@ee.ccu.edu.tw

ABSTRACT

A multi-agent architecture is a distributed agents architecture which is derived from the distributed system concept and emphasizes on agent independence, and the cooperation between agents. For defining on intelligent agent, we divide a general agent framework into four components, the communication component, the control component, the agent knowledge component, and the execution monitoring component. A five-level approach has been proposed in agent system design: the detail level, the intermediate level, the policy level, abstract level, and the organization level.

We apply the concept of ontology to implement the agent knowledge. The advantage of ontology is knowledge sharing and reuse, and these advantages can facilitate the cooperation between agents. The conceptual graph is a good approach to implement ontology. Using the technical of multi-agent system (MAS), we proposed an intelligent stock investment system as application.

1.INTRODUCTION

Some people might believe that the more information you get, the more things you understand. With the increasing amount of information collected, there are difficulties in filtering, evaluating, and using information in problem solving. Therefore, besides the problem of locating information source, accessing, filtering, and integrating information become very critical. The notion of multi-agent systems (MAS) offers solutions [1-4]. The MAS access, filter, evaluate and integrate the information actively and asynchronously. By gathering more data they help users to make more right decisions. In the field of software engineering, there are many useful techniques being developed for a software design. Just like those software engineering techniques, the agent based system technology has generated lots of excitement and perspective in recent years. It introduces a new paradigm for providing conceptualized design and implementing software systems. This approach is particularly attractive for creating a distributed system in an open environment such as Internet [5].

The MAS have many advantages to be used as an approach to solve a complex problem, but there are still many issues and challenges about a MAS. Sycara has proposed these advantage and issues in [5]. The multi-agent approach can enhance the performance of a system like computational efficiency, reliability, extensibility, robustness, maintainability, responsiveness, flexibility, and reuse. The issues and challenges in MAS can be summarized as:

. problem allocation and result synthesis,

. communication and interaction,

. coherence,

. coordinating,

. conflict resolution,

. technology platform and methodologies.

Because of the issues and challenges which discussed above, we propose an agent framework for MAS and apply it in the area of stock investment. We extend the idea of Tim Finin [6] in which an agent-based system design need to consider the following levels:

Detail level: how agents send and receive messages.

Intermediate level: what the individual message mean.

Policy level: how agents structure conversations.

Abstract level: how each agent constructs the knowledge about itself and the environment.

Organization level: how to connect systems in accordance with constituent protocols.

Considering those levels, there are four components of an agent, which need to be considered such as the communication component, the control component, the agent knowledge component, and the execution monitoring component. The communication component offers the function for the detail level, the intermediate level, and the policy level. The agent knowledge offers the function for the abstract level. The control component and execution monitoring component offers the function for the organization level.

In Section 2, we take a look at the approach in agent knowledge representation. Section 3 presents one application of our agent framework and proposes design methods. Finally, we suggest our future works and conclusions in Section 4.

2 APPROACH IN AGENT KNOWLEDGE REPRESENTATION

In this section, we introduce how we use ontology approach to represent the knowledge of agent and implement ontology using conceptual graph. Well-designed agent knowledge plays an important role for problem solving and decision making in the development of an agent-based system. Also we believe a well-designed agent knowledge having the property of sharing, and reuse will facilitate agent cooperation and negotiation. There are two good reasons for using ontology to organize the agent knowledge. It allows for a more disciplined design of the agent, and it facilitates knowledge sharing and reuse [7].

2.1 Knowledge sharing and reuse of agents through ontology

We use stock investment as an illustration to present our idea, when we need to make a good stock investment decision, we may watch the TV or newspaper that try to find useful information, or communicate with stock investment agent for investing suggestion. Because we all use the same language – Chinese / English etc., we all understand the basic element of the subject under discussion--stocks. If an investor is not an expert of stock market, he/she would rather have his/her investment advice software agent to collect the information and contact other stock analysis agents for making decision. How to make agents communicate with each other and let them agree the word “stock”? By the property of sharing and reuse, ontology is a good approach. The word "ontology" has a long history in philosophy, in which it refers to the subject of existence. One good definition of "ontology" is *"the branch of metaphysics that deals with the nature of being."* [8] and is also *"a specification of a conceptualization"* [9].

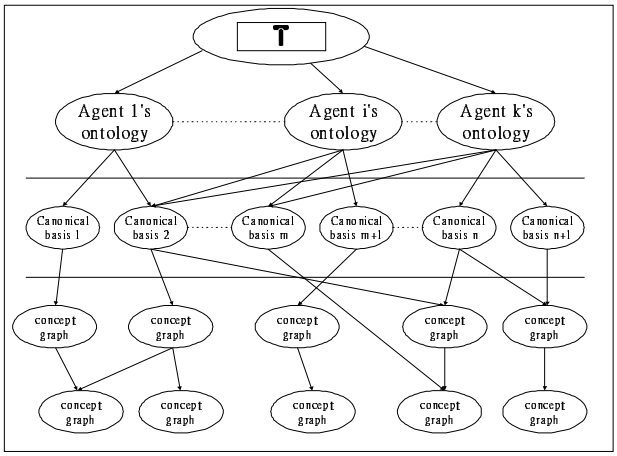

Figure 1. The hierarchy of agent’s ontology.

We implement ontology with the idea of conceptual graph [10] in this paper. By this approach, we divide our agent knowledge into three levels including agent’s ontology level, canonical basis level, and concept graph level. An agent’s ontology consists of many canonical bases which is shared, reused and cannot refined among agents. From these canonical graphs we can derive the concept which we want to indicate. Fig.1 shows this approach. Each agent has their ontology and the canonical bases is shared with acquaintance to implement environment knowledge. The concept graphs is derived from canonical bases, and different agents have different concept graphs to implement the domain knowledge.

2.2 Implementing ontology in a conceptual graph

There are various approaches to implement ontology. We choose the conceptual graph because it is equivalent to predicate calculus in their expressive power, and the mapping is straightforward from conceptual graphs into predicate calculus notation. The type hierarchy of conceptual graph shown on Fig.2 is a lattice, a common form of multiple inheritance systems. Consider two types x and y, if $y \leq x$ that y is said to be a sub-type of x and x is said to be a super-type of y. A type may have more than one super-type and more than one sub-type. A common sub-type or super-type might exist. However every pair of types must have a minimal common super-type and a maximal common sub-type. Two special types exist. A super-type of all types is called universal type, and a sub-type of all types is called absurd type.

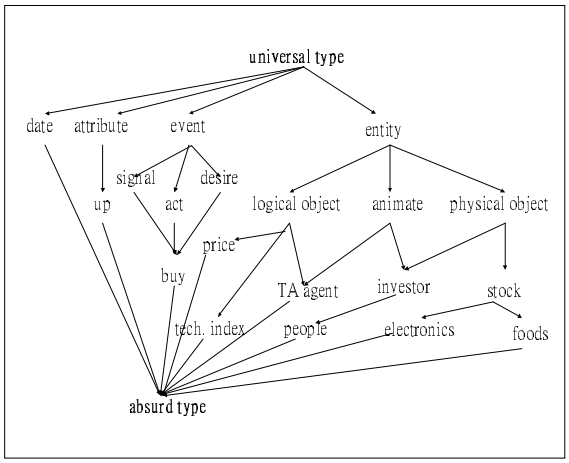

Figure 2. The sample concept type hierarchy.

From existing graph to new graph, there are a number of operations in the theory of conceptual graphs. The operation is *copy*, *restrict*, *fuse*, *join*, and *simplify* [11]. These five operations are called canonical formation rules. They are not rules of inference and they cannot guarantee to derive a true graph from a true graph. Although they are not sound inference rules, canonical formation rules form a basis for much of the plausible reasoning [12]. As inheritance can be implemented by *fuse*, *join*, and *restrict* operation, Fig.3 shows how foods stocks inherit the property of having price from the class stock by replacing a type label with its sub-type. It also shows how the individual, VeWong, inherits this property by instantiating a generic concept. Similarly, we can use *fuse*, *join* and

restriction operations to implement the prediction ability of agents. If the technical analysis agent is told that "Give me the technical index of the stock VeWong today". It might automatically make a number of assumptions: the user wants to buy the stock VeWong, the user needs investment advice, there is a need to cooperate with technical analysis agent, etc. This reasoning can be done using *fuse*, *join* and *restrict* operation. The agent forms a conceptual graph of user's request and joins it with the conceptual graphs for price, stock VeWong and date. The resulting graph lets the agent assume that it needs to cooperate with other agent and to give an invest advice to the user.

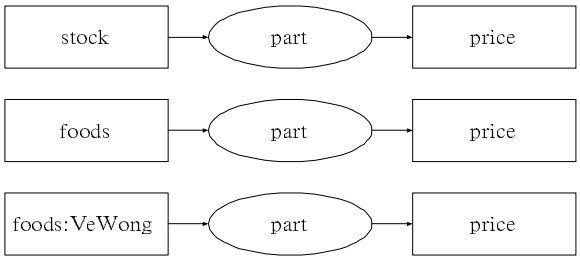

Figure. 3. Inheritance in conceptual graphs.

2.3 Storing and retrieving agent ontology

The agent knowledge base is divided into two models, environment model and agent-self model, in our agent architecture. The environment model contains the information of related peers and resources. The agent-self model contains the knowledge of problem solving or decision making. The environment model is implement by canonical basis. Every canonical basis is shared with related agents. By these canonical bases and some rules, an agent can infer the information of other agents, like their ability, desire, location etc. to facilitate their cooperation. The agent-self model can be derived from these canonical bases. Because every ontology in agents are implemented by a conceptual graph, and the common data structure used to store conceptual graphs is a hierarchy, we can reuse the storing and retrieving algorithm in conceptual graph [11].

3. APPLICATION — INTELLIGENT STOCK ANALYSIS SYSTEM

Our agent framework presented before can be used in many applications in which a complex problem exists. In this section, we apply our framework in an application—an intelligent stock analysis system. The whole system architecture is shown in Fig.4. Because there are many factors need to be analyzed, such as the stock price, industrial category, company, investor, etc., the stock analysis is one of the most complex problems in the world. Everyday, investors are provided with lots of news, financial data, and rumors. More information can help the people solve the problem better but lots of information often might confuse him from making a clear and prompt decision. There are many existing systems trying to help the investor analyze the stock market, and different techniques are employed like rule base [13,14], blackboard [15], artificial neural networks [16,17], genetic algorithm [18], etc. They just analyze a part of the factor which affects the stock price.

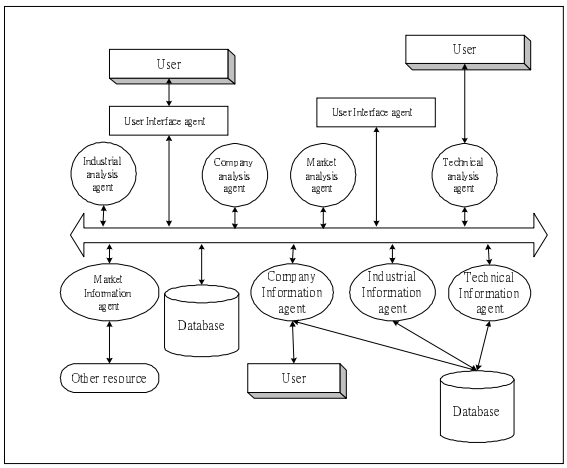

Figure 4. The intelligent stock analysis system architecture.

A good stock selection approach includes two phases. First, an investor needs to analyze the stock index, change-rate, foreign-trade, price, the balance of company, etc. and choose a company worthy of investing, and this step is also called fundamental analysis. Second the investor uses the technical index like the mean average curve, bias index, relative strength index, etc. to determine when to invest the stock. In Section 3.1 we will discuss the system requirements when creating an intelligent stock analysis system, Section 3.2 shows the system architecture, Section 3.3 discusses the system function, and Section 3.4 discusses the goal and advantage when we implement the intelligent stock analysis system using our MAS approach.

3.1 System requirements

The intelligent stock analysis system provides the user investment advice, displays the investment information which user needs, and acquires/filters the investment information for user. For the purpose of making a good investment decision, the system needs to make some analyses like the fundamental analysis and the technical analysis. By [19], the fundamental analysis is divide into three parts, market analysis, industrial analysis, and company analysis. Such analysis is shown below:

Market analysis (e.g. Taiwan market): analyzing the factor like stock-price index, prosperity cyclical, interest rate, change rate, foreign-trade, etc. determining the whole market if worthy of investing.

Industrial analysis (e.g. electronic industry): analyzing the factor like industry belongs to growth, defensive or cyclical industry, and industry competing situation, etc. determining the "HOT" industry.

Company analysis (e.g. VeWong company): analyzing the balance sheet, income statement, and profitability ratio, etc. determining which company worthy of investing.

3.2 System architecture

By the objective of service, there are three kinds of agents in our MAS: user interface agent, analysis agent, and information agent. The analysis agent and the information agent can be divided into four types: market, industrial, company and technical. The agent hierarchy is shown in Fig.5. Every agent has its own interface to interact with users, other agents, related database, and other resource. Because the agent in the net is dynamic, the agents in our architecture are loosely coupled. Each agent can work alone, or cooperate with others. The user interface agent's agent-self knowledge stores the user model in which record the user name, user's identification number (ID), and the user's investing record etc. This information can help the agent make a more suitable decision for investment like investing combination. An example of user model is shown in Fig.6 which means that an investor named Lee whose ID is Q123456789 and own three piece of stock VeWong and two piece of stock GVC. The analysis agent is the kernel of problem solving and decision making. Its agent-self knowledge stores the information for market, industrial, company, or technical analysis. It receives a request from user or other agents, analyzes the data supplied from information agent, database, or user, and sends the result to the requester. The information agents collect and filter the information from user, other agents, or the web and maintain the related database. Because of the large amount of stock data, we also need the related database to store the history data.

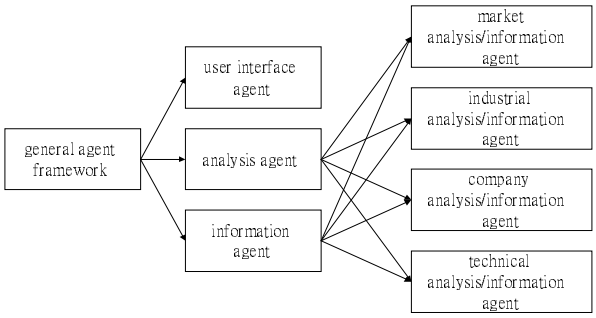

Figure 5. The agent hierarchy.

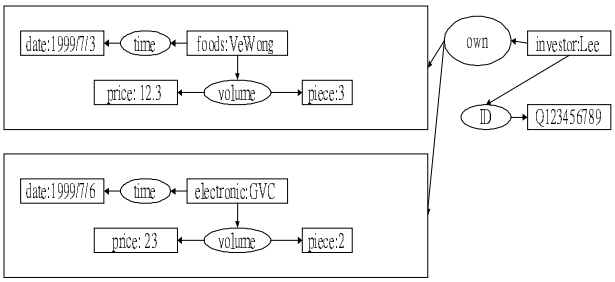

Figure 6. An example of user model.

3.3 System functions

Our intelligent stock analysis system offers several functions for query, advice, and manipulation of the information about stock. The basic operation modes are the analysis mode and the advice mode. The analysis mode does not give any advice for users, but displays the information what the user are interested. In this mode our system act like a distributed database management system. The advantages of such system are dynamical and automatic. The difference between our system and traditional database is that our system is self-adjustable through learning. The advice mode accepts the user's request, gives the advice for investment, and has an explanation for the advice. The user can use this mode to determine the investment policy.

3.4 Design methods

Considering the levels of an agent-based system design, we propose constructing the detail level, the intermediate level, the policy level, the abstract level, and the organization level to implement the MAS. When implementing the agent framework, these five levels also can be considered in the communication component, the agent knowledge component, the control component, and the execution monitoring component. Fig.7 shows the five level of agent-based system design.

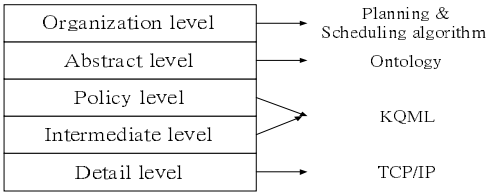

Figure 7. The five levels of agent-based system design

3.5 Discussion

The stock investment is a complex problem in which there are many factors need be considered and might not be solving in one agent. The MAS is a good solution for this challenge. The intelligent stock analysis system we designed used the techniques of our proposed methods in the previous sections. It acquires and filters the stock information from the environment resources, offers the processed information to users, and gives suggestions for stock investment. When the user is registered in the system and has more information stored, the system can give the portfolio and advice automatically. Our system realizes the objective of the MAS that is assisting users to make a good investment decision.

4. Conclusion And Future Work

We have designed a general agent framework used in MAS that has advantages of flexibility and efficiency. By the definition of the intelligent agent, the agent framework can be divided to four components, the communication component, the control component, the agent knowledge component, and the execution monitoring component. The communication component offers the ability of interacting with the environment. We consider problems of how to send the messages, what the individual message mean, which conversation protocol used, and how to make perception and action coordination. The control component determines the agent how to cooperate. The task is stored

in task hierarchy formalism. Because the MAS environment is varied dynamically, execution monitoring is essential to make sure the system is reliable. The agent knowledge sharing and reuse is an important issue and challenge in this paper. We proposed the approach that implements the agent ontology using conceptual graph to offer the knowledge sharing and reuse. These advantages will facilitate the cooperation of agents and reducing the agent communication load. We also can reuse the existed conceptual graph theories like searching/retrieving algorithm [11] and universal data structure [20], etc.

We propose the MAS architecture for stock investment using our agent framework. There are many agents existing in MAS, and by the objective of service that we divide these agents into three kinds, user interface agents, analysis agents, and information agents. By this architecture we can offer the intelligent service for the user. We believe this architecture also can offer the solution for other complex problem. One advantage of MAS is reuse. In our agent framework there are many modules which can be reused when implementing other applications such as perception, action, planning, knowledge management, etc.

There is still much work that needs to be done to improve our work. We divide our future works into two categories: the agent knowledge problem and the communication problem. The agent knowledge problem includes the issues of combinatorial explosion, retrieving efficiency, and adapting correctness. From the time goes by, the agent knowledge may become large, and the combinatorial explosion problem will appear. We need to filter the useless information, store the useful agent knowledge, and retrieve the agent knowledge efficiently. How to adapt the correct knowledge is another challenge. We can try the methodology proposed in [11] to reduce our agent knowledge. The communication problem includes the issues of problem allocation, result synthesis, negotiation, and cooperation. These problems are ad-hoc and there are many people researching in these problems.

5.ACKNOWLEDGMENT

This research is supported in part by National Science Council under grant NSC 89-2213-E-194-021.

6. REFERENCES

[1] B. Hayes-Roth, R. V. Gent, R. Reynolds, M. V. Johnson, and K. Wescourt "Web Guides " *IEEE Intelligent Systems*, pp. 23-27 Mar./Apr. 1999

[2] K. Sycara, A. Pannu, M. Williamson, and D. Zeng, "Distributed Intelligent Agents, " *IEEE Expert*, vol. 11, pp. 36-46, Dec. 1996

[3] J. A. Pastor, S. L. Taylor, D. P. McKay, and R. McEntire, "An Architecture for Intelligent Resource Agents, " pp151-159 *Proc. of CoopIS '97*. June 1997

[4] N. R. Jennings and E.H. Mamdani, "Using Archon to Develop Real-World DAI Applications, Part 1" *IEEE Expert*, pp. 64-70 1996

[5] K. P. Sycara, "Multiagent System, " *AI Magazine*, pp.79-92, Summer 1998

[6] T. Finin et al.. "KQML : An Information and Knowledge Exchange Protocol, " in *Knowledge Building and Knowledge Sharing*, K. Fuchi and T. Yohoi, eds., *Ohmasha and IOS Press*, 1994.

[7] N.J.I. Mars, "The Role Of Ontologies In Structuring Large Knowledge Bases, " *Knowledge Building and Knowledge Sharing*, K. Fuchi and T. Yokoi, eds., pp240-248, *Tokyo: Ohmsha*, 1994.

[8] B. Chandrasekaran and J. R. Josephson, "What are ontologies, and why do we need them? " *IEEE Intelligent systems*, Jan./Feb., pp. 20-26, 1999

[9] http://www-ksl.stanford.edu/kst/what-is-an-ontology.html

[10] J.F. Sowa, *Conceptual structures: information processing in Mind and Machine*, Addison-Wesley, Reading, Mass., 1984

[11] G. Ellis, "Compiling conceptual graphs, " *IEEE Tran. on knowledge and data engineering*, Vol. 7, No. 1, pp. 68-81, Feb. 1995

[12] G. F. Luger and W. A. Stubblefield, *Artificial Intelligence – Structures And Strategies For Complex Problem Solving, third eds.*, Reading MA UAS, Addison Wesley Longman Inc, 1998

[13] N. K. Liu and K. K. Lee "An Intelligent Business Advisor System For Stock Investment, " *IEEE Expert System*, Vol. 14, No. 3, pp.129-130, Aug. 1997

[14] G. H. John, P. Miller, and R. Kerber " Stock Selection Using Rule Induction, " *IEEE Expert system*, Vol. 14, No. 3, pp. 129-139, Aug. 1997

[15] S. Vranes, M. Stanojevic, V. Stevanovic, and M. Lucin, "INVEX: Investment Advisory Expert System, " *IEEE Expert System*, Vol. 13, No. 2, May 1996

[16] G. S. Jang, F. Lai, B. W. Jiang and L. H. Chien "An Intelligent Trend Prediction And Reversal Recognition System Using Dual-Module Neural Network, " *IEEE*, pp.42-51, 1991

[17] J. Roman and A. Jameel, "Back-Propagation And Recurrent Neural Network In Financial Analysis Of Multiple Stock Market Returns, " *Proc. of the 29th annual Hawaii int. conf. On system sciences*, pp.454-460, 1996

[18] F. Kai and X. Wenhua, "Training Neural With Genetic Algorithm For Forecasting The Stock Price Index, " *Int. Conf. On Intelligent processing systems*, pp.401-403, 1997

[19] J. C. P. Shieh, *Contemporary Investments—Analysis And Management*, Taipei, Taiwan, Best-Wise, 1998

[20] Robert Levinson, "UDS: A universal data structure, " *Proc. ICCS '94*, pp. 231-250, Aug. 1994