A Society of Natural Language Agents on the Internet

Wen-Lian Hsu, Yi-Shiou Chen and Yuan-Kai Wang
Institute of Information Science
Academia Sinica
Taipei, Taiwan, Republic of China

Abstract

As the Internet is flooded with different types of documents (unstructured, semi-structured texts) and heterogeneous databases, information retrieval and extraction become increasingly difficult. To deal with the information service problem on the Internet, we are experimenting with an agent society in which agents act as spokesperson for various web pages, databases and provide information integration services in a cooperative manner. Furthermore, these agents are equipped with various levels of natural language abilities. These agents are particularly useful in information retrieval, extraction and integration services where dialogues in natural language are appropriate.

We have constructed an agent system (which is actually running on the Internet) called *NIA* with around 20 agents each capable of extracting certain information on the Internet such as traffic, travel, stock and financial information. We are currently integrating various types of information provided by these agents for decision making as well as planning. Preliminary experiments indicate that this approach is very promising.

1. Introduction

The abundance of information on the Internet has made things difficult to search for since an important piece of information is often buried within a sea of seemingly relevant information. A brute-force search algorithm that does not understand the purpose of a user is no longer satisfactory [2]. An ideal search algorithm should be able to (1) understand, to a large degree, users' queries; (2) understand roughly the content of each web page and database; (3) make an appropriate connection between a user's query and the information in relevant web pages and databases.

Besides searching, there are many other important services on the Internet such as data mining and information extraction. Many of these tasks require tremendous amount of domain knowledge as well as the capability of utilizing the knowledge [3,4]. Similar problems have been addressed by Hammer et al. [5] and Ambient et al

[6]. We propose to deal with the above problem by building an agent society in which agents act as spokesperson for various web pages, databases and provide information integration services in a cooperative manner. Furthermore, these agents can communicate with *natural language* (*NL*). Such natural language is sufficiently constrained so as to avoid ambiguity as much as possible. These agents are particularly useful in information retrieval, extraction and integration services where dialogues in natural language are appropriate.

The task of understanding natural language is quite formidable. Most of the terms we used are not "clearly" defined and their "interpretations" could vary among different people. For example, we often hear people say that he or she has a friend who is a *good tennis player*. The fact is that everyone has a different notion about how a *good* tennis player (or good student, good friend, …etc.) should perform on the court. Furthermore, such a notion can be very fuzzy even for the same person since one *good* player in his mind could be significantly better than another *good* player. Thus, most people only have a rough idea about his or her own standard of a *good player* and it would be difficult to stipulate explicitly the conditions that a good player should satisfy. An interface agent has to deal with issue when interacting with the user directly.

To support the use of *NL* in knowledge representation and utilization, we shall construct a framework by adopting a *context sensitive model (CSM) for concept understanding* [7]. In such a model, the level of understanding depends on the kind of queries that can be answered correctly by the agents. We shall start with agents that are capable of answering a few queries and gradually upgrade their ability to be able to answer a high percentage of frequently asked questions (*FAQ*). Such a model allows us to set up an *NL* protocol for the society so that each agent can perform the following:

(1) advertise what services it can provide to other agents (or the users);
(2) understand and make use of those services provided by other agents (such as generating appropriate queries for other agents and utilizing those answers to enhance its own service);
(3) have limited *NL* query answering ability.

Since knowledge on the web is very extensive, we shall first focus on understanding a limited number of domains such as travel, electronic commerce and etc. (those that are most popular with Internet users). We equipped our agent with some basic NL ability to deal with *FAQ* within these domains. There are several advantages of using *NL*:

(1) It becomes easier for these agents to deal with human users directly (when they are capable of answering a high percentage of *FAQ*). .

(2) By constructing, for each database, an agent that can communicate with RNL, our NL agent society would likely resolve the problem of integrating heterogeneous databases satisfactorily.

(3) This approach would eventually make it possible for non-programmers to write NL scripts to manipulate these agents.

We shall refer to our NL agents as *natural intelligent agents* (*NIA*). Our *NIA* system has around 20 agents each is capable of answering queries by extracting certain information on the Internet such as traffic, travel, stock and financial information. We are currently at the stage of integrating various types of information provided by these agents for decision making as well as planning. Preliminary experiments indicate that this approach is very promising.

This paper is arranged as follows. In the next section we introduce our model for concept understanding. Agent architecture is discussed in Section 3. Different roles of agents such as database agents, broker agents and web tag agents are described in the following three sections. Section 7 gives several prototype agent systems that are currently under construction. Finally, we discuss some future research in Section 8.

2. A Context Sensitive Model for Concept Understanding

In this section we shall describe our model for *concept understanding*. It should be emphasized that, since this framework is to be implemented in a computer to build a simulation model, our definition has a strong "problem-solving" flavor. In other words, this approach is not meant to be a cognitive study of the true nature of human understanding, it merely represents a system designer's viewpoint for a plausible simulation. Our model has the property that the level of understanding depends on the kind of queries that can be answered correctly.

The model presented here follows the one discussed in [7]. Our basic assumption is that the specific schemes for representing the meaning of a concept is not crucial as long as they can provide *satisfactory* answers to many basic questions related to this concept. Since a concept is almost never considered outside its context, we shall refer to it as a *contextual concept*. Each contextual concept *C* will be associated with a set *Q* of queries. This *CQ*-pair, (concept name, query set), serves as a description of the contextual concept. For example, the *FAQ* about a concept could serve as a good example of the query set *Q*, though in practice the query set may be much larger. If the query set *Q* is obtained from a particular person's view, we say that the *CQ*-pair is a description of the contextual concept *in the mind of* that person. Thus, it is appropriate to think that the query set provides the context for this contextual concept.

For each *CQ*-pair description of a contextual concept, we say that a person

understands the contextual concept relative to its associated query set provided this person can achieve a satisfactory score in a test sampled from the associated query set. One can measure the level of understanding by the scores obtained in the test. Since different people might have different understanding about a contextual concept, this will be reflected in our model by associating different query sets with this contextual concept. Thus, our concept understanding model is, by definition, *context sensitive*. We shall refer to our model as the *CSM*. Two people are said to *agree with each other* on a contextual concept if each of them can pass the other's test.

Such a query-answering model is quite popular for testing human understanding since most tests given to people can be regarded as sampled query sets for the understanding of certain topics.

Another way of looking at our concept understanding model is that the above *CQ*-pair description provides a procedure for issuing a certificate (good for a certain period) that a person understands a contextual concept *C* relative to the query set *Q* much like the way a driver license is issued. This view is especially useful in our agent society. Since software agents are usually designed by different people, we cannot require that they all agree a hundred percent with the *definition* of services provided by each agent, especially when these definitions are now written in *NL*. Thus, some sort of testing (and certificate issuing) mechanism has to be set up before the agent society can function properly.

Another important aspect of our concept understanding model is that some sort of sampling techniques must be adopted in query selection for a test set. Since the entire query set associated with a concept can be quite huge, a smaller representative test set is often used in practice. Therefore, there is a probability factor involved in the test. A person scoring well against a specific query test yesterday might not necessarily score satisfactorily against a different query test today. This phenomenon provides a statistical uncertainty in the understanding of contextual concept, which is also very natural for human understanding. In the following we give a few more examples:

1. *The driving test*: In this case, the government stipulates a standard to issue driver license. The *CQ*-pair description can be (eligible driver, driving test), which can be regarded as government's view on an eligible driver. However, unlike a written test whose scores can be calculated unambiguously, a driving test has to be conducted by testing officers and personal judgement is inevitable. So these officers have to be trained to have a common sense on how to conduct the test. Now, a person failing the driving test yesterday is not guaranteed to pass the test today by only correcting the mistakes he or she made yesterday since a different test set might be adopted by a different testing officer. However, anyone who "really "knows" how to drive has a high probability of passing most tests.

2. *A man's understanding of his girlfriend's love towards him*: First of all, this is a contextual concept that his girlfriend may not necessarily agree with. Furthermore, such an understanding could fluctuate rapidly depending on various circumstances occurring at different times. Any delicate change in her behavior or gesture could have unpredictable interpretations.
3. *Law-abiding citizens*: A citizen who obeys the laws should at least "know" what most of the laws are. However, this is quite unlikely since government laws are so complex and change so rapidly that even the most sophisticated lawyer would have trouble keeping track of all the details, not to mention that the interpretation of each law could also be quite different. Therefore, a probability factor becomes a necessary component in such a qualitative assessment.

In the next section we illustrate how to apply our concept understanding model to agent advertising and verification.

3. The Agent Architecture and Agent Handbook in *NIA*

For ease of exposition, we shall divide agents in *NIA* into two types: database agents and broker agents. Those agents who access data directly from a database or a web page are called *database agents*. Those who fetch data indirectly through other agents are called *broker agents*, though such a distinction sometimes is fuzzy.

Each agent in *NIA* publishes a *handbook* containing the set of queries it can answer in *NL* as well as the format of answers. Take a database agent as an example. Denote by *A-handbook* the handbook of a database agent *A*.

In order for agent *A*'s handbook (or part of it) to be utilized by another agent *B* we require that agent *A registers* its handbook (or part of it) with agent *B*. Agent *A* is then said to be a *subagent* of *B* and agent *B* is said to *collect* the *A*-handbook. An agent *B* can collect many handbooks. However, for each handbook collected, agent *B* is expected to have enough domain knowledge to analyze queries in this handbook so that agent *B* can answer queries "about" this handbook. The least that agent *B* can do is to include queries in its collected handbooks in B-handbook so that it can direct relevant queries to the corresponding subagents through keyword matching.

A handbook in our agent architecture serves many purposes as follows:

(1) The *A*-handbook reveals what services agent *A* can provide for others and the mechanism for getting these services (by asking the "right" questions in the right format).

(2) Queries in *A*-handbook provide the test queries for other agents to verify that A is capable of getting correct answers for queries in A-handbook. This might not be as trivial as it seems. For example, agent A might claim that it can find all flights

between two cities. Since this schedule is constantly changing, A must be able to update this information automatically.

(3) Queries in *A*-handbook provide a goal for the designer of agent *A*. In order to ensure that a database agent *A* "understands" the database it must be able to utilize knowledge in the database to answer those queries through database commands.

(4) Handbooks and related knowledge provide an effective way of knowledge representation. The knowledge of a broker agent can be regarded as the aggregation of the knowledge in the handbooks of its subagents plus the additional integrated knowledge.

Therefore, the concept of a handbook is particularly suitable for the design of large distributed agent systems.

One of the main issues of current database research is to exchange data among heterogeneous databases. Often, special purpose protocols are designed to enable a specific group of databases to exchange data. However, such an approach is too costly to develop into a universally acceptable standard. Besides that, a host of other information sources such as web pages contain lots of semi-structured documents or unstructured texts and it is important to be able to extract information from these *HTML* documents. Our *NL* agents are designed specifically to handle such data.

There are several advantages in using *NL* as agent communication language. First of all, *NL* is the de facto standard for human communication. Because people share a lot of common sense, they rarely have to exchange "ontology" when communicate with *NL*. Hence, a software agent that understands more *NL* (including related common sense knowledge) can likely provide better service (for example, answer more difficult *FAQ*) for the user. Secondly, *NL*, as a communication protocol, appears to be the fastest solution for integrating information gathered from heterogeneous databases and unstructured web texts. This is because *NL* agents can be constructed independently by different people at different locations in parallel. Furthermore, each agent can be tested independently using *FAQ* in their respective domain.

Databases are normally managed by *MIS* people who have little training in *NLP*. Hence, we shall insert a layer of broker agents between the user and database agents so that database agents only need minimal command of NL and queries for them (issued through broker agents instead of the users) can be easily translated to database queries by the *MIS* people. A broker agent normally has better command of *NL* and can translate users' queries to a language that can be understood by database agents. A typical query for a database agent is encoded in the formatted natural language (*FNL*) query boxes we often see in the web. In the following sections we will delve into detailed discussion on the design of agent handbook and the incorporation of NL in

query formation.

4. Database Agents

In this section we describe a typical (dumb) database agent. The handbook of a database agent consists of a set of queries referred to as *FAQ*. Each *FAQ* contains the following four items:

(1) an *FNL* question

(2) a collection of word declarations

(3) a database query

(4) an answer template

An *FNL* question is a question written in NL in which certain words are designated as variables, and the semantics of these variables are specified in word declarations. A database query (ex. an *SQL* query) formed according to the *FNL* question and word declarations is used to retrieve information from database. An answer template describes the format of an answer in *FNL*. The following table gives an *FAQ* of an airline database agent:

FNL Question	What is the departure time of the earliest flight from Boston to Chicago on July 4th ?
Word Declaration	The earliest flight = flight number; Boston = city of U.S.; Chicago = city of U.S.; July 4th = date.
Database Query	Select time from scheduling where from = <Boston> and to = <Chicago> and flight time = <the smallest flight time> and date = < July 4th >.
Answer Template	6:00

Figure 1. The four items in an *FAQ* of a database agent

In this example, the *FNL* question shows that this *FAQ* can answer the earliest departure time, while Boston, Chicago, and July 4th are variable words that can be substituted by other words as listed in the word declarations. Database queries are used to retrieve an answer, which will be expressed in the “hour:min” format as specified by the answer template.

Items 1, 2 and 4 of an *FAQ* are used for communicating with broker agents. A database agent can register its handbook with several broker agents. The broker agent then tries to understand the database agent by analyzing the *FNL* questions, the word declarations, and the answer templates. The broker agent can test the database agent by sending test *FAQ* and verifying the answers. If the test result is satisfactory, the broker agent will include the queries in this handbook in its own handbook and start

utilizing its service by sending inquiries.

When a broker agent sends an inquiry to a database agent *A*, this inquiry must match an *FAQ* in *A*-handbook (essentially, the broker agent fills out an *FNL* question with permissible words).

Note that we use the term "database agent" in a rather restricted sense to make contrast with broker agents. In other words, our database agents only perform the most primitive functions. Thus, one could refer to them as *dumb* database agents. Dumb database agents require very minimal effort on the *MIS* people. Whenever necessary, an *MIS* department can certainly upgrade their dumb database agent to a more powerful broker agent, which can interact directly with people.

5. Broker Agents

The most difficult and crucial part of NIA is the construction of various broker agents. The role of a broker agent is to support and enhance the communication between users and agents, and among agents themselves [14]. Sometimes, such communication can be established through several broker agents [15,16].

A broker agent publishes its handbook to users and other broker agents. It also collects handbooks from other agents. A broker agent needs to understand the domain knowledge of its subagents. It also has to be able to take more complicated queries from users or other agents than those given to database agents. Thus, a broker agent must have good command of *NL* and the ability to utilize the handbooks of its subagents. For example, in *NIA*, a travel agent that plans an itinerary for a customer must communicate with the airline agents, train agents, map agents, hotel agents and etc. In addition, the travel agent needs to know how to integrate the information gathered from all these subagents to create a feasible and economical schedule.

A broker agent answers a query in its own handbook by sending queries to its subagents and integrating their answers. Compared to a database agent whose *FAQ* must be "filled out" correctly, a broker agent should be more flexible in matching a user query. In our current *NIA* system, a broker agent operates based on a finite collection of *events*. Events are implemented using frames [17], which are used to match with sentences to extract corresponding events. Based on the frame techniques of *NLP*, a broker converts handbook queries of its subagents into events. The broker prepares a script for each of its *FAQ* event. The script contains natural language descriptions about how to carry out the event by sending queries to its subagents. Event formation and mapping are done by frame approximation. Hence, a broker agent is expected to understand queries that are close or synonymous to those in its handbook. An event concept is considered *understood* if the broker agent execute the

corresponding actions correctly.

To summarize, the tasks of a broker agent consist of the following:

(1) understanding the queries in its collected handbooks by converting them into appropriate events (Initially, this is done with human help, but we expect the computer will pick up a good portion later on)
(2) verifying that the queries in each handbook can be satisfactorily answered by sending test queries to the corresponding subagent
(3) identifying events in the user's query with one of its *FAQ*, preparing a script for each of its *FAQ* and sending queries to its subagents
(4) integrating subagents' answers to form an answer for the user's query

A broker agent can also be tested using queries in its handbook.

A script-like event description is useful in event decomposition and planning. The concept understanding mechanism of broker agents is implemented by the *CSM* described in section 2. Furthermore, a procedure of broker agents contains more instructions than a single database query. There are instructions for communicating with subagents, for calling other *FAQ*, for calculating time, space, money and etc.

6. Dealing with Less Structured Documents

In the web, a lot of information in *HTML* documents is expressed in *NL*, semi-structured *NL* or tables. Information extraction for these documents requires a variety of technique [18]. There have been some work done in this area. We proposed [19] to apply information tagging for web pages to assist information extraction. Our information tagging employs an *HTML* parser together with an *NL* semantic tagger. Our semantic tags, more general than part-of-speech tags used in linguistics, give the underlying semantic meaning of part of texts (usually a phrase) such as people, events, time, places and things. Basically, the semantic tags for people, time, places and things are used to create event tags, which are the most important information.

Take the travel agent as an example. Consider a resort area that has several hotels nearby. Some major hotels have beautiful web pages with friendly *FNL* query boxes. Some medium size hotels may have an *HTML* table summarizing their room rates and locations. But the smaller hotels might only have a plain text description. For a major hotel we can associate a regular database agent. For those pages with *HTML* tables we shall analyze the information in those tables through information tagging and association. Finally, for hotels with plain text description the relevant information can be extracted through tagging based purely on *NL* understanding.

As described above, there is a great deal of natural language processing required for the agents. We shall assign a database agent for each knowledge domain

specializing in extracting relevant information. This is no easy task in general. However, in our approach, we only have to worry about matching those FAQ listed in handbooks rather than dealing with those worst case situations in linguistics. For each such *FAQ*, we want to have a high probability mapping algorithm and a natural language script for dispatching jobs to its subagents. Those manually created scripts need to be interpretable by *NIA* and converted into actions. Thus, *NIA* steadily augment its collection of *FAQ* to meet the users' demand.

7. Examples of Agents in NIA

Based on the above agent architecture, we have constructed several prototype agents. They are briefly introduced below. More complete versions will be released later on. The event formation and mapping in most of these agent systems are based on the arithmetic agent [20] we have constructed in 1996, called *EMMA* (elementary school mathematics agent). *EMMA* can read an elementary school mathematical word problem, solve it and explain the solution procedure in natural language. With some basic inference capability, *EMMA* can resolve two main problems in natural language (in the elementary school math. domain), ellipsis and anaphora.

Because of the intensive domain knowledge required for these agents, each agent system will specialize in certain predefined domains. However, their handbooks can be expanded whenever needed.

1. *Personal Navigating Agent (PNA)*
   PNA [18] can take a natural language search query and navigate through several web pages to locate target information in the same fashion as we locate that information through web browsing. Currently, the target information we can find includes people's affiliation, *URL*, email address, telephone and fax numbers. The information to be extracted can be easily extended to hotel rates, stock prices and interest rates. *PNA* needs to classify home pages and to construct navigation maps in order to identify a navigation course efficiently. In each target page, *PNA* extracts the required information based on semantic tags. A main advantage of using semantic tags is that our *PNA* can locate where the correct information is even after the target web pages have been modified.
2. *WWW Search Agent (WSA)*
   Our search agent will analyze a user's query by converting it into events, search the corresponding event indices based on prewritten event scripts. For those queries that cannot match with prescribed events, the agent will adopt standard keyword search. The event indices were attached to each web page in the

preprocessing stage based on certain semantic tags. For example, a query such as "How can I apply for an American visa?" will be interpreted as "Go to the nearest American Consulate web site and find out the page about visa application". Such a "semantic" search tends to have very high precision. Again, a wide variety of search *FAQ* need to be analyzed and robust mapping algorithm constructed.

3. *Travel Agent (TRA)*

   Our *TRA* will work with several related agents such as airline agents, bus agents, map agents and etc. to plan a schedule. It will send queries to *PNA* to locate or update travel information. In some cases mathematical programming will be adopted to find optimal solutions. A *TRA* needs to identify the coefficients in the constraints of the linear program. We have developed natural language scripts for a *TRA* to instruct its subagents. Similar schemes have also been adopted to construct bank agents, stock agents and commodity information agents.

8. Conclusions and Future Research

An agent society communicating with *NL* is presented in this paper. The agents' ability to manipulate *NL* is a deciding factor in the usefulness of such a society. In constructing the actual agent systems, we have sufficiently constrained the scope of the *NL* knowledge required in their tasks and it appears to be quite successful. As we emphasized before, event description is a key in constructing these *NL* agents. The capability of these agents will gradually improve as the depth of our knowledge representation model improves.

Concept understanding and *NL* understanding are treated in a similar fashion. We believe there is a good potential for our *CSM* to capture important features in human understanding.

References

[1] Huhns, M. N., and M. P. Singh, "Internet-based agents: applications and infrastructure," *IEEE Internet Computing*, vol. 1, no. 4, pp. 8-9, July/August, 1997.

[2] Rich, C., and C. L. Sidner, "COLLAGEN: When agents collaborate with people," *Readings in Agents*, M. N. Huhns and M. P. Singh (Eds.), Morgan Kaufmann Publishers, USA, pp. 117-124, 1998.

[3] Luke, S., L. Spector, D. Rager, and J. Hendler, "Ontology-based web agents," *Proceedings of First Int. Conf. On Autonomous Agents*, 1997.

[4] Li, R., and L. M. Pereira, "Knowledge-based situated agents among us : a preliminary report," *Intelligent Agent III: agent theories, architectures, and languages*, J. P. Muller, M. J. Wooldridge and N. R. Jennings (Eds.), Lecture Notes in Artificial Intelligence 1193, Springer-Verlag, pp. 375-389, 1996.

[5] Hammer, J., H. Garcia-Molina, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. Information translation, mediation, and mosaic-based browsing in the TSIMMIS system. In *Proceedings of the ACM SIGMOD Internation Conference on Management of Data*, San Jose, CA, 1995.

[6] Ambite, J.-L., Y. Arens, N. Ashish, C. Y. Chee, C.-N. Hsu, C. A. Knoblock, W.-M. Shen, and S. Tejada. *The SIMS manual: Version 1.0.* Technical Report ISI/TM-95-428, University of Southern California, Information Sciences Institute, 1995.

[7] Hsu, W. L., Yi-Shiou Chen and Yuan-Kai Wang, “A Context sensitive model for concept understanding” Proceedings of ITALLC’98, (1998), 161-169.

[8] Sycara, K., K. Decker, A. Pannu, M. Williamson and D. Zeng, "Distributed intelligent agent," *IEEE Expert*, vol. 11, no. 6, pp. 36-46, December, 1996.

[9] Kinny, D., and M. Georgeff, "Modelling and design of multi-agent systems," *Intelligent Agent III: agent theories, architectures, and languages*, J. P. Muller, M. J. Wooldridge and N. R. Jennings (Eds.), Lecture Notes in Artificial Intelligence 1193, Springer-Verlag, pp. 1-20, 1996.

[10] Castelfranchi, C., "To be or not to be an agent," *Intelligent Agent III: agent theories, architectures, and languages*, J. P. Muller, M. J. Wooldridge and N. R. Jennings (Eds.), Lecture Notes in Artificial Intelligence 1193, Springer-Verlag, pp. 37-39, 1996.

[11] Franklin, S., and A. Graesser, "Is it an agent, or just a program?: a taxonomy for autonomous agents," *Intelligent Agent III: agent theories, architectures, and languages*, J. P. Muller, M. J. Wooldridge and N. R. Jennings (Eds.), Lecture Notes in Artificial Intelligence 1193, Springer-Verlag, pp. 21-35, 1996.

[12] Finin, T., R. Fritzson, D. McKay and R. McEntire, "KQML as an agent communication language," *Proceedings of the 3rd Int. Conf. On Information and Knowledge Management*(CIKM), New York, ACM Press, 1994, also available from http://www.cs.umbc.edu/kqml/papers/.

[13] Kuokka, D. and L. Harada, "On using KQML for matchmaking," *Proceedings of the First Int. Conf. On Multiagent Systems(ICMAS-95)*, pp. 239-245, San Francisco, CA, June, 1995.

[14] Decker, K., M. Williamson and K. Sycara, "Matchmaking and broking," *Proceedings of the 2nd Int. Conf. On Multi-Agent Systems*, (ICMAS'96), pp. ???, December, 1996.

[15] Mostafa, J., S. Mukhopadhyay, W. Lam and M. Palakal, "A multilevel approach

to intelligent information filtering: model, system, and evaluation," *ACM Trans. On Information Systems*, vol. 15, no. 4, pp. 368-399, October 1997.

[16] Kuokka, D., and L. Harada, "Matchmaking for information agents," *Proceedings of the Int. Joint Conf. On Artificial Intelligence*(IJCAI '95), Montreal, Quebec, Canada, pp. 672-679, August, 1995.

[17] Minsky, M., 1975, "A Framework for Representing Knowledge", in *Psychology of Computer Vision*, P. Winston (Ed.), McGraw-ill, New York.

[18] Davies, N. J., R. Weeks and M. C. Revett, “Information agents for the world wide web,” *Software Agents and Soft Computing: Towards enhancing machine intelligence*, Springer, pp. 81-99, 1997.

[19] Wang, H. L., W. K. Shih, C. N. Hsu, Y. S. Chen, Y. L. Wang and W. L. Hsu, “Personal Navigating Agent,” submitted to AGENT’99, (1998).

[20] Hsu, W. L., “Elementary school Math. tutoring agent”, *Proceedings of Agent Technology Workshop*, 1997.