Multi-Agent-Based Adaptive AV Interface

Tarek El-Basuny[1] and Makoto Amamiya[2]
[1]Dept. of Information and Computer Science, King Fahd University of Petroleum and Minerals, SA
[2] Department of Intelligent Systems, Kyushu University, Japan

Abstract: *In order to build adaptive interfaces, we need adaptive interaction and dialogue handling methods. We have presented an advanced model for interaction and dialogue management to support adaptive natural language Audio Visual (AV) interface. Our multi-agent-based Natural Language (NL) interface is a software application environment that breaks up NL interpretation into a community of collaborating, learning agents. It allows users to control AV appliances in NL, rather than using remote control devices. It learns and remembers the way a user does things, customizes its performance to match the user's behavior. This paper shows at first the basic feature of AV agent system, and then reports the implementation and experimentation for Japanese version, which connect multi-agent-based NL interface with actual appliances and Sound Recognition Engine (SRE). By the experiment, our system works well; it provides an impressive degree of accuracy, measured as the percentage of requests that translate into the operation intended by the user. But we consider that the miss recognition of SRE should be absorbed more by the multi-agent system to make this system easier and comfortable to the users. Therefore, we propose an absorption theory by learning the habits of the SRE and the users, and then absorb the recognition errors of SRE after a time of training.*

Keywords: *Agent-oriented-programming, NL interface, learning, adaptation.*

Received March 8, 2005; accepted May 14, 2005

1. Introduction

Most human-AV appliance remote control devices being used today are complicated and difficult for a lot of users. This is due mostly to the growing number of features the control devices should provide. Recently, agent technologies have attracted a lot of interest in both academe and industry as an emerging programming paradigm [9, 12, 14, 15, 16, 17, 19]. Our goal is to design and implement a multi-agent based software methodology that allows people to communicate with AV appliance in the same way as with another person like "*AV power on*", or "*Terebi wo tsukete*", or "إفتح ألتلفاز". Agent-oriented programming paradigm yet to be appreciated expands our experience in implementing many different agent-based applications [2, 3, 4, 5, 18]. Our multi-agent system makes it easy for every person to interact with the AV appliances, releasing them from interfaces. It allows users to control electronic devices and AV appliance in plain voice, rather than using remote control devices. In our system, we designed each agent's learning and communication modules so that a software designer can concern only with the design of the agent's assignment. This means the designer assigns a process to each agent [1, 6, 7, 8, 11, 12] and states the agents with which this agent will be communicating at the beginning of its assignment. In our system, agents learn the needs and habits of a particular user. Each agent remains responsible for the specific restricted domain to which it was originally assigned, but at the same time improves its performance in that domain by learning. At its simplest level, our proposal greatly improves ease-of-use, while reducing a new user's learning time. Our multi-agent system learns and remembers the way a user does things, customizing its performance to match the user's behavior. This paper reports the implementation and experimentation of Japanese AV agent system, with actual appliances. As the history of AV agent, English version was developed at first [7] in which actual appliances are not used but keyboard-input as input, device-representation window as output.

The rest of this paper is structured as follows. Section 2 gives the basic theory of AV agent based on the English version [7]. Section 4 shows the extension into the AV Japanese version, while section 5 describes the experimental results. Section 6 discusses the future improvement of AV agent based on the result of the experiments, its main topic, how to implement the reinforcement learning technique in the system. Section 7 lists the features of using our system as an AV interface. Finally, section 8 gives the conclusions and the future work.

2. Basic Theory of AV Agent System

The most common approach to natural interaction is to start at the top with the components of the language and analyze text in terms of the rules of the language [3, 10, 11, 12]. A good top down natural interaction system requires a large dictionary, a thesaurus, and a full understanding of the ways people create sentences. It also requires a great deal of general cultural knowledge, including the relationship between the meaning of idiomatic expressions and the literal meanings of the words themselves. An alternative is to consider language from the bottom up, starting with the tasks to be performed and considering all the different ways potential users might phrase their wishes. In our multi-agent-based natural interface, a network of agents has been employed. In this network, agents merely have to recognize the subset that is relevant to their particular responsibilities. This provides an impressive degree of accuracy. It can also be extended to support additional vocabulary and additional application functions.

2.1. Agent AV

The goal of AV agent system is to integrate AV appliances and control them in voice with simple implementations. Figure 1 shows the structure of the system, we mainly developed. The proposed system enables end users to control devices directly through intuitive words or phrases, thereby saving time and avoiding frustration. The basis of the presented technology is an engineering methodology that breaks up complex software into a community of simpler, collaborating, message-driven components known as agents [6, 7]. The agent gets the string as input which is already separated into words by SRE, and sends the control commands to the AV appliances instead of the remote-control device. Agents are organized to address a given target domain. Since no single agent has the ability to solve the entire problem, they attack it as a group, sometimes competing, often cooperating, but always moving forward collectively towards a solution. The approach is non-centralized, yet modular, agent subgroups are specialized in different aspects of the problem, and work in parallel on pending tasks. Ambiguities may arise and are resolved by other agents or through dialog with the user.

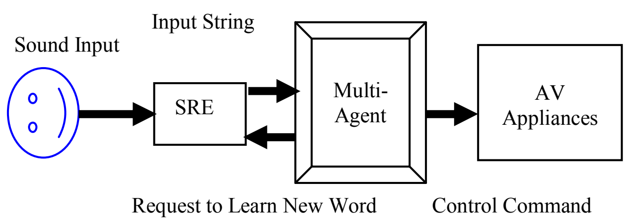

Figure 1. Structure of AV agent system.

2.2. Speech Recognition by Multi-Agent System

In multi-agent part, there are a lot of agents and they communicate with each other. Figure 2 shows the structure of agents. One agent takes a role of one appliance, an operation, and so on. Once an input sentence is thrown in the top agent named input agent, the sentence is sent to each agent in turn and each of them do its interpretation which is based on keyword matching between the input command and the Semantic Policies (SP) of the agent. The SP is used to decide whether or not the input commands belong to that particular agent. Because these interpretations are done mainly by keyword matching, this system essentially has the advantage of language-independence. Another advantage of the system is that we can add or remove appliances dynamically. This is simply done by adding or removing the agent and modifying the links of the agent.

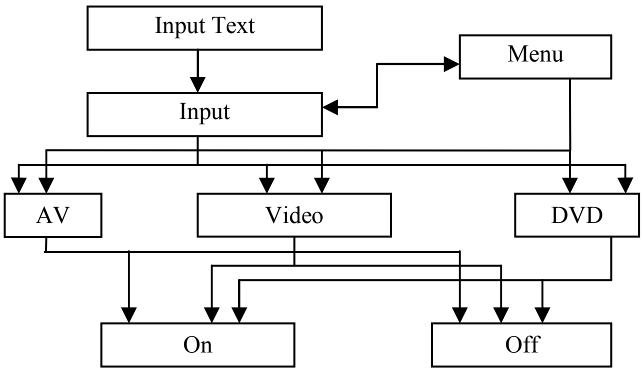

Figure 2. Example of the structure of agents in multi-agent part of agent AV.

3. Agents Interaction Phases

Processing of the input is done in three main phases. An *interpretation phase*, in which the agent, or agents responsible for actuating an input are located, a *delegation phase*, in which the processes that have been located are called, and an *Actuation phase*, in which agents collaborate to create the appropriate output.

3.1. Interpretation Phase

The input agent which is the entry point of the system initiates the interpretation phase for that input string, generating unique query command IDs for new input. This does not mean that other agents do not query input agents. In the interpretation phase, each agent, upon receiving input with an "*Is-This-Yours?*" performative, attempts to interpret the input by itself. If interpretation is successful, the agent will report claims using the "*It-Is-Mine*" performative. On the other hand, if an agent cannot interpret the input as its own, before reporting failure, it must check with other down-chain agents. If all down-chain agents report "*Not-Mine*", this agent will also report "*Not-Mine*" to its requesting agent. If at least one down-chain agent is able to interpret the input successfully and reports back with "*It-Is-Mine*", the parent agent will also report success. It follows that agents that have no down-chain agents to query may report *"Not-Mine"* upon failure to find a SP that applies to the contents of the query message they have received.

To prevent agents from repeatedly processing the same queries in a cycle, each agent keeps track of queries it has processed and will reply *"Not-Mine"* to any query it has already responded to and has no new claims for it. As said before, an agent that does not have a suitable interpretation for input contents of a message sent to it with the "*Is-This-Yours?*" performative will propagate this message to its down-chain agents. By suitable interpretation, we imply that in cases where the SP uses a small part of the whole input as its decision making focus, the agent may decide to query down-chain agents on the remainder of the input anyway, so as to make more accurate claims.

The SPs comprised of a set of rules used to decide to return a claim that a piece of the input belongs to that particular agent. The interpretation criterion may be the message content but is not limited to it. Process history, probabilities and outside information (e. g., interaction with other agents) are examples of some of the other parameters that may be used by the SP. The software designer is responsible for providing each agent with its SP. Note that policies do not determine whether a particular input does not belong to the agent. The policies determines whether an input does not fall into the scope of responsibilities of an agent, as well as whether it does, amounts to modeling the world ($W = P \cup \sim P$) and undermines the distributed nature of agents. Therefore, the application of SP to the input results either a successful interpretation or a "don't-know" state [2, 3, 4, 6, 7, 8].

The agent uses fuzzy-*matching* to find the closest match when making string comparisons for SP (see section 4). The fuzzy matching algorithm will return a confidence factor that shows the degree of similarity between what was encountered and what was expected in a semantic rule. Semantic rules contain weights to adjust the standard confidence factors generated by the matching process. Agents can add or modify their semantic rules. Any modification or additions to the semantic rules need to be marked and time-stamped so as to allow for rollbacks. At the same time, existing semantic rules may be rewarded based on their success in resolving ambiguities. This allows for conflicting semantic rules to co-exist in the same agent. The reward system will also provide for a means to optimize agent behavior using evolutionary optimization methods.

3.2. Delegation Phase

In the delegation phase, the "*This-Is-Yours*" performative is used to select from down-chain agents that have made claims to the input at the interpretation phase. In other words, the delegation phase selects the best path or paths to down-chain claiming agents in order to initiate the actual processing of input. Agents receiving a "*This-Is-Yours*" request may reinterpret the delegated input, or they may use pre-stored interpretation or down-chain query results in turn to delegate further down-chain.

3.3. Actuation Phase

In this phase selected agents perform the actual processing with respect to the input. This processing may involve further collaboration between certain agents. Special purpose performatives are used to coordinate this processing.

4. Technologies Implemented for Japanese Version

Thanks for the characteristics of this system, i. e., for that of language-independence, grammatical difference between Japanese and English does not become a big problem to develop the Japanese version. Although the interface between multi-agent system and SRE took a big change, new technologies described have been implemented.

4.1. Confidence Factor

In the phase of interpretation by each agent, keyword matching should be done not exactly but fuzzily and that is what actually happens in the Japanese version, where agents interpret sentences by fuzzy matching and estimate how likely the sentence belongs to them. The estimated value is called confidence factor. This is necessary to absorb the difference between the SRE output and the agent's SP.

The idea is that traditional string comparison returns only Boolean values true or falls as to its equivalence. Given two strings, for example, "channel" and "channel", "ESPN" and "ISPN" a common misspelling, string comparing would return false, while the casual observer can fairly and accurately discern the intended spelling. We are using fuzzy logic for getting the certainty of the similarity between the input string and the SP of each agent of the AV system.

4.2. Demonstrating the Interpretation Technique

We assume that string 1 represents the SP and string 2 represents the part of the input command concerning an agent. The method for comparing the input strings with the SP begins with the first element, and continues until the end of the string is reached, or a difference is encountered. For a given string of length *N*, one can perform boolean matching on each element on a *one-to-one* basis with the second string, returning 1 for a match, and 0 for a mismatch. Returning to the above example of *channel* & *channal*, record & recard, and so on.

Example 1: In case of mismatching a character with one in the same position:

String 1 C H A N N E L
String 2 C H A N N A L

Results 1 1 1 1 1 0 1

Summing up the individual results yields: 1 + 1 + 1 + 1 + 1 + 0 + 1 = 6, divided by N (the length of the string) yields the 85% of the matching character rate.

Example 2: When two characters are swapped:

String 1 C H A N N E L
String 2 C H A N E N L

Results 1 1 1 1 0 0 1 = .71

We can improve this by comparing characters relative to their current position. In the next example, we will demonstrate this by not only examining the character at location N, but also at $N - 1$ and $N + 1$ if the first comparison fails. If there is a match, we can't just assign a true value, but some degree of truthfulness. We assume that we can give a 25% penalty on matches made at an offset of +1 / -1. This increases the accuracy of the comparison considerably.

Example 3: The above example with different position would yield:

String 1 C H A N N E L
String 2 C H A N E N L

Results 1 1 1 1 .75 .75 1 = .92

This is useful in case of user says SEPN instead of ESPN, HNK instead of NHK and so on. Finally, this method is fruitful also in case of misunderstanding of the phonetic matching process by the Speech Recognition Engine (SRE). If the output of the SRE has some conflictions according to the incorrect pronunciation from the users, then there is a possibility that some characters will be given to the text input agent while it is not the correct one. For instance, in the comparison of “Channel” and “Channal”, “play” and “pray” we can easily see that the intended spelling was “Channel” and "play” that a simple error based on similar sounds in English language or Japanese language was made. The idea here is to estimate how close a letter sounds to another. For example, a “c” may sound like a “s”, or a “s” may sound like a “z”. In case of Japanese language, a "chi" may sound like a “shi”, or a “ku” may sound like a “ko”. Ideally these values would be derived from statistical analysis of spelling errors. In this example we have assigned a value of .66 for the phonetically match between an A and an E, yielding:

String 1 C H A N N E L
String 2 C H A N N A L

Results 1 1 1 1 1 .66 1 = .95

4.3. Using Confidence Factor with the AV Agent System

A new variable named Confidence Factor (CF) will be added to the SP of each agent. The CF variable can be any number between 0~1. Upon initialization the CF of each policy would be 1. But this could change upon learning or upon reading a CF from the SP file of the agent. The format for entering the CF in the SP of each agent is (Semantic Policy: CF). Matching an input string with the SP is done by a function called check string, this function explained above using fuzzy logic and it returns a Str_Match CF. The returned CF = Str_Match CF * CF. If an agent receives two or more inclusive claims on the same input then it should resolve the ambiguity using the CF of each claim. Here is the implemented algorithm for comparing the CF of each claim:

1. If one or more claims with a CF equal to 1 exist, then the system resolved the ambiguity by interacting with the user.
2. If all claims are below 1, then the system selects the highest claim if and only if the claim is ψ points bigger than CF of all other claims in the ambiguous claims set (ψ is a constant, for instance we set ψ = 20).
3. If there are two or more claims within the 25% range of each other, then ambiguity remains and the user is asked to select from the top ω claims (ω is a constant, for instance we set ω = 6).

4.4. Dealing with Garbage

Garbage is the word, which is unreliable because the likelihood of SRE is not enough. In our system, when some output words of SRE are not reliable, SRE puts the special symbol “<G>” in front of such words so that the agents can easily distinguish garbage from reliable words (like “terebi <G>wo tsukete” where “wo” is unreliable). When an agent interprets the sentence with garbage and the garbage belongs to the agent, the agent lower estimates its confidence factor. Sometimes garbage is the word, which is not yet registered in SRE. Most of the case, the word is simply ignored in the interpretation, but sometimes the garbage stands for an important word to the interpretation. (For example, a person may call a television “movie”, but since the word is not registered, it becomes garbage). In this case, the appropriate agent in the agent network should learn the word, and then let SRE knows the word is being learned as a new policy word.

5. Experimental Results and Evaluation

We will explain our implementation on a Japanese AV agent system. The idea is to distribute the NL processing over nodes that represent different levels of a hyper structure covering the functionality space. The NL interpretation is done through a series of claims and delegations carried out by the agents.

5.1. Experimental Environment

As shown in Figure 3, the experimental environment consists of Microphone: For the input voice, PC: In which SRE program and multi-agent programs are running, SRE: GrapHvite by Entropic co, Multi-agent part: On JDK 1.3.2, Infrared ray's remote-control device: Connected with the PC by RS-232C and AV Appliances: Television, VCR, DVD player, and satellite tuner (for BS/CS).

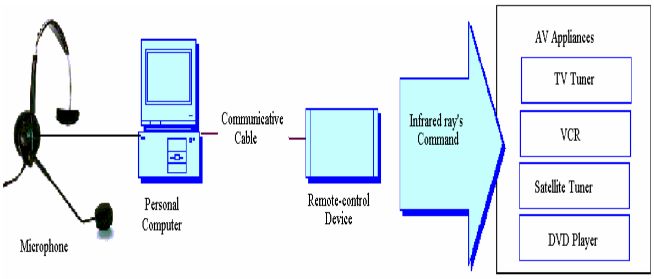

Figure3. Experimental environment.

5.2. Experimental Method

In the experiments, thirty people tested the system, they spoke predefined 114 sentences; some of them are shown below:

- "AV no channeru wo 37 ni" (change the channel to 37).
- "5 ni shite" (make something to 5) is ambiguous and the multi-agent part should solve the ambiguity using history and context. If there still some ambiguities, it then asks the user through interface menu.
- "channeru wo NHK ni shite" (change channel to NHK). When neither SRE nor any agents know the word NHK, then the multi-agent part should ask the user, what does NHK mean? And an appropriate agent learns the response of the user.
- "yoru 10 ji kara 12 ji made 4 channeru rokuga shite" (record programs of channel 4 from 10 to 12 in the evening). Even such a pretty long sentence can be interpreted correctly in our system if the SRE recognized well.

Various kinds of people were chosen of age from 20s to 50s. They were 14 Japanese/ foreigner males and 16 Japanese/ foreigner females.

5.3. Recognition Rate and Execution Rate

There are 2 parameters to judge the capacity of this system:

1. *SRE recognition rate*: The rate that SRE can output the same sentence as the user spoke.
2. *Execution rate*: The rate the system interprets the command and executes the correct action.

By these definitions, the difference between these two parameters shows the absorption rate of SRE's recognition error by the agent network. Table 1 shows both SRE recognition rate and execution rate arranged by the subjects' gender and age. By this table we can find that execution rate exceeds the SRE recognition rate in most of the cases. It proves that agents in the multi-agent part have the ability to absorb the miss recognition of SRE. As effects of age and gender, we didn't find any prominent differences.

- Result (%): The rate the system executed correctly without menu.
- SRE (%): The rate SRE output the same string as the user said.
- Abs. (%) = Result – SRE: How many execution multi-agents recover.
- Menu (%): Menu displayed to ask the user, and executed correctly.
- NG (%) =100 - Menu-Result: The rate the system worked ill.

Table 1. Recognition rates arranged by age and sex.

Age	Sex	SRE %	Result %	Abs. %	Menu %	NG %
20s	F	49.23	60.93	11.70	16.56	22.52
20s	M	50.00	67.26	17.26	19.03	13.72
30s	F	52.21	63.42	11.21	16.22	20.35
30s	M	57.24	67.14	9.89	19.43	13.43
40s	F	52.65	62.39	9.73	17.70	19.91
40s	M	52.72	60.24	7.51	19.00	20.77
50s	F	54.24	64.66	10.42	20.32	15.02
50s	M	47.35	54.41	7.06	20.29	25.29

5.4. The Capability of Multi-Agent Part

Table 2 shows the relation between SRE recognition and execution rate. The data shows there are some cases that execution fails in spite of SRE recognition success. By the experiment, we found that, there were some cases that multi-agent part fails to execute in spite of SRE recognized them correctly. Most of the cases occurred when the agents were not able to process their history well. For example, when the user said "AV off" after "AV sound on", then the agents guess the input "AV off" means "AV sound off" from the history, and executed AV mute instead of AV power off. History processing is important to deal with ambiguity, but the result shows that the processing history of current system must be improved.

Table 2. Relation between SRE recognition and execution.

%	SRE OK	SRE NG	Total
Result OK	49.2	13.4	62.6
Otherwise	3.3	34.1	37.4
Total	52.5	47.5	100.0

- SRE OK (%): The rate SRE output the same string as the user said.
- SRE NG (%): The rate SRE output different string from user input.
- Result OK (%): The rate the system executed correctly without menu.
- Otherwise (%): The rate the system worked ill or worked well but with menu.

5.5. Consideration

The experimental results show our system works well, but it also has the room to improve.

1. Even if the recognition rate of SRE is not so good, multi-agent part executes well by learning.
2. Improve history processing; to reduce this kind of miss execution, learn the habit of users.

To achieve them, we consider implementing reinforcement learning because SRE tends to output the characteristic recognition errors, and users also tend to say the characteristic words by each user. So, we can absorb the errors by learning such SRE dependent and user dependent features.

6. Necessity of Reinforcement Learning

The goal of learning in AV agent system is to reduce the occurrence of the ambiguity and to adapt the behavior of the system. The result of the experiments shows that, most of execution failures are because of miss recognition of SRE. Since we cannot manipulate the SRE, we have to absorb the miss recognition by the multi-agent system to enhance the system performance. To achieve this, we implemented the fuzzy matching propose and on going to implement reinforcement learning to the system. Such a system learns the habits of the SRE and the users, and then absorbs the recognition errors of SRE after a time of training. This theory may be applicable to any symbolic recognition by multi-agent system whose input contains any noisy. Next section describes our proposed theory, but we note that it is currently under implementation.

Since the keywords are not all equally important for content representation of the SP of each agent, weights are assigned to the keywords in proportion to their presumed importance and position [4, 18]. The SP contains the keyword, the weight, the reward factor, the agents evolved, co-occurred words. We use the keyword co-occurrence as follows: If a keyword α is significantly correlated with another word β, then (α, β) is considered a co-occurrence pair. When α occurs in the command, it triggers β. We expect the keyword β to appear somewhere after α with some confidence. For instance, the word *on/off* with high confidence should come as co-occurrence word of *Power/AV/VCR* and so on. In addition, it is important to keep the current state of the system, the history of the user and that is why we are creating a User Profile (UP). The representation of a UP is similar to that of the SP. A UP consists of the command, the action agents, and the path history.

7. Features of the AV Agent System

- *Natural expression*: The user can express his/her intentions as freely and naturally as possible.
- *Ease of change and upgrade*: The application designer is easily being able to upgrade or change the system with minimum compromise to the adaptation the system has made to users.
- *Language independence*: The semantic modeling minimizes the development required to move from one language to the other and even have support for more than one language at a time or a mix of languages.
- *Smart interactions and dialogues*: Interacting with the user can be used during coordination as a means to resolve contradicting claims from agents.
- *Unpredicted input*: Ungrammatical, unpredicted (garbage) input will be accepted by the system. The system downgrades gracefully to a dialogue system to get the user to the desired functionality.
- *Runtime changes to lingual scope*: Addition or removal of agent sub-networks as the system is running is possible and changes the lingual and contextual scope of the application.

8. Conclusions and Future Work

This paper discussed a multi-agent-based approach to build a NL interfaces to a home theater system and it is being implemented as a web browsing interface also. The current version of our agent system has been implemented in Java because of its unique portability, multi-platform execution capabilities and multi-threading features. In this paper we introduced our system as a novel NL voice interactive interface. Our proposal is flexible, primarily because there is no rigid predetermination of valid input. It is modular providing for easier revision, extension and development. In our multi-agent system, agents can be re-used inside an application or in other software. The independent nature of our agents provides for an inherently parallel architecture. Agents can run and communicate over a network of heterogeneous hosts. Addition of new agents is possible and therefore incremental development and evaluation is possible. The built-in learning and ambiguity resolution features make our agent system be more intelligent software architecture. As a future work, we want to make a precise evaluation and solve the negative comments of the experimenters. We have several research plans on this architecture to implement a reinforcement learning techniques, adapt the SP overtime and then develop the NL Arabic AV agent system.

Acknowledgments

We would like to thank Kyushu University and King Fahd University of Petroleum and Minerals for providing of the computing facilities. A part of this research work was partially supported by the Information-technology Promotion Agency of Japan.

References

[1] Glass J. and Hazen T. J., "Telephone-Based Conversational Speech Recognition in the Jupiter Domain," *in Proceedings of International Conference on Spoken Language Processing (ICSLP'98)*, Sydney, Australia, 1998.

[2] Helmy T., Amamiya S., and Amamiya M., "Collaborative Agents with Automated Learning and Adapting for Personalized Web Searching," *in Proceedings of the 13th International Conference on Innovative Applications of Artificial Intelligence (IAAI/IJCAI-2001)*, Seattle, USA, pp. 65-72, August 2001.

[3] Helmy T., Amamiya S. and Amamiya M., "Pinpoint Web Searching and User modeling on the Collaborative Agents," *in Proceedings of the 2nd International Conference on Electronic Commerce and Web Technologies (EC-WEB 2001)*, Germany, pp. 305-314, September 2001.

[4] Helmy T., Amamiya S., Mine T., and Amamiya M., "A New Approach of the Collaborative User Interface Agents," *in Proceedings of IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'03)*, Halifax, Canada, pp. 147-153, October 2003.

[5] Helmy T., Amamiya S., Mine T., and Amamiya M., "Agents Coordination-Based Web Infrastructure for Personalized Web Searching," *in Proceedings of the International Conference on Artificial Intelligence (ICAI'2001)*, USA, pp. 97-103, June 2001.

[6] Hodjat B. and Amamiya M., "Usability Evaluation and Interface Design," in Smith M. J., Salvendy G., Harris D., and Koubek R. J. (Eds), *Cognitive Engineering, Intelligent Agents and Virtual Reality*, Lawrance Erlbaum Associates Publishers, London, pp. 405-409, vol. 1, 2001.

[7] Hodjat B. and Amamiya M., "An Agent-Oriented Architecture for Natural Language Interfaces," *in Proceedings of the 6th World Multi-Conference on Systemic, Cybernetics and Informatics (SCI02)*, pp. 128-133, 2002.

[8] Hodjat B., Savoie B., and Amamiya M., "Adaptive Agent Oriented Software Architecture," *in Proceedings of Pacific Rim International Conference on Artificial Intelligence (PRICAI'98)*, November 1998.

[9] Hofmann M. O., McGovern A., and Whitebread K. R., "Mobile Agents on the Digital Battlefield," *in Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*, 1998.

[10] Kautz H. and Selman B., "A New Approach to the Application of Theorem Proving to Problem Solving," *in Proceedings of AIPS98 Workshop on Planning as Combinatorial Search*, pp. 58-60, 1998.

[11] Lamel L., Bennacef S. K., Rosset S., Devillers L., Foukia S., Gangolf J. J., and Gauvain J. L., "The LIMSI RailTel System: Field Trials of a Telephone Service for Rail Travel Information," *Speech Communication*, vol. 23, no. 1-2, pp. 67-82, October 1997.

[12] Larsson S., Cooper R., and Ericsson S., "Menu2dialog," *in Proceedings of IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, San Diego, pp. 41-45, 2001.

[13] Levin E., Pieraccini R., and Eckert W., "Using Markov Decision Process for Learning Dialogue Strategies," *in Proceedings of ICASSP'98*, Seattle, WA, USA, May 1998.

[14] McDermott D., "PDDL: The Planning Domain Definition Language," *AIPS*, 1998.

[15] Nichols J., Myers B. A., Higgins M., Hughes J., Harris T. K., Rosenfeld R., and Pignol M., "Generating Remote Control Interfaces for Complex Appliances," *CHI Letters: ACM Symposium on User Interface Software and Technolog*y, 2002.

[16] Rudnicky, A. and Xu W., "An Agenda-Based Dialog Management Architecture for Spoken Language Systems," *in Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop*, pp. I–337, 1999.

[17] Seneff S., Hurley E., Lau R., Pao C., Schmid P., and Zue V., "Galaxy-II: Reference Architecture for Conversational System Development," *in Proceedings of ICSLP'98*, Sydney, Australia, November 1998.

[18] Tarek El-Basuny, "A Ubiquitous Approach for Next Generation Information Retrieval System," *in Proceedings of the IEEE/CS International Conference on Information and Computer Science (ICICS'2004)*, KFUPM, Saudi Arabia, pp. 501-513, November 2004.

[19] Turunen M. and Hakulinen, J., "Jaspis: A Framework for Multilingual Adaptive Speech

Applications," *in Proceedings of 6th International Conference of Spoken Language Processing*, 2000.

Tarek El-Basuny is an assistant professor in the Department of Information and Computer Science, at King Fahd University of Petroleum and Minerals. He got a PhD from Kyushu University, Japan in 2002. His research interests include operating systems, multi-agent systems, artificial intelligence, and personalized web services.

Makoto Amamiya is a professor in the Department of Intelligent Systems, at Kyushu University. He is the president of the Information and System Society, Institute of Electronics, Information and Communication Engineers (IEICE), Japan. He got a PhD from Kyushu University in 1978. His research interests include parallel processing systems, processor design, operating system, and multi-agent systems from kernel to application.