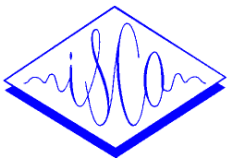

6th International Conference on Spoken Language Processing (ICSLP 2000)
Beijing, China
October 16-20, 2000

# DESIGNING A DOMAIN INDEPENDENT PLATFORM OF SPOKEN DIALOGUE SYSTEM

*Kazumi Aoyama, Izumi Hirano, Hideaki Kikuchi and Katsuhiko Shirai*

Department of Information and Computer Science, Waseda University

3-4-1,Okubo,Shinjuku-ku, Tokyo 169-8555 Japan

Email:kazumi@shirai.info.waseda.ac.jp

## ABSTRACT

Our purpose is to construct a domain independent platform for task-oriented spoken dialogue system. It is expected to generalize architecture of spoken dialogue system for making it easy to develop a system that can support and act to solve problems for user by using speech input and output. In this research, we propose a method of describing hierarchically internal knowledge for solving problems and one for controlling dialogue as rules. A system designer only should describe two types of rules those are called problem-solving rules and action-management rules. And we designed and implemented the platform and evaluated how well the rules can express knowledge.

## 1. INTRODUCTION

The purpose of this study is to construct a domain independent platform for task-oriented spoken dialogue system. Usually, in implementing a spoken dialogue system that can support and act to solve problems for user, internal knowledge for spoken dialogue is not modularized and it is designed individually for each task. So we cannot reuse the knowledge for controlling dialogues. This problem causes that it is difficult to describe knowledge for solution of a problem and one for controlling dialogues separately. Therefore, it is expected to generalize internal knowledge and architecture of spoken dialogue system for making it easy to construct a system.

To solve this problem, some of studies are being investigated previously. Domain independent platform of spoken dialogue interface for information query has been developed at Kyoto Univ. [1]. This platform makes system developers to construct information query dialogue system easily, because this dialogue system presents limited acceptable utterance patterns. Automatic Dialogue Generator (ADG) has been developed at Bell Lab [2]. ADG is the platform that automatically builds dialogues approximated by a finite state specification. The individual states are generated in a consistent and uniform manner that is portable to other domains.

If we describe knowledge of various dialogue strategies out of the dialogue system, we need to handle a large number of parameters, so we have to describe complex rules. But if we describe knowledge of controlling dialogue easily, we cannot control a variety of dialogue strategies. In this research, we focus on the trade-off problem between variety of dialogue strategies and easiness to describe rules.

In this paper, we propose a method of describing hierarchically knowledge to solve problems and one for controlling dialogue as rules. Concretely, we designed the platform of spoken dialogue system using hierarchical levels for solving problems. These levels are problem-solving-level and actual-system's-behavior-level. The rules of problem-solving-level is called problem-solving rules that are described as production rules. And the rules of actual-system's-behavior-level are called action-management rules that are described as finite-state automaton.

In Section 2, we explain an outline of this domain independent platform of spoken dialogue system. In Section 3, we explain methodology of Action management unit, which is a part of our platform. And we propose regulation of problem-solving rules and action-management rules. In Section 4, we tried to describe these rules along this method using actual dialogue data, information retrieval task. And we evaluated the ability of describing dialogue by using this method. Finally, in Section 5, we conclude this research and future works.

## 2. OUTLINE OF THE PLATFORM

### 2.1. Design policy of the platform

Spoken dialogue systems should manage some kinds of internal knowledge for controlling dialogues. The knowledge should be independent on the platform to construct spoken dialogue systems for various tasks. But there is the above-mentioned trade-off problem. So we propose the description of dialogue strategies divided into two steps to solve this problem. One step is the problem-solving-level, which decides abstract action-plan. And the other step is the actual-system's-behavior-level, which decides concrete system's action such as system's utterance. Concretely, rules of problem-solving-level called problem-solving rules, which is an upper level described as production rules. And rules of actual-system's-behavior-level called action-management rules, which is a lower level described as finite-state automaton. In this way, we can realize to control variety of dialogue and easiness to describe rules of dialogue.

### 2.2. Structure of the platform

The domain independent platform of spoken dialogue system which we designed and implemented is based on slot-filling task. So it can apply to various types of task domains, for example, telephone shopping, database retrieval, hotel reservation, etc. The structure of this platform is shown in the Fig.1.

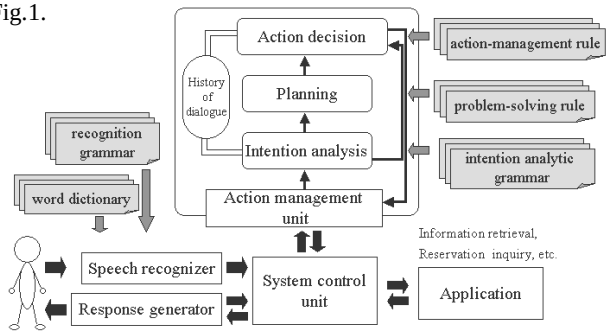


**Fig.1:** Structure of this platform of spoken dialogue system

Action management unit reads the two-steps rules (action-management rules and problem-solving rules). And it solves the problem by using this knowledge out of the system as shown in this figure. As for a movement in action management unit and regulation of these rules, we explain in detail in the next section. This spoken dialogue system is composed by Application such as the data base retrieval corresponding to a task domain, Speech recognizer, System control unit, Action management unit and Response generator. Also, the system was implemented as the multi-agent system for a various control of modules.

## 3. METHODOLOGY OF ACTION MANAGEMENT UNIT

### 3.1. Management process of Action management unit

The flow chart of Action management unit is shown in the Fig.2. First, Action management unit reads problem-solving rule and action-management rules in initialization as shown in this figure. Usually, the processes after the intention analysis is executed for every message received from the System control unit.

When Action management unit receives a message that indicates “A result of speech recognition”, the intention of user’s speech is analyzed by using the Intention analytic grammar. After that, the need of planning is judged. When Action management unit receives a message that indicates “A result of the application”, that result goes past the Intention analysis part, and the need of planning is judged. When Action management unit receives a message that indicates “A pause length exceeded the threshold”, Action decision part decides concrete system’s behavior directory. When Action management unit judged no need of planning, Action decision part decides concrete system’s behavior directory, too.

When planning is necessary, the Planning part reasons backward by using problem-solving rules. First, a goal to solve problem is established corresponding to the current state and the precondition to attain a goal is evaluated. If a precondition is satisfied, that goal is attained. If a precondition isn't satisfied, the rule which has precondition part in the conclusion part is looked up to satisfy the precondition. And the precondition part of that rule is evaluated again. Repeating these steps, rules are backtracked until a goal is finally attained. When it is necessary to execute action plan to attain a goal to solve problem while a condition is evaluated, the part which action plan is decided (Planning part) chooses that action plan and Decision on action part decides concrete system's behavior.

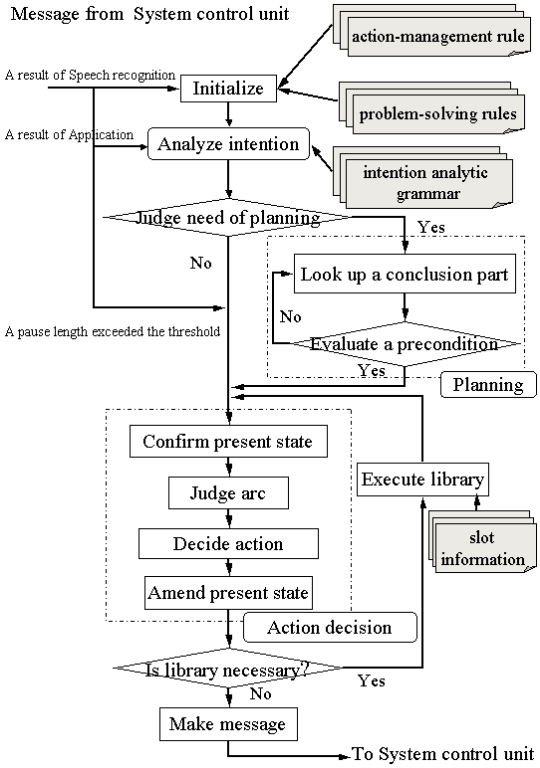


**Fig.2:** Flow chart for Action management unit

In the Action decision part, which decides the concrete system’s behavior corresponding to the action plan by using action-management rules. When plan is finished in Action decision part, next action plan is decided as the next planning by Planning part.

In this way, Action management unit decides the next system's behavior by using problem-solving rules and Action-management rules in accordance with the present state of the spoken dialogue system, the input message, and so on. Then, an output message is made at last.

### 3.2. Description of rules

The rules which a system designer should describe in the Action management unit are problem-solving rules, action-management rules, Intention analytic grammar and Slot information as shown Fig.2. Problem-solving rules are described using production rules to solve the problem in accordance with the goal of the dialogue. Action-management

rules are described as finite-state automaton for every abstract action plan decided with problem- solving rules. Problem-solving rules and action-management rules are described in SGML-like format. Intention analytic grammar is described in template. Slot information is the description of the sets of attribute-value pair prepared in advance. In this section, we explain the description form of two main rules, problem-solving rules and action-management rules.

### 3.2.1. Problem-solving rules

Problem-solving rules are the description to decide a goal for a solution of a problem. A goal for a solution of a problem sometimes changes dynamically. For example, we can think about the next two goals in case of an academic paper retrieval task.

1. A user wants to look for the specific paper
2. A user wants to collect many paper related to the keyword.

When more than one such goal exists at the same time and when they change in dialogue, planning is necessary to decide dialogue strategy corresponding to the goal for a solution of a problem. Problem-solving rules are the rules to decide an abstract action plan corresponding to the goal for a solution of a problem by reasoning backward. The description form of problem-solving rules is as the following.

```
<rule name = "Rule Name">
<cond "Precondition Part">
<conc "Conclusion Part">
</rule>
```

**Description form of problem solving rule**

Precondition part and conclusion part consist of procedure names and arguments; they can have some values by 'or/and'. When a precondition part has some rules using 'and', each procedure is decided to carry out in order. The description example of problem-solving rules that took an a paper retrieval task for instance is shown in the following (as shown Fig.3).

```
<rule id=1>
<cond "SubGoal" "Acquisition_of_the_retrieval_condition"
and "Act The_paper_indication ">
<conc "Goal Special_paper_indication">
</rule>

<rule id=2>
<cond "Act Acquisition_of_the_retrieval_condition _AND"
      and "Act The_paper_retrieval"
      and "Equal Number_of_retrieval_result 1">
<conc "SubGoal Acquisition_of_the_retrieval_condition">
</rule>

<rule id=3>
<cond "Act Retrieval_condition_focusing">
<conc "Equal Number_of_retrieval_result 1">
</rule>

<rule id=4>
<cond "Act Acquisition_of_the_retrieval_condition _OR">
      and "Act The_paper_retrieval (Retrieval_condition)"
      and "AlmostEqual Number_of_retrieval_result 10"
      and "Act The_paper_list_indication"
      <conc "Goal The_related_paper_list_indication">
</rule>

<rule id=5>
<cond "Act Add_the_retrieval_condition_OR">
<conc "AlmostEqual Number_of_retrieval_result 10" >
</rule>
```

**Fig.3:** Example of problem-solving rules (paper retrieval task)

Three blocks of the beginning in problem-solving rules of the Fig.3 are rules for the goal that user wants to look for the specific paper. The last two blocks in problem-solving rules are rules for the goal that a user wants to collect many papers (10 papers) related to the key word.

### 3.2.2. Action-management rules

Action-management rules are the description of rules for state transition. The state of this automaton is the action of the spoken dialogue system in principle. The arc of this automaton is user's action, a result of the application command execution and the outside event such as a result of library execution. Slot information is necessary to execute some libraries. Automaton is described for every abstract action plan decided with problem -solving rules. This automaton is called sub-automaton. The relation of sub-automatons that are materialized is shown in the Fig.4 taking a paper retrieval task for instance.

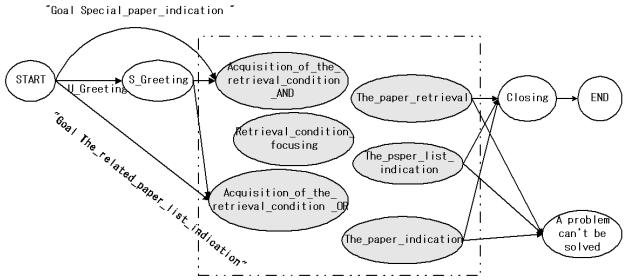


**Fig.4:** Relations of sub-automatons (paper retrieval task)

Only when "the greeting" utterance type is received from intention analysis of the first user's utterance, the state transits to greeting sub-automaton in the Fig.4. When "special paper indication" or "related paper list indication" is received as a goal for a solution of a problem, the action plan of "Acquisition_of_the_retrieval_condition_AND" or "Acquisition _of_the_retrieval _condition_OR" is chosen. And current state transits to the initial state of sub-automaton corresponding to each action plan regardless of the transition in greeting sub-automaton. Current state of Sub-automaton is "end", what means the action plan that coped with sub-automaton is attained. If another sub- automaton is chosen because a goal for a solution of a problem is changed before reaching "end" state by problem-solving rules, sub-automaton is changed. And current state is changed to the "start" state. One action plan in Fig.4 is expressed as sub- automaton is shown in Fig.5.

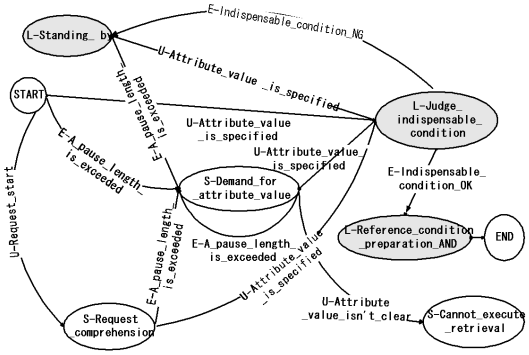


**Fig.5:** Example of sub-automaton (paper retrieval task)

Sub-automaton is described by description language of action-management rules. Also, action-management rules have the description form like SGML format.

# 4. EVALUATION OF ACTION-MANAGEMENT RULES

We tried to describe the action-management rules in reference with the actual dialogue data of academic paper retrieval task. By comparing described action-management rules with some other dialogue data of paper retrieval task, we evaluated generality of the action-management rules' regulation.

## 4.1. Method of evaluation

To evaluate action-management rules' regulation, we took 6 dialogue data in total with recorded at Waseda Univ., Science Univ. and Univ. of Electro-communications by each 2 data. A tag that corresponds to the arc and the state of action-management rules was given by every utterance of these data. The method of evaluation is as following. A correct tag set $S_c\{S_{c1},S_{c2},\ldots S_{cn}\}$ is compared with the evaluation tag set $S\{S_1,S_2,\ldots S_n\}$ which are got from every utterance of each dialogue data by tagging. '$c_n$' means the number of correspondence of tags, and '$l_n$' means the number of in-correspondence of tags when the utterance can't be tagged because it doesn't exist in the $S_c$. '$i_n$' means the number of insertion that is extra transition contained in the $S_c$. '$s_n$' is the number of substitutions, which is the case that the utterance tagged as explicit confirmation with the evaluation tag set S, but the utterance is tagged as implicit confirmation in the correct tag set $S_c$, for example. Then, an adjustment rate (A.R) and correct precision (C.P.) were found from these values. An equation is as the next.

$$A.R. = (c_n - l_n - s_n) / \text{Whole number}$$

$$C.P. = (c_n - l_n - s_n - i_n) / \text{Whole number}$$

These values are calculated for only state (an action of the system) or using only arc (outside event such as an action of user), using state and arc.

## 4.2. Result and discussion

The adjustment rate and correct precision caluculated in the way of evaluation expressed with 4.1 are shown in the table 1.

Value(%) = A.R.(C.P)

| Evaluation data | Only state | Only arc | State and arc |
|---|---|---|---|
| Waseda Univ. | 33%(14%) | 76%(36%) | 50%(20%) |
| Science Univ. of Tokyo | 67%(-10%) | 100%(44%) | 89%(-5%) |
| Univ. of Electro-Comm. | 85%(-7%) | 87%(29%) | 79%(7%) |

**Table1:** Adjustment rate and correct precision

As for evaluation data of Waseda Univ., it is found that the description ability is lower than other data from the Table 1. The reason of this tendency is that there are many turns in the data of Waseda Univ., so many transitions which can't be described with sub-automaton. Examples of transitions that can't be described are the utterance of a proposal from the system and the utterance of a question to the system. From this result, we should implement sub-automaton considering these utterances. So we should consider using history of dialogue to the system.

As for evaluation data of Science Univ. of Tokyo and data of Univ. of Electro-Comm., it is found that description ability (state's adjustment rate) of the system's intention of utterance is low because there is much substitutions from the Table 1. The reason of this tendency is that both implicit goal and explicit goal are seen in those evaluation data. So the consideration of the way of confirmation is important.

These results are only indicating the description ability of action-management rules for the dialog data used in the evaluation this time. But if a system designer refines rules based on these results, his requirement can be satisfied. This feedback loop of describing rules is effective for improving easiness to develop the system.

# 5. CONCLUSION AND FUTURE WORK

In this research, we designed and implemented a domain independent platform for task-oriented spoken dialogue system. We proposed how to describe the knowledge for problem-solving and one for controlling dialogue as rules considering trade-off problem between variety of dialogue strategies and easiness to describe rules. Concretely, we constructed the platform of spoken dialogue system that can act using problem-solving rules and action-management rules. Furthermore, we evaluate the ability of dialogue description by using this method.

In the future, the consideration of the way of confirmation and using history of dialogue are necessary. And we will examine how to make action–level rules semi-automatically. Furthermore, we will use other modalities, for example, face expression, eye gaze, gesture, and so on.

# 6. ACKNOWLEDGEMENTS

The current work was supported by Japan Society for the Promotion of Science as a Project on 'Research for the Future' (Project NO.JSPS-RFTF-96R15201).

# 7. RECERENCES

1. T.Kawahara, K.Tanaka and S.Doshita, "Domain-Independent Platform of Spoken Dialogue Interfaces for Information Query," *Proc. ESCA workshop on Interactive Dialogue in Multi-Modal Systems*, pp.69-72, 1999.

2. A.Pargellis, J.Kuo and C.-H.Lee, "Automatic Dialogue Generator Creates User Defined Applications," *Proc. EUROSPEECH '99*, pp.1175-1178, 1999.