EUROPA : A Generic Framework for Developing Spoken Dialogue Systems

Munehiko SASAJIMA, Takehide YANO, and Yasuyuki KONO
Kansai Research Center, TOSHIBA Corporation
8-6-26 Motoyama Minami, Higashi-Nada, Kobe 658-0015, JAPAN
{sasa,yano,kono}@krl.toshiba.co.jp

ABSTRACT

Voice interfaces are not popular since they are neither useful nor user-friendly for non-specialist users. In this paper, EUROPA, a new framework for developing spoken dialogue systems, is introduced. In developing EUROPA, the authors focused on three points : (1) acceptance of spoken language, (2) portability in terms of domain and task, and (3) practical performance of the applied system. The framework is applied to prototyping a car navigation system called MINOS. MINOS is built on a portable PC, can process over 700 words of recognition vocabulary, and is able to respond to a user's question within a few seconds.

Keywords: spoken dialogue system, voice interface, car navigation system

1. INTRODUCTION

Recently, some systems such as car navigation systems or information desk systems are equipped with a voice interface. However, voice interfaces are not popular since they are neither useful nor user-friendly for non-specialist users. To solve this problem, this paper focuses on the following three points.

First, the voice interface should accept spoken language. Application of the voice interface is effective especially when the user is not able to use his/her hands because of another task, for example, driving a car, cooking in a kitchen, or operating a power plant. It would be difficult for such users to communicate their intention in written language, which involves many grammatical constraints.

Second, it is very important for voice interfaces to be independent of task and domain as much as possible. In developing voice interfaces, the need for modifications, such as enhancement of domain knowledge or change of task, frequently arises. The task- and domain-independent framework reduces the cost of feeding back the requests to the interface.

Lastly, the system with the voice interface should answer users' questions within a short time. Some tasks such as car navigation or cooking require prompt answers. Slow systems are not useful.

The authors adopted three approaches and integrated them into a framework called EUROPA, which stands for "Environment for building Utterance RecOgnizable PAckages." To accept spoken language, the voice recognition module of the spoken dialogue system based on EUROPA does keyword-spotting. A sequence of spotted keywords represents a sentence as well as a user's intention.

Next, to make dialogue systems task- and domain-independent, EUROPA separates modules/data into the domain-dependent modules/data and domain-independent modules/data. To run a dialogue, the dialogue system must control a set of modules which belong to one of the two groups. Our framework copes with this point by adopting an interpreter for such codes that manage dialogue.

Lastly, to gain enough speed for the response, we have developed a BTH parser[1] for parsing keyword-lattices. Also the script for managing problem solving is compiled beforehand into another form to be interpreted faster.

EUROPA is applied to prototyping a car navigation system. The system answers two types of questions. One is the location of something such as a facility, a service-area, a parking lot or a shop. The other is a duration between two locations. All necessary modules are built on one notebook PC(Pentium II 266MHz), which accepts more than 1 million patterns of sentences and in almost all cases answers within 2 seconds.

2. THE EUROPA FRAMEWORK

2.1 Overview

We have designed EUROPA as a generic framework for spoken dialogue systems. Figure 1 shows the overall process of man-machine dialogue in EUROPA.

The word-spotting engine recognizes the user's utterance and it generates a keyword lattice as a recognition result. The obtained lattice is parsed by the BTH parser

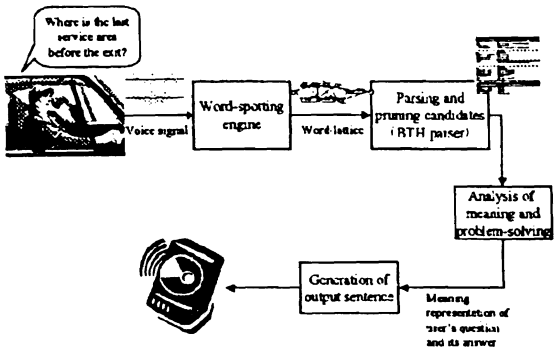

Figure 1: Dialogue process

to generate a set of possible keyword sequences, each of which is acceptable by the given grammar. The set is analyzed and solved in terms of meaning and the most plausible keyword-sequence in the set is selected. Finally, the reply to the user's question is played back by synthesized voice.

2.2 BTH Parser

To accept spoken language, our voice recognition module does keyword-spotting and outputs a keyword lattice. Parsing the keyword lattice, the parser extracts plausible word-sequences. Each of the word-sequences represents the user's intention. In the case of spoken Japanese, misuse or loss of particles often occurs. Keyword spotting does not deal with them, which simplifies acceptable grammar rules. This characteristic also solves the problems of dealing with unnecessary terms such as "aah" or "well." Just by excluding them from the keyword set, we can accept sentences with these words.

Furthermore, change of word order which also occurs in Japanese spoken dialogue can be easily dealt with. The BTH parser is employed for efficiently parsing the keyword lattice, which is obtained by keyword-spotting, and it is transformed into a set of possible keyword-sequences. The details of the BTH parser are described in [1].

In the case of the task, it is common for over 100 spotted words to be notified from the recognition engine, and consequently over 1 million possible word sequences can be generated by unfolding the corresponding lattice even if word-class bi-gram is applied to the lattice. BTH, however, is able to parse such a large lattice within a practical time.

2.3 USHI:Script-based method for Dialogue management

Obtaining the set of possible keyword-sequences from the BTH parser, the dialogue controlling module processes it to generate a reply. The overall configuration of the module is depicted in Figure 2.

2.3.1 Dialogue management process

Firstly, it transforms the given set into a set of representations of input intention, each of which corresponds to a word-sequence candidate, by employing Intention Translator.

Then it resolves a user's question by referring to the knowledge base and generates meaning representation of the query that corresponds to the most plausible word-sequence and its answer. Obtaining the information, the Response Generator generates responding sentences. The problem-solving process is based on unification of feature structures.

2.3.2 Enhancement of portability

Portability is required for the framework for building a spoken dialogue system. In the prototyping and testing cycle, the need for modifications, such as enhancement of domain knowledge or change of task frequently arises, even if the domain and the task do not change.

As shown in Figure 2, EUROPA separates modules into two groups. One consists of the domain dependent modules/data such as a word dictionary, grammar rules, rules for translating user intentions, rules for generating sentences as an answer to the user input, and domain-specific problem solvers, such as Place-expression Resolver and Route Resolver. The other consists of domain-independent modules/data such as a lattice parser, an interpreter for translating user intentions, and an interpreter for generating answer sentences.

2.3.3 USHI specification

To run a dialogue, the dialogue system must control a set of modules, which belong to one of the two groups. For example, a car navigation task that solves a location specified by a user utterance, requires not only a generic parser but also a domain-specific problem solver which resolves the location the user intended. Embedding such control codes including domain-specific parts reduces portability of the framework.

EUROPA solves this problem by adopting an interpreter for such codes that manage dialogue. We call this interpreter the "USHI Interpreter" which stands for "Unification-based Script Handling Instruction set Interpreter." A system developer describes the process of meaning analysis, problem-solving, and response generation in an USHI script, and the module interprets the script.

USHI script language is a Pascal-like language and has the following three features: (1) Subset of statements for expressing selection and looping, (2) Unary and binary operators for calculation and comparison, and (3) Ability to invoke a function defined in the other part of the script or a system embedded one written by C++. For dealing with feature structures and knowledge base, the script has two more features.

(1) Unification between feature structures.

(2) Access to the knowledge base.

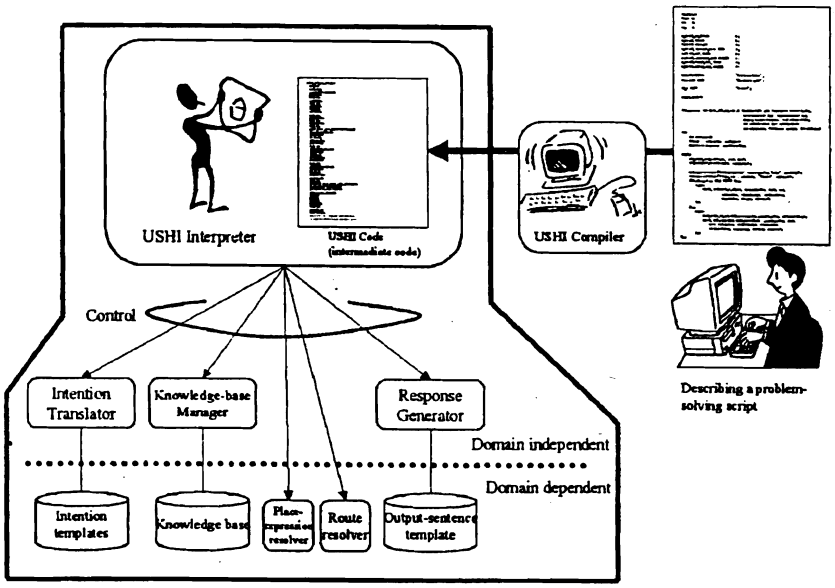

Figure 2: Dialogue controlling modules

Furthermore, to gain enough speed for the response, the USHI script for managing problem solving is compiled beforehand into another form to be interpreted faster, as shown in Figure 2.

2.3.4 Dialogue management example

For example, a car navigation task which solves the user's query about the location of an interchange is controlled by the USHI interpreter and an USHI script which includes a procedure consisting of two steps.

Step1: If the user's intention is to seek for a place that satisfies some conditions, then call a function which resolves the user's expression about a place.

Step2: If the user's expression is resolved, then call the response generator to generate an answer sentence which includes the user's original expression and the answer.

When a word-sequence "Where is the next interchange?" is input, the Intention Translator translates it into a data structure called "user intention." A user intention is a form of typed feature structure. Translator is a domain independent module and has a knowledge base which is initialized by Intention templates that specifies mapping rules among utterance patterns and the user intentions. The utterance of this example is translated into the following structure [1]. Letters left-hand side of the sign of aggregation "(" means the type symbol, left-hand side of the ":" means a feature, and the right-hand side of the ":" means the value of the feature.

```
UserIntention(Intent:AskNameOfPlace
    Expression-Of-Place:
        Place-expression(
            ClassExpression:Interchange,
            RelativeOrderExpression:Next))
```

Next, the USHI interpreter interprets the USHI script ***Step1***, and calls the Place-expression resolver with an argument which specifies the condition of the location.

```
Place-expression(
    ClassExpression:Interchange,
    RelativeOrderExpression:Next)
```

The Place-expression resolver utilizes the Knowledge-base Manager and accesses the route data set by the user, data about geographical features, location of objects such as shops or gas-stations, and so on.

If the Place-expression is resolved, the USHI interpreter proceeds to the ***Step2***. For example, if the resolved answer is "Suita Interchange," then an answer sentence "The next interchange is Suita Interchange." is generated. The sentence in the form of text is input to the TTS module, which is a former version of the TOS Drive TTS[2], and output.

2.4 MINOS: an application of EUROPA

EUROPA is applied to prototyping a car navigation system, called MINOS. Figure 3 shows a view of the system. A sample screen copy of MINOS is shown in Figure 4.

The system answers two types of question. One is the location of something such as a facility, a service-area,

[1] The structure implemented on MINOS has more features, and unbound features are not printed here.

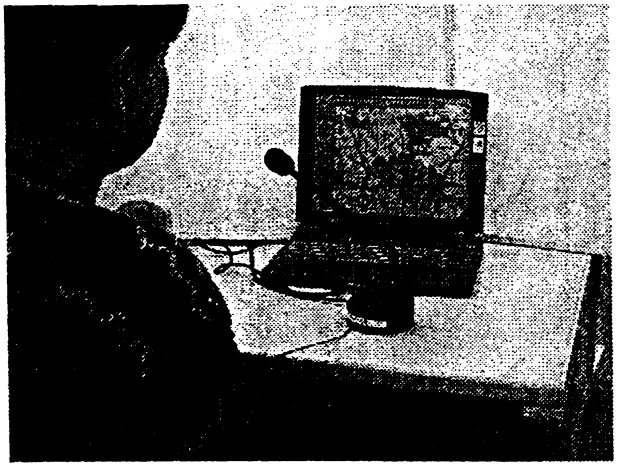

Figure 3: View of MINOS

a parking lot or a shop. The other is the duration between two locations. All modules including the voice recognition, the dialogue control, and the text to speech engine are built on one notebook PC (Pentium II 266MHz), which accepts more than 1 million patterns of sentences and in almost all cases answers within 2 seconds.

MINOS is able to respond to queries with respect to locations made by a user while driving a car. Modules in MINOS run collaboratively, communicating with each other via socket-based messaging. As shown in Figure 4, the application module of MINOS has a map-based appearance, which is based on ProAtlas, a map software developed by ALPS Mapping Co., Ltd. It simulates car driving by replaying prerecorded position data obtained from the Global Positioning Satellite (GPS) system.

The application module simulates driving a car and continuously notifies the dialogue module of the simulated current position at regular intervals of time. When a user asks a question in Japanese, e.g., "Deguchi-no-mae-no-saigo-no-service-area-ha-doko? (Where is the last service area before the exit?)", the word-spotting engine recognizes the utterance and notifies the dialogue controlling module of a keyword lattice. MINOS can process over 700 words of recognition vocabulary.

Next, MINOS analyzes the keyword lattice by employing the BTH parser and obtains a set of candidate word-sequences, referring to the over 1 million sentence patterns. The word-sequences are sorted in the descending order of the initial score of each candidate, which is calculated from phonetic scores of the candidate's words. Each word-sequence candidate is converted into a user's intention in the form of a typed feature structure, and is resolved by using current position information and the knowledge base. The knowledge base is a semantic network that contains all the knowledge required to solve questions about locations on the displayed map.

The score of each candidate is revised in terms of phonetic value and meaning, which is the cost of problem solving, and the list of the candidates is reordered. As a result, the candidate with the best score is selected and the text of the answer corresponding to the question is generated, e.g., "Amagasaki Service Area is the last service area before Nishinomiya Interchange" and the dialogue controlling module notifies the TTS engine of it.

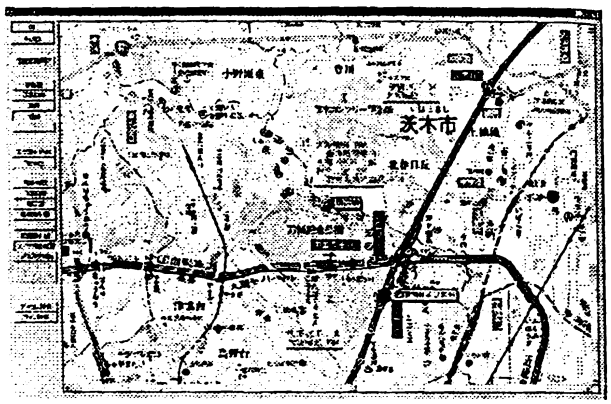

Figure 4: A screen copy of MINOS

One more function of EUROPA is asking back to the user when the score of the sentence is lower than a given threshold, such as "Did you ask about a restaurant just before the destination?"

3. CONCLUSION

We have developed EUROPA, a generic framework for spoken dialogue systems. Aiming at enhancement of the scale of the task and portability in terms of task and domain, it employs an efficient keyword lattice parser, BTH[1] and script-based method for dialogue management, USHI. EUROPA was applied to prototyping a PC-based spontaneous speech interface for car navigation tasks, MINOS.

The result shows promise with respect to implementation of the function of a spontaneous speech interface in next-generation car navigation systems equipped with RISC MPUs of about 70MIPS.

Currently, MINOS is a question-answer system, and future work will consider about more interactive dialogue and the resolution of the reference such as usage of pronouns.

4. REFERENCES

[1] Kono, Y., Yano, T., and Sasajima, M.(1998), BTH: An Efficient Parsing Algorithm for Word-Spotting. *Proceedings of ICSLP '98*, pp. 2067–2070.

[2] Akamine, M. and Kagoshima, T.(1998), Analytic Generation of Synthesis Units by Closed Loop Training for Totally Speaker Driven Text to Speech System (TOS Drive TTS). *Proceedings of ICSLP '98*, pp. 1927–1930.