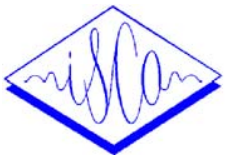

4th International Conference on Spoken Language Processing (ICSLP 96)
Philadelphia, PA, USA
October 3-6, 1996

Dialog in the RAILTEL Telephone-Based System*

S. Bennacef, L. Devillers, S. Rosset, L. Lamel

LIMSI-CNRS, BP 133
91403 Orsay cedex, FRANCE
{bennacef,devil,rosset,lamel}@limsi.fr

ABSTRACT

Dialog management is of particular importance in telephone-based services. In this paper we describe our recent activities in dialog management and natural language generation in the LIMSI RAILTEL system for access to rail travel information. The aim of LE-MLAP project RAILTEL was to assess the capabilities of spoken language technology for interactive telephone information services. Because all interaction is over the telephone, oral dialog management and response generation are very important aspects of the overall system design and usability. Each dialog is analysed to determine the source of any errors (speech recognition, understanding, information retreival, processing, or dialog management). An analysis is provided for 100 dialogs taken from the RAILTEL field trials with naive subjects accessing timetable information.

1. INTRODUCTION

The LE-MLAP project RAILTEL "Railway Telephone Information Service" aimed to evaluate spoken language technology in the context of interactive voice services for railway transportation. This project included an assessment of user needs, of service provider needs, and the technical adequacy of available techniques. State-of-the-art spoken language technologies were validated through a field trial with naive users accessing train timetable information.

The LIMSI RAILTEL system[6] provides access to the SNCF static timetable information, as well as limited additional information about services offered on the trains, fare-related restrictions and supplements is also available. The system is largely based on the spoken language system developed for the ESPRIT MASK project. The system is composed of a speech recognizer, and components for natural language understanding, dialog management and response generation. The speech recognizer transforms the input signal into the most probable word sequence and then forwards it to the natural language understanding component which carries out a caseframe analysis and generates a semantic frame representation. The dialog manager prompts the user to fill in missing information and then generates a database query. The retrieved information is formatted in a natural language response by the response generator (taking into account the dialog context) and vocal feedback is provided to the user. To ensure high quality speech output, synthesis by waveform concatenation is used where dictionary units are put together according to the generated text string.

*The field trials were partially financed by the LE-MLAP project 63-022 RAILTEL. The continuation of this work will be partially financed by a follow-up project.

Since the prototype system is only voice-activated, all interaction with the user and all information returned by the system, such as the list of possible trains, train departure and arrival times, changes, fares, etc, must be exchanged vocally. Therefore, oral dialog management, response generation, and high quality speech output have a strong influence on the perceived performance and usability of the system. This paper focuses on issues in the dialog design, response generation and evaluation.

2. SPEECH UNDERSTANDING

The speech understanding component transforms speech acoustic signal into semantic-pragmatic representation following the three stages: speech recognition, literal and contextual understanding.

2.1. Speech recognition

The speech recognizer is a medium vocabulary, speaker-independent, continuous speech recognizer[3]. It is a software-only system (written in ANSI C) that runs in real-time on a standard Risc processor. Speaker independence is achieved by using acoustic models trained on speech data from a large number of representative speakers, covering a wide variety of accents and voice qualities.

The recognizer uses continuous density HMM with Gaussian mixture for acoustic modeling and a *n-gram* backoff language models[5]. The feature vector contains 12 MFCC cepstral coefficients computed on the 0.3-4kHz telephone band and their first and second order derivatives[4]. The *n*-gram statistics are estimated on the transcriptions of spoken queries. Since the amount of language model training data is small, some grammatical classes (such as cities, days, months, etc) are used to provide more robust estimates of the *n*-gram probabilities. The current RAILTEL recognition vocabulary contains about 1500 words, including the 680 station/city names specified by the SNCF.[1]

2.2. Literal understanding

After recognition, each utterance is analysed, using a caseframe grammar[1], in order to build one or several semantic frames which are saved in a semantic frame network. This analysis does not require verifying the correct syntactic structure of the whole utterance, but rather extracting its meaning using local syntax as a constraint only whenever necessary. The caseframe parser has been implemented in C++. The caseframe grammar is described in a declarative file so as to allow for easy modification of the cases. The concepts for the RAILTEL task are **train-time**, **fare**, **change**,

[1]The recognition vocabulary used in the field trials contained 800 words, including 58 station names.

10.21437/ICSLP.1996-66

type, **reserve**, **service** and **reduction** and have been determined by analysis of queries taken from the training corpora to augment the *a priori* task knowledge. The resulting semantic frame contains a set of slots instanciated by the meaningful words of the utterance.

2.3. Contextual understanding

Contextual understanding consists of interpretating the utterance in the context of the ongoing dialog, taking into account common sense, task domain knowledge. Semantic frames issued from the literal understanding are reinterpreted using rules to supply default values and to transform qualitative values.

Default value rules supply default values not specified by the user. For example, if the departure month has not been specified *"I would like to go on the 6th"*, the current month is taken by default (or the next month if the 6th has already past).

Interpretative rules transform imprecise values given by the user into appropriate ones used by the system. For example, the utterance *"I want to go this morning"* is transformed into *"I want to go today between 6 am and 12 noon"*.

Semantic frames corresponding to the current utterance are then completed from the **dialog history** in order to take into account all information previously given by the user: departure, arrival cities, departure day, etc..., as well as questions generated by the system. These questions are saved as part of dialog history - the **generation history**. The generation history enables the system to interpret the users responses to system initiatives. For example, it allows to resolve ellipses as in the following exchange:

System: *What is your departure city ?*
User: *Paris* (Paris is assumed to be departure city in the context of the previous question.)

3. DIALOG MANAGEMENT

3.1. Dialog structure

The information retrieval dialog is divided into three phases: main information exchanges, preceded and closed by formalities. Each dialog is structured into a hierarchy of sub-dialogs with a particular functional value. This value may concern the task to be achieved, the dialog itself, or the metadialog.

- Subdialogs concerning the task are application-dependent, in so far as the information exchanged includes values directly related to the task. Task-specific subdialogs include *request, response, precision* and *explanation*.
- Subdialogs concerning the dialog are application-independent and involve the dialog structure and organization, such as the *opening* and *closing formalities*.
- The metadialog corresponds to the parts of discourse which do not directly concern the information enquiries, but relate to the dialog itself and the way communication is handled: for example *reformulation, confirmation, hold-on* and *restart*.

In order to formalize the dialog, we have combined formal grammars and speech acts theory to represent the dialog model by a set of rules[2]. The grammar non-terminals correspond to subdialogs, and terminals correspond to dialog acts. Some example rules for initiating different subdialogs are given below. Each rule generates a dialog act which controls the opening, closing and message generation of the subdialog.

Opening formality subdialog: At the beginning, the system starts an *opening formality subdialog* which consists of the presentation of the system. If the user responds with a formality, the system asks a specific question to guide the user, an example of a *restart subdialog*.

Closing formality subdialog: When the user closes the dialog with a politeness form, the system opens a *closing formality subdialog* to thank the user. In fact, one of the hardest control problems is to detect that the dialog is finished.

Task rules: If the semantic frame is incomplete with respect to information needed for database access, a *precision subdialog* generates questions requesting the user to supply specific information. When the semantic frame is complete, a DBMS query is generated (in SQL, for example) using *task request generation* rules. Similarly a response is constructed using the *natural language response generation* rules and played to the user.

Explanation subdialog: If the user does not respond to a system request for information, but instead asks for an explanation, an explanation subdialog is initiated.

Reformulation subdialog: If the semantic analyser fails to build a semantic frame, the dialog manager asks for repetition by opening a *reformulation subdialog*. Reformulation messages, such as *"I am sorry, I have not understood, can you repeat that?"* invite the user to repeat their previous request.

Confirmation subdialog: Incoherence detection rules check for incoherence in the semantic frame. For example, if someone says (or the system understands) *"I want to go from Paris to Paris"* the dialogue manager opens a *confirmation sub-dialog*.

Metadialog: In telephone-based dialog, messages are important to keep the user informed and online. For example, if the speech recognition or database access times are long, a *hold-on sub-dialog* generates messages (*"Hold-on please, we are trying to satisfy your request"*) to inform the user they need to wait.

3.2. Dialog strategies

The spoken language system uses a mixed-initiative dialog strategy, where the user is free to ask any question, at any time. However, in order to aid the user, the system prompts the user for any missing information needed for database access. Experienced users are thus able to provide all the information needed for database access in a single sentence, whereas less experienced users tend to provide shorter responses, allowing the system to guide them. Example dialogs solving the first scenario in Figure 2 are given in Figure 1.

Another strategy of the system is to never give a negative response to the user, unless the information is really not available. To do so the system must relax the constraints provided by the user in order to propose a solution. For example, if the temporal constraints given by the user are too restricted, the system suggests the closest train to the specified departure or arrival time.

An important issue is to correctly manage the dialog history. To do this, it is necessary to be able to add and remove information from the history. A set of rules determine which constraints previously specified by the user should be forgotten when, so as to provide a more natural and flexible dialog. The principle is to attach to each constraint, a set of other constraints via *functional dependencies*. Two approaches allow information to be forgotten:

System Opening greeting
Expert *I'd like to know the time of a direct train to Lille leaving Paris around 10am on March 14th.*

System Opening greeting
Novice *I would like to go to Lille.*
System What city are you leaving from?
Novice *I'm leaving from Paris.*
System What date do you wish to travel?
Novice *March 14th.*
System What time of day do you want to leave?
Novice *Oh, I guess around 10.*

Figure 1: Example dialogs for expert and novice users solving scenario **A** in Figure 2.

Explicit approach: If the user explicitly changes the request, by asking for example about all trains, the system forgets all previously specified information except for departure and arrival cities, and the departure date.

Implicit approach: Each time the user modifies a constraint, all dependent constraints are removed from history. For example, *if the speaker changes the name of the* departure-city, *the system deletes all linked constraints in the dialog history such as the arrival-city, departure-time, etc..., except the* departure-day.

4. MESSAGE GENERATION

In contrast to the MASK kiosk where different media are used to return information to the user, there is no visual support in the telephone communication. Since the only possibility is to return information orally, response generation plays a very important role in the overall system. The generation of responses is complex because if too much information is given, it may be difficult for the user to extract the important part, yet if not enough information is returned, the interaction will take longer, as the user will need to ask for additional information.

Different types of responses can be generated during the dialog depending upon dialog structure: system presentations, prompts (hold-on sub-dialog), restarts, requests for specific information (precision sub-dialog), responses, reformulations, confirmations and domain explanations. The response generator is based on a formal grammar, where non-terminals are conditioned by the context. At each user dialog act, the response generator builds a sentence where gaps are filled in from the content of the current semantic frame, the dialog history and the DBMS response. Careful attention has been paid to construct natural sounding sentences that contain the appropriate contextual information, when possible, summarized in a single sentence.

The top level grammar's rules for interaction with the user are:

- If there are more than 10 possible trains, inform the user of this and ask for additional information about time period to limit the possibilities.
- If there are between 3 and 10 trains, tell the user the number of trains, giving the departure time and type (or fare) for the first and last train. Ask for a more precise departure time.
- If there are 3 or fewer trains, return the departure time, type (and optionally fare) for each train.
- To obtain more information, such as train identifier, changes or services, the user must select only one train.

5. EVALUATION

Evaluation of spoken language systems remains an outstanding research issue. We have chosen to use a multilevel evaluation approach, which distinguishes 3 different levels in the system: recognition, understanding and dialog. Each level is evaluated by differentiating errors caused at the current level from errors propagating from the lower levels. Thus, to evaluate the understanding level we separate out errors due to recognition errors from those attributable to the understanding component. Similary, the dialog is evaluated by differenciating between errors due to the recognition and understanding levels and those arising from the dialog level. The LIMSI system was evaluated in a field trial using a methodology commonly defined by RAILTEL partners.

5.1. Field trial methodology

A total of 100 naive subjects (48 female/52 male ranging in age from 18 to 65 years) were recruited for the field trial. For each subject, it was the first call to the system in natural conditions (from at home). Each subject was asked to complete a questionnaire immediately after interacting with the system.

Half of the subjects had a scenario of type **A**, the other half had a scenario of type **B**, as shown in Figure 2. In scenario **A** *city_A* and *city_B* must be connected by a direct train, and the time and date of travel are specified. In scenario **B** traveling from *city_A* to *city_B* requires a change of trains, and the time and date of travel are specified only in general terms. There are multiple formulations for each kind of scenario. For example, another wording for scenario **A** is: *You want to go from Paris to Lille on March 14th, leaving around 10 am.* Combined with the different town names, dates and train times, a large set of different scenarios can be generated.

A- You want a direct train from [*city_A*] to [*city_B*] on [*date*] [*time*].
(*You want to take a direct train from Paris to Lille on March 14th leaving at 10 am.*)
B- You would like to know the arrival time of an [*time-period*] train from [*city_A*] to [*city_B*], [*relative-date*].
(*You would like to know the arrival time of an evening train from Grenoble to Paris next Monday.)*

Figure 2: Prototype scenarios used in the field trials.

After completing the field trial scenario, each subject solved 4 other scenarios in increasing difficulty in order to collect data for a wider variety of situations. Some of the scenarios required subjects to ask about information not yet treated by the system, to see their reaction when the system could not provide the information they wanted.

5.2. Results and Analysis

Global results: The field trial results are reported for 100 calls, the first 50 calls received for each scenario type. The average number of turns per call was 3 for type **A** and 5 for type **B**. Scenarios of type **B** had more turns due to the imprecise specification of the time of travel and some problems encountered interpreting arrival times. As a consequence, the type **A** calls had a shorter duration than type **B** (193s vs 245s). The overall dialog failure rate was 27% (24% for type **A**, 32% for type **B**).

Multilevel evaluation: We have carried out a multilevel analysis of the dialog system. The lowest level, recognition, is the most simple to evaluate. For this we use the commonly adopted measure of word error. The speech recognition component was evaluated on an independent set of test sentences, and has a word error of about 18%. However, this number can be misleading as the word accuracy measures all differences between the exact orthographic of the query and the recognizer output. Many recognition errors (such as masculine/feminine forms, or plurals) are not important for understanding.

The understanding evaluation is done on the semantic frame corresponding to each query. For each slot which is incorrectly instanciated, the error source, recognition or understanding, is marked. It is then straightforward to compute the incorrect slot instanciation rate (recognition or understanding) for the semantic frame by simply dividing the number slot errors by the total number of slots. The average recognition and understanding query error rates for the field trial data are given in Table 1. The error rates by slot type are given in Table 2, where the errors are divided by the total number of instantiated slots of that type after literal understanding. Understanding errors for DepCity and ArrCity usually involve reversing the two.

Scenario Type	*Recognition*	*Understanding*
A	23.2%	10.7%
B	20.0%	6.0%

Table 1: Understanding error rate on semantic frames per queries.

Type	*DepCity*	*ArrCity*	*Date*	*DepTime*	*ArrTime*
#slots	72	69	202	264	-
A (rec)	13.8%	15.4%	16.4%	19.1%	-
A (und)	0.0%	0.0%	8.8%	17.2%	-
#slots	98	96	144	191	142
B (rec)	8.8%	8.6%	19.2%	23.7%	13.4%
B (und)	0.2%	1.1%	0.0%	24.4%	17.3%

Table 2: Incorrect slot instantiation rates due to recognition and literal understanding errors by type of slot.

The dialog evaluation is done by looking at the system response. Each time the system response is judged incorrect, the source of error is indicated as recognition and/or understanding (reco/und) or dialog management. The dialog errors are calculated by the ratio of sentences marked as erroneous and the total number of system responses.

The Table 3 shows the dialog error rates for both scenario types. There are a larger percentage dialog errors due to recognition and understanding for the type **A** scenarios than for type **B** scenarios. This is due to digit recognition errors which were more common in the type **A** scenarios in which explicit dates and times were specified. These errors did not generally result in dialog failure, as the user usually corrected the error in a later turn, thus successfully completing the call. Type **B** scenarios had a larger number of dialog errors, as the system did not correctly respond to queries where the arrival time was specified and it was necessary to depart the previous evening to arrive at the specified time. Even though this error was due to database access, we have considered it a dialog error as the system response was not correct in the context of the user's query. The field trials turned up a database access problem, that was corrected early on (column DB).

Scenario Type	*# Dialogs*	*Correct*	*Cause of error*		
			Reco/Und	*Dialog*	*DB*
A	50	58.5%	34.3%	0.9%	6.3%
B	50	60.5%	29.2%	10.3%	-

Table 3: Source of dialog error per system response.

6. CONCLUSION AND PERSPECTIVES

The dialog component of the LIMSI RAILTEL spoken language system has been described, and 100 dialogs from the field trial have been analysed. It is important to point out that these results were obtained in the context of a directed task, where naive users were asked to solve relatively simple scenarios. We have found that subjects tended to closely follow the instructions. The majority of first queries (81% for type **A** and 97% for type **B**) contain all the information necessary for database access. Each subject also solved an additional 4 scenarios, with different difficulties and presentation styles to collect more varied data. The results of these additional dialogs are underway. The qualitative assessment of the service was favorable, with subjects judging the system both user-friendly and quick, while expressing the need for improvement. Most subjects were interested in using such a service.

One problem of the system driven aspects of the RAILTEL dialog manager is that it does not take into account the user's intention. While it is difficult to know the intention of the user, modeling different typical user interactions can eventually provide guidance for constraint relaxation, efficient history management and selection of confirmation strategies. Modeling user intention can play an important role in correcting recognition errors.

Another important issue in dialog management is to correctly handle the dialog history. In the current system, the history is maintained as an unstructured set of semantic frames without links to the dialog structure. We plan to restructure the history and so as to be able to extract a meta-history which can give an overview of a dialog state at each exchange, for more accurate analysis of the new query.

REFERENCES

[1] S. Bennacef, H. Bonneau-Maynard, J.L. Gauvain, L. Lamel, W. Minker, "A Spoken Language System For Information Retrieval," *ICSLP'94*.

[2] S. Bennacef, F. Neel, H. Bonneau-Maynard, "An Oral Dialogue Model based on Speech Acts Categorization," *ESCA Workshop on Spoken Dialog Systems*, Vigsø, Denmark, Spring 1995.

[3] J.L. Gauvain, L.F. Lamel, G. Adda, M. Adda-Decker, "Continuous Speech Dictation in French," *ICSLP'94*.

[4] J.L. Gauvain, L. Lamel, M. Adda-Decker, "Developments in Continuous Speech Dictation using the ARPA WSJ Task," *ICASSP-95*.

[5] S.M. Katz, "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer," *IEEE Trans. ASSP*, **35**(3), 1987.

[6] L. Lamel, S. Bennacef, H. Bonneau-Maynard, S. Rosset, J.L. Gauvain, "Recent Developments in Spoken Language Sytems for Information Retrieval," *ESCA Workshop on Spoken Dialog Systems*, Vigsø, Denmark, Spring 1995.