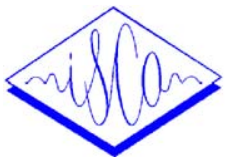

6th European Conference on
Speech Communication and Technology
(EUROSPEECH'99)
Budapest, Hungary, September 5-9, 1999

DESIGN STRATEGIES FOR SPOKEN LANGUAGE DIALOG SYSTEMS

Sophie Rosset, Samir Bennacef,† Lori Lamel
LIMSI-CNRS, BP 133, 91403 Orsay cedex, France
†VECSYS, 3 r. de la Terre de Feu - Les Ulis, 91952 Courtabœuf, France
{rosset,lamel}@limsi.fr sbennacef@vecsys.fr

ABSTRACT

The development of task-oriented spoken language dialog system requires expertise in multiple domains including speech recognition, natural spoken language understanding and generation, dialog management and speech synthesis. The dialog manager is the core of a spoken language dialog system, and makes use of multiple knowledge sources. In this contribution we report on our methodology for developing and testing different strategies for dialog management, drawing upon our experience with several travel information tasks. In the LIMSI ARISE system for train travel information we have implemented a 2-level mixed-initiative dialog strategy, where the user has maximum freedom when all is going well, and the system takes the initiative if problems are detected. The revised dialog strategy and error recovery mechanisms have resulted in a 5-10% increase in dialog success depending upon the word error rate.

INTRODUCTION

The dialog manager is the core of a spoken language dialog system (SLDS). It is the window through which users observe the behavior and apparent "intelligence" of the system. In the context of the LE3 ARISE project our objectives were to obtain a high dialog success rate with a very open dialog structure, where the user is free to ask any question or to provide any information at any point in time. Dialog management is based on three kinds of strategies: 1) strategies dependent on ergonomic choices, 2) strategies dependent on choices for metacommunication, and 3) strategies dependent on the task and domain knowledge. The first two types of strategies are quite closely inter-related.

The term "dialog strategy" is widely used, but its definition is quite vague and can be interpreted in different manners. By dialog strategy we refer to the interaction of the system with the user, which is evidently dependent upon design decisions determining the system characteristics and functionality.

BASIC PRINCIPLES

Ergonomic Choices

Ergonomic considerations determine the usability and essential characteristics of the SLDS. Given that our goal is to develop an operational prototype service, the ergonomic choices influence all other design considerations, subject of course to the technical constraints.

• *Freedom and flexibility:* Our goal is to allow the user the maximum freedom of expression, avoiding to impose constraints on the order in which information is provided as long as the dialog is going well. The user is free to say what s/he wants when and how s/he wants. This desired freedom has a direct influence on the understanding component and dialog managment.

• *Negotiation:* This is the possibility for the user to accept or refuse system proposals, adding and/or changing some constraints. Negotiation has direct implications for response generation, as the system must inform the user that this capacity exists. Negotiation is particularly important when there is no database entry satisfying the constraints specified by the user.

• *Navigation:* This is the identification of, or detection of a change in the task. Navigation also includes the possibility to ask about the functionality of the system and what types of information are available.

• *Initiative:* The initiative concerns who (the user or the system) directs the progression of the dialog. We have chosen to use a mixed-initiative dialog allowing the user maximal freedom.To enable an open dialog strategy with potential recognition and understanding errors, a multi-level dialog strategy is used, where the system can take the initiative if the ongoing dialog is not progressing well.

• *Contact with the user:* Keeping in close contact with the user is crucial. Two objectives were defined: to never let the user get lost and to answer directly to the users' questions[5]. The generation component is very important in assuring this contact.

Given the number of tasks and their interdependencies and related constraints, the negotiation and navigation capabilities are very important.

Metacommunication

Metacommunication is concerned with the overall flow of the dialog and in our view, involves mainly the identification of (or change in) the task and the detection of errors.

• *Identification of (or change in) task:* This is directly related to the ergonomic choice of providing the user with a navigation possibility. For the ARISE system the tasks are timetable information, fare information, reservations, other information (type of train, services, etc). In a more general tourist information system more tasks are supported and the system may suggest tasks to the user[3]. This aspect is mainly user-initiated but can be system-initiated, when specified by the task model. For example, in accordance with the French Railways (SNCF) specification, when making a reservation with the ARISE system, the user is systematically asked if s/he is eligible for a reduction. In response to such a system prompt, a simple term means that the user wants this type of reduced fare, which is added as a constraint but the task does not change. The same word can

also signal an additional constraint on a fare (leading to a negotiation dialog), or a request for information about reductions (resulting in a task change).

• *Detection of errors:* This aspect concerns the repair and clarification metacommunication whether system or user-initiated. Either the user or the system can have the impression that something went wrong and needs to be corrected. The system detects a potential error when contradictory information is obtained, which usually occurs when new information contradicts with what was previously understood.[1] The system can choose to ignore the new information, replace the old information with the new information, or enter into a confirmation dialog. The choice here is closely related to the development strategy chosen.

Error detection is of particular importance in an unconstrained dialog, as few constraints can be applied to minimize recognition and literal understanding errors[5]. In order to have as natural a dialog as possible, the system attempts to react according to the type of error detected, at the time it is detected. A correction initiated by the user can result from a system error (usually recognition or understanding) or can reflect a change of mind. Detecting a user correction can be difficult given the openness of the system, as users may not clearly state their correction. After a recognition error on the departure city a user said "No I don't want to reserve because the destination is not right, I want a train leaving from ..."[2] This example illustrates inconsistency on the part of the user who user said that the arrival city was incorrect but corrected the departure city.

Task Functionalities

The goal of the dialog manager is to respond to the user's query, when necessary asking the user to supply the items required for database access. The desired task functionalities and system capabilities determine the interdependency of constraints, such as what information to discard when a previously specified value is modified. For example, in the LIMSI ARISE[5] and MASK systems[1], a fare can only be provided for a specific train,[3] so the user must first make a selection. Concerning the reservation task, for many trains it is possible to purchase a ticket without a reservation, or to purchase a reservation alone. Two strategies are possible: the system can systematically ask the user if s/he wants to make a reservation or the system can simply assume that the user will want to reserve. In the case that a reservation is needed, the system should propose only trains where places are available in the correct category. On trains where reservations are required, the system should inform the user of this constraint and initiate the reservation task.

STRATEGIES & SLDS DEVELOPMENT

The above strategies are derived from our desire to develop a mixed-initiative dialog system which is user-friendly and intuitive, allowing the user to interact in a natural manner. The system should support negotiation, be capable of detecting topic changes (navigation), and to the extent possible, be able to detect and deal with errors. If all is going well, the user should be able to express him/herself freely, providing information in any order. We try to minimize the number of dialog turns, so as to respond as quickly as possible to the user's query. While evidently different users require different levels of guidance from the system, in general, long dialogs indicate that the user is experiencing problems. These considerations have a direct influence on development of the component modules of the SLDS, which will function according to the ergonomic and design decisions. The initial dialog strategies of the ARISE system were based on the study of human-human dialogs from the SNCF information center in Douai, France, and on our experience with other systems L'ATIS, MASK, RAILTEL. As data have been collected with the system, the strategies are more and more corpus based, taking into account how users interact with the system.

Dialog Manager

Different dialog phases can be identified: acquisition, negotiation, navigation, post-acceptance and metacommunication. During the acquisition phase, the system obtains the information needed to complete the current task. For timetable information, this is the departure and arrival cities, and the travel date; for fare information the class and reduction type must also be specified. The negotiation phase occurs when the user modifies his/her request as a function of the information returned by the system and navigation refers to when the user changes from one task to another (it is common to change from a fare request, to information about different types of reductions and then back to fares). If the user accepts the solution proposed by the system, the dialog enters into the post-acceptance phase. If the user does not spontaneously ask another question, the system will suggest another task. The new task depends on the current one: if the user has accepted a time, the system will ask if the user wants to make a reservation on this train. All pertinent items are considered to be accepted by the user and no longer subject to negotiation. If the user does not enter into any of the proposed tasks, the system closes the dialog. Metacommunication concerns the detection and treatment of errors.

The input to the dialog manager is the output of the literal understanding component. Based on this input and the different knowledge sources available to it, the dialog manager needs to generate an appropriate response to the user. The knowledge sources used and maintained by the dialog manager are the system-user exchange pairs history, the long term dialog history and a task model. From these sources a dialog state is inferred. The system also memorizes information even if it is not directly in response to the question. In the example of Figure 1 the system asks for a travel date and the caller responds with the time. The system reprompts the user with an explicit request for the travel date, but keeps track of the time so that it can access the database once the date is provided.

Each query passes through 3 stages of interpretation, each associated with specific actions.

1- Contextual understanding: This is an interpretion of the

[1] Contradictions rarely occur within a single query.

[2] All examples are translated from French.

[3] This is due to the tarification, as the price depends on the time and type of train.

R: I'd like to know the times of trains from Paris to Lille
S: From Paris to Lille, what day are you traveling?
R: at 5pm
S: Please give your travel date, for example June 4th.

Figure 1: Dialog excerpt. **S** is the system response and **R** the recognized string.

semantic frame output by literal understanding in the context of the ongoing dialog.
2- Switching: The dialog manager decides if there is a change in task (navigation) or an error (an inconsistency between the result of contextual understanding and the dialog histories).
3(a) Continuation: If there is no change in task or error detected the ongoing task is pursued (item acquisition or database access). *(b) Change of task:* The ongoing task is closed and the new task activated according to the task model. A back trace to the previous task is kept so that the system can return. A frequent example is the request for information about reduction types while completing a reservation task. *(c) Error recovery:* If an error is detected, the error recovery strategy depends on dialog phase (acquisition or post-acquisition), the current task, the dialog state and histories. The performance of the speech recognizer can be accounted for to treat frequent or systematic errors (such as confusions between "non" and "Nantes"). In all cases the detection and the treatment of errors is based on a context larger than the previous query/response pair, with the criteria of minimizing the number of transactions.

If conflicting information is detected during the acquisition phase, the new information can either replace the old information or can be verified. The top of Figure 2 illustrates an example were the user first asks to go to Lyon, and then changes the request to go from Lyon to Lille. The system detects the conflict and asks the user to respecify the cities. In the negotiation phase of the dialog (ie. the departure and arrival cities and travel date have been confirmed), only the time and additional constraints are the subject of negotiation. An example where the dialog manager ignores conflicting information is shown in Figure 2(bottom). In the context of a reduction type, the city "Nantes" is considered spurious. When contradictory information appears within a single query, the last information is given the priority, unless there is supporting information in the dialog history or strong lexical markers for a previous item. There are two conditions when a constrained dialog is initiated: if there is difficulty in obtaining one of the basic items (departure/arrival city, travel date); or if there is an incoherence between what the system already believes and a new information item. This incoherence can be due to an old error that was not detected or a new one.

Natural Language Generator

The dialog manager generates a dialog act in the form of a frame dependent on the dialog state, the dialog and generation histories and the information returned from the database. The dialog state is used to generate an appropriate response to the speaker's query. According to the ergonomic choices, extensive use is made of implicit confirmation, trying to provide the user with direct feedback

R: I'm going to Lyon
S: **To Lyon**, where are you leaving from?
R: I'm going from Lyon to Lille
S: I'm sorry, I don't understand. Please say your departure city, for example, Paris.
R: Lyon
S: From Lyon, what is your destination?

S: On Wed 18/11 there is a direct train leaving Paris at 9:16 am and arriving in Lille at 10:14 am. Is this ok? Do you want to reserve?
R: I'd like to know the price
S: Are you eligible for a reduction?
R: yes from Nantes
S: What reduction do you have?

Figure 2: Examples of error detection and recovery.

U: Paris Lille, February 17th at 9am
S: ***Wednesday, 17/2/99*** there is a train from ***Paris-Nord to Lille-Flandres*** at 9:10am, arriving at 10:20am. Is this ok?
U1: that's good, I'll reserve
U2: Uh, no, I'd like to leave earlier
S: ***The previous train*** leaves at...

Figure 3: Dialog excerpt illustrating negotiation. In U1 the user accepts the system's proposal, in U2 the request is modified.

of what the system has understood. To do so, the returned information is limited to that which is new or of high relevance. For example, if a caller asks for an earlier train, only this information is returned, so the user does not have to find the desired information among other information (see the last system response in Figure 3). This allows the caller to rapidly check and correct the system if necessary.

When the dialog manager detects that the dialog is going astray, more directive prompts reflect the progressively constrained dialog. At the first level an example is provided in the generated message (see Figures 1 and 2). If problems persist in obtaining a city name, the caller is asked to repeat the city and optionally spell it. *Specify your departure city and spell it if you want to, for example, Paris P A R I S.*

Negotiation allows the user to refuse a solution proposed by the system and to modify his/her request (specify additional constraints, remove or change constraints). Negotiation is common in human-human dialogs and leads to a more natural information retrieval dialogs. The natural language component generates responses which let the user know that negotiation is possible as shown in Figure 3.

EVALUATION

A corpus of over 3700 calls was used to develop, refine and test the dialog strategies. The system was also periodically evaluated by the French Railways, a partner in ARISE. In November'98 the SNCF carried out the final performance assessment within the context of the ARISE project. Subjects were recruited via a market survey company. The overall dialog error rate in obtaining timetable information on the 163 calls was 21.5%. If calls where the user made no attempt to correct the system when the system gave incorrect information are eliminated, the dialog error rate on the remaining 151 calls was 15.3%. Improvements to the speech recognition, understanding and dialog com-

Data Set	*Fare*	*Reserve*	*Service*	*Reduction*
May98 (50 dialogs)	37/25	67/55	38/37	26/24
Oct98 (59 dialogs)	53/48	61/57	24/22	41/32

Table 1: Detection of changes in task. Number of user attempts/number of task changes.

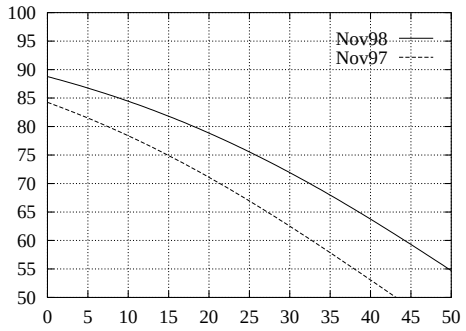

Figure 4: Dialog success as a function of word error rate.

ponents developed have reduced the dialog error rate from 47% in Nov97 to about 15% in Nov98[5].

Table 1 shows the number of user queries with a change in task and the number of times a task change resulted for fares, reservation, services and reductions measured on 109 dialogs. The system correctly detected 65% of fare requests in May98 which improved to 90% in Oct'98. The detection of reservation requests also rose from 82% to 93%. Requests for services are relatively easy to detect (97% and 93% correct). In contrast there is a noticable decrease in the detection of reduction requests, about half are due to new query formulations and half results from recognition errors. We analyzed in detail the task changes for the Oct98 calls. Although a navigation error occurred on 31 of the 59 calls, 22 were successfully recovered. The two main problems observed were incorrect switch to the "information reduction" task (25% of the errors); and longer interactions when dialog manager did not detect the end of the reservation task (19% of the errors).

A measure of the error recovery is shown in Figure 4 which plots dialog success rate as a function of word error rate for the Nov'97 and Nov'98 test calls. For word error rates under 15%, the dialog success rate has been improved by about 5%. For higher word error rates, a larger improvement is seen, at the cost of longer dialogs. Part of the difference is due to the rejection of low scoring words, which tends to lead to longer but more successful dialogs.

DISCUSSION

Many strategic decisions are taken in the design of a spoken language dialog system. Some are based on ergonomic or technical constraints, others are based on the designer's experience and intuition. Decisions taken for system development have an impact on the implemented dialog strategies. An example of the such a strategic choice concerns system functionality, where the designer can choose to develop and test individual aspects one at a time, or can implement and test a complete system which includes all required functionalities. Another example concerns how to treat speech recognition errors. When developing an SLDS it is common practice to bootstrap the speech recognizer with acoustic and language models trained on data from other tasks. This in turn implies that the word error of the recognizer will be higher than what can be expected with the final system after data collection and model adaptation. The system designer is thus faced with the choice of how to deal with this hopefully temporarily high recognition error. One choice is to simply accept the errors and to do nothing in particular to deal with them. This tends to lead to longer dialogs if the user notices the error and tries to correct the system; to user frustration if the system just doesn't seem to understand; and/or to higher dialog failure rates if the user does not or is not able to correct the system. Another choice is to try to detect potentially erroneous words, and ignore these, thus avoiding the propagation of errors. In the second case, the system can appear to not "hear" what was said, asking the user to repeat the information.

Even given perfect speech recognition, the understanding component must be able to interpret a wide range of spontaneous queries, from simple, clean ones "I want to go from Paris to Lyon tomorrow morning" to "I want to go to Paris, I live in Dijon" or "It doesn't really matter which day I travel" and "information about reserving a sleeping cars about prices" and "uh yes another information that is to say I'd like to return from uh Nice to Paris December 9th it's a Wedndesday also I think the evening around uh well leaving I'd like to have 6pm or 7pm". These are translations of queries recorded with the LIMSI ARISE system. Dealing with such a wide range of speaking styles requires a context larger then the current query. Compared with the RAILTEL system, the literal understanding component [5] in ARISE makes fewer decisions (only when strong markers are present), allowing the dialog manager to carry out the contextual interpretation.

It is important to note that the system is unable to differentiate problems arising from the recognizer from errors on the part of the user, therefore these are treated in the same manner. Handling of errors is complicated by the fact that a new error can occur while in the process of correcting an earlier one.

ACKNOWLEDGEMENTS

We gratefully acknowledge the contributions of colleagues at LIMSI, the SNCF and the VECSYS company to this work.

REFERENCES

[1] J.L. Gauvain et al., "Spoken Language component of the MASK Kiosk" in K. Varghese, S. Pfleger(Eds.) "Human Comfort and security of information systems", Springer-Verlag, pp. 93-103, 1997.

[2] L. Devillers, H. Bonneau-Maynard, "Evaluation of Dialog Strategies for a Tourist Information Retrieval System," *ICSLP'98*, pp. 1187-1190, Sydney, Dec. 1998.

[3] H. Bonneau-Maynard, L. Devillers "Dialog Strategies in a tourist information spoken dialog system," *SPECOM'98*, St. Petersbourg, Oct. 1998.

[4] S. Bennacef et al., "Dialog in the RAILTEL Telephone-Based System," *ICSLP'96*, pp. 550-553, Philadelphia, Oct. 1996.

[5] L. Lamel et al., "The LIMSI ARISE system", *IVTTA'98* pp. 209-214, Torino, Sept. 1998.