Dialogue Management for Telephone Information Systems

Scott McGlashan*     Eric Bilange†  Norman Fraser*
Nigel Gilbert*        Paul Heisterkamp†  Nick Youd§

May 22, 1996

1 Introduction

The work described in this paper has been carried out as part of the Sundial project (Speech UNderstanding in DIALogue) which is currently one of the largest collaborative speech technology projects in Europe1. The goal of the project is to build real-time integrated computer systems capable of maintaining co-operative dialogues with users over standard telephone lines. Systems have been developed for four languages – French, German, Italian and English – within the task domains of flight reservations and enquiries and train enquiries.

In order to maintain spoken dialogues with users, each system carries out three principal functions: the interpretation of user utterances; the generation of system utterances; and management of the dialogue so that system utterances are

*Social and Computer Sciences Research Group, University of Surrey, Guildford, U.K. Email: scott@soc.surrey.ac.uk; norman@soc.sussey.ac.uk; gng@soc.surrey.ac.uk
†Cap Gemini Innovation, 118 rue de Tocqueville, 5017 Paris, France. Email: bilange@capsogeti.fr
‡Daimler-Benz AG, Forschungsinstitut Ulm, Wilhelm-Runge-Strasse 11, D-7900 Ulm, Germany.
§Logica Cambridge Ltd, Betjeman House, 104 Hills Road, Cambridge, U.K. Email: nick@logcam.ucc

1The Sundial project (P2218) is partly funded by the Commission of the European Communities under the ESPRIT programme and involves partners from France (CNET, CAP GEMINI INNOVATION and IRISA – University of Rennes), Germany (Daimler-Benz, Siemens and the University of Erlangen), Italy (CSELT, Saritel and Politecnico di Torino) and the UK (Logica and the University of Surrey).
natural and coherent in light of user utterances. Each system has been designed so that these functions are distributed over five modules\(^2\). Linguistic interpretation is dealt with by two modules: an acoustic front end module which inputs acoustic signals and outputs word or phoneme lattices; and a linguistic processing module which extracts a plausible string in order to provide syntactic and semantic representations of the utterance. The dialogue manager module takes each linguistic representation and gives it an interpretation within the dialogue context. Using this interpretation, it then decides how the dialogue might continue and, if it is the system’s turn to speak, plans a schematic linguistic representation for the system utterance. Generation of the system utterance is carried out by the message generation module, which produces a detailed linguistic representation, and the speech synthesis module, which synthesizes the representation for telephone output.

In this paper, we describe the Sundial approach to dialogue management and illustrate it with a detailed example\(^3\). Two principal problems in dialogue management have been addressed.

The first is that dialogue management needs to be \textit{generic}: i.e. the dialogue manager module must be able to interpret and generate utterances in more than one language and across more than one task domain. With generic dialogue management, a task- and language-independent dialogue manager is designed and configured for integration into systems which vary in language and task domain.

The second problem is that dialogue management must provide \textit{co-operative interaction} with the user. To achieve this, the system needs to produce utterances which are perceived by the user as natural, coherent and helpful within the context of the dialogue. Part of this context is the purpose and modality of the dialogue: in our case, task-driven speech-based dialogues. A number of specific problems present themselves in task-driven dialogues; for example, tasks which are outwith the task domain, tasks which need to be clarified by the user and tasks about which the user has given incorrect information. For each of these

\(^2\)See Peckham (in press) for a detailed overview of the Sundial system.

\(^3\)We are grateful to Wieland Eckert, Mark Guyomard, Mouloud Kharoune, Jean-Yves Magadur, Jacques Siroux, Jutta Unglaub and Robin Woodliff for their invaluable contributions.
problems, the system should provide a response which does not lead to dialogue failure: while the goal of the interaction – providing the requested information – may fail, the dialogue *per se* should not.

Speech brings special problems for co-operative interaction, especially continuous speech from untrained users over conventional telephone lines. Although speech technology is advancing quickly, the number of acoustic problems is likely to remain high in the short term. Dialogue management must tackle problems of poor recognition so that the dialogue is successfully maintained (cf. Young and Proctor 1989).

The Sundial solution to these problems is embodied in the **distributed database architecture** of the dialogue manager. This design has an empirical as well as theoretical basis (cf. Luzzati and Neel 1989). Empirical data come from analysis of two types of corpora of dialogues within the task domains: dialogues between human agents and human users; and dialogues between computer agents and human users, where the computer agents were simulated using the ‘Wizard of Oz’ technique (cf. Fraser and Gilbert 1991; Ponamale et al. 1990). A central feature of the design is that the dialogue manager constructs and maintains a model of the interaction. The system is obliged to report some states and goals associated with this model to the user. The user is thus able to verify the system’s model against their own interactional model (cf. Frankish 1989). If verification fails, the user has an opportunity to make the problem explicit and so correct it. The advantage of this approach is that it may forstall more serious problems which lead to irreparable breakdowns in the dialogue. Moreover, the architecture is modular: the interactional model, and associated dialogue management functions, are distributed across a number of semi-autonomous modules. With a distributed database architecture, both portability across languages and task domains, as well as the treatment of co-operation, are supported in a principled manner.

## 2 Design

Within the Sundial system, the dialogue manager is responsible for providing a co-operative interaction with the user (cf. Gerlach and Horacek 1989). From the
user’s point of view, whether the interaction is co-operative or not is judged solely on the basis of what the system says in the dialogue. It is therefore imperative that system utterances are appropriate and helpful to the user in the context of the dialogue. To achieve this, both interpretation of user utterances and selection and planning of system utterances must be informed by the past and current states of the interaction. Co-operative dialogue management, therefore, requires the construction and maintenance of an interactional model: i.e. a model which specifies the layers of structure which can be distinguished in dialogues. Following Grosz and Sidner (1986) we distinguish linguistic structure, attentional, or belief structure, and intentional structure. Intentional structure is further differentiated into dialogue structure and task structure (cf. Bunt (1989) who distinguishes communicative and meta-communicative acts in information dialogues). As a result, four structural layers have been identified: linguistic, belief, dialogue and task structure.

One approach to modelling dialogue interaction involves the construction of a unitary model. This model represents different layers of structure and content in the dialogue as a single structured representation. The advantage of this approach is that it captures both the distinctiveness of these structural layers and their interrelatedness. The disadvantage, however, is that such an approach does not lend itself to the development of a generic dialogue manager since portability is not easily supported: changing the task or linguistic structure requires that every unitary representation be changed.

The alternative approach, the adopted in the Sundial dialogue manager, is to use a distributed model of the interaction. Each structural layer of the dialogue is represented in a separate (sub-)model: i.e. a linguistic model, a belief model, a task model and a dialogue model (cf. Sabah 1990; Siroux et al. 1984). In addition to representing clearly the distinct layers of structure in dialogue, this approach also supports portability in a principled manner – the structures can be replaced or re-configured without directly affecting other models.

Before an overview of this architecture is presented, an indication of the nature of the sub-models can be gleaned from an examination of the representation of
belief.

2.1 Representing Belief

One influential approach to the problem of interpreting natural language input in dialogue, whilst maintaining a thread of cohesion, has been to maximise the role played by a user’s intentionality, by using plan inference to guide utterance interpretation (cf. Allen 1983; Litman and Allen 1984; Carberry 1989; Lochbaum et al. 1990). Such an approach has drawbacks however, relying as it does on a considerable amount of defeasible inference, the knowledge and rules underlying which often bear only an arbitrary relation to linguistic forms (cf. Suchman 1987; Chapman 1987). Moreover there is often a lack of symmetry, with the system reasoning about a user’s intentions, but not its own (cf. Gilbert 1987).

An alternative approach, pursued in our work, is to capitalise on the capability of cooperating agents jointly to evolve a common model of the discourse situation:

- taking advantage of the inherently interactive and collaborative nature of dialogue to reduce the complexity of inferences required for truly co-ordinated understanding in everyday conversation. (Garrod and Anderson 1987)

Such models are falsification definite since agents assume rather than infer that their belief model corresponds to the model used by other agents. This places a natural obligation on agents to make explicit the state of their belief model so as to give other agents the opportunity to correct or modify it. The main effect of this is that the effort for each agent is minimized, due to the presence of interactional conventions which obviate the need for complex plan inferences.

These conventions govern the coherence and consistency of the evolving common discourse model, according to two basic principles. The default is that interpretation and production of utterances are guided by the relevant local state of the assumed model—with utterances representing instructions to navigate or extend this. However, an agent has the possibility of overruling these local conventions by taking the initiative. In cases such as explicit contradiction or topic shifts, linguistic marking usually serves to make the agent’s intentions clear. A cooperative
interlocutor will generally follow by taking up again the default convention.

An approach to utterance interpretation based on these principles does not deny intentionality, but leaves it implicit. Underlying the progress of agents in navigating and extending the common discourse model, one could if desired infer the intentions of agents to visit first this concept, then that. But this is not necessary. Nonetheless, interpretation is task-oriented (cf. Grosz and Sidner 1986), due to tasks being represented as bona fide concepts along with the rest, and because navigation, or focussing conventions, are largely task-based. It may occasionally happen that the conventions for shared model building expounded here are insufficient, and recourse has to be made to inferred intentions. We have not found, however, that this occurs sufficiently often in our analysis of dialogues to warrant being elevated into a major principle of interpretation.

3 The Distributed Database Architecture

The Sundial dialogue manager builds and maintains a model of interactions. This model, together with dialogue management functions, is distributed across five functional modules: the linguistic interface module, the belief module, the task module, the dialogue module and the message planning module. Each module is composed of the relevant sub-model of the interaction and a set of rules. With the exception of the message planning module, each of these models has a symmetrical representation of the user and the system, falsification definiteness, and uses interactional conventions in preference to complex inference\(^4\). In addition, the models represent partial rather than total information (cf. Bunt 1989). As partial models, they are associated with a number of update rules which determine how the model is extended through system and user utterances. For the modules to carry out their functions, they need to collaborate with each other. For example, update rules may depend upon more than the current representation plus the current state of the model: they may depend upon the local interactional context and this context is represented as the current states of models in other

\(^4\)The message planning module lacks user-system symmetry for obvious reasons.
modules. Modules then need to give to, or request from, other modules information about their portion of the interactional model. To achieve this, each module has a **bi-directional interface**. Bi-directional interfaces permit not only one module to pass information to other modules, but they also permit other modules to send requests to, and receive replies from, that module. For the latter type of message passing, modules need to be interactive: i.e. capable of suspending their current processing state and re-entering it when the request has been satisfied.

In this way, modules in the distributed database architecture are semi-independent. They are independent to the extent that they alone are responsible for carrying out a particular function of dialogue management including maintaining part of the interactional model. However, they are dependent upon each other to the extent that such functions may be dependent upon information in parts of the interactional model which they are not responsible for maintaining. Apart from such cases, both the functionality and the model of each module are not visible to the other modules. As a result, this architecture facilitates portability and the development of a generic dialogue manager.

What follows is a brief description of the functionality of these modules as well as their interactions, especially those between the dialogue, task and belief modules.

### 3.1 Task Module

The task module is responsible for maintaining a model of the task structure of the dialogue and consulting domain-specific application databases. The task structures for dialogues within our task domains have been established on the basis of empirical analysis of corpora. With flight reservations, three main phases in the dialogue have been identified: a ‘request formulation’ phase in which the user is invited to request task information; a ‘request resolution’ phase in which the system confirms and clarifies details of the flight, such as the date and place of departure; and a ‘reservation’ phase in which reservation details, such as whether the ticket is a single or return, the class and the name and telephone number of the passenger(s), are requested and confirmed.
In terms of interaction between modules, the task module informs the dialogue module of the current state of the task. Typically, this involves deciding whether sufficient task information has been provided by the user and, if not, what additional parameters are required satisfactorily to resolve the reservation request (cf. Sadek 1990). Less typical, but no less important for co-operative dialogue, are interactions which arise when a user provides sufficient, but incorrect, information in the request resolution phase: for example, there is no flight to the stated destination city. This problem is detected in the reservation phase of the dialogue when the task module fails to obtain a reservation from the application database\(^5\). Rather than simply informing the dialogue module that the reservation task has failed, the task module attempts to relax the given task parameters and suggest alternative values for them (cf. Guyomard and Siroux 1988). For example, if there are no flights available in the morning of the given departure day, flights in the afternoon may be suggested. In such cases, system utterances are selected and formulated so as to make explicit to the user the fact that an alternative reservation is being offered. Where the suggested alternatives do not satisfy the user, the user is invited to further clarify or modify the request. If this fails, the enquiry is re-directed to a human operator.

### 3.2 Belief Module

This module has the purpose of building and maintaining a representation of the common belief model which contains not just concepts created directly as a result of user’s utterances, but also any assumed initial common ground, together with inferential extensions. Such a representation is used to derive a contextual interpretation of utterances, locating concepts corresponding to referring expressions, anchoring elliptical utterances in context, and deriving task-relevant interpretations. Extending the belief model draws on two complementary principles: on the one hand, existing structures may be visited or added to, using the normal incremental techniques; on the other hand, where there is a potential mismatch

\(^5\)Some inconsistency may be detected earlier than this. For example, non-task specific inconsistencies, such as the departure and arrival locations being identical, can be detected by the belief module in the request resolution phase.
between the knowledge of different agents, or where one agent entertains alternatives, perspectives may be used to separate out the various representations. At a later stage, once an irrevocable choice of perspectives has been made by both parties, their contents may be merged.

In order for the correct context for interpretation to be established, the belief module cooperates with the dialogue module, firstly, by the dialogue module informing it of possible contexts, or anchor points, for an utterance; and secondly, by the belief module informing the dialogue module of the resulting status of the interpretation process. For example, knowing whether an utterance simply repeats information will guide the dialogue module in arriving at a correct interpretation, assigning say the label “confirmation-request” under a particular context.

The belief module additionally cooperates with the message output subsystem, in order for appropriate referring expressions to be planned and generated, and with an aim of assigning the correct prosodic focal status to constituents of an utterance. This latter task is essential for the production of appropriate intonational accents in the synthesis of voice output (cf. Hirschberg and Pierrehumbert 1986).

### 3.3 Dialogue Module

The dialogue module is responsible for maintaining a model of dialogue context, building an interpretation of user utterances and determining how the dialogue should continue. Dialogue structure is based on the linguistic analyses of Roulet and Moeschler (cf. Moeschler 1989) in which dialogue is hierarchically structured into exchanges, interventions and dialogue acts (see Wachtel (1986) and Ferrari and Reilly (1986) for complementary approaches). In order to determine the appropriate contextual interpretation of user utterances, the dialogue module interacts with the belief module which supplies it with a referential interpretation of the utterance. For example, if the system initiated an exchange to determine the departure date and the user reacts by providing the appropriate referential information, this exchange is (provisionally) closed. The dialogue module then constructs a set of dialogue allowances: i.e. a set of possible continuations given the current state of the dialogue. These allowances are principally con-
structured on the basis of the effect user utterances have on the dialogue, task and belief models. For example, if the task module requires further task parameters, the dialogue module may plan a dialogue allowance in which these parameters are requested from the user. In addition, the belief module may inform the dialogue module that the acoustic recognition score associated with the user utterance is sufficiently poor for it to plan an allowance confirming the parameters. Whether one, or both, of these allowances is realized in the next system utterance is, in part, determined by the dialogue strategy.

The purpose of the dialogue strategy is to modulate the output of the system in accordance with the progress of the dialogue so as to minimize the risk of breakdown (cf. Luzzati 1989). Two key factors in selecting allowances for realization are the acoustic scores associated with user utterances and the degree of embeddedness in exchanges. For example, if the acoustic score for a task parameter provided by the user is high (i.e. a high probability of correct recognition), then the next system move may merge confirmation of the given parameter with a request for another parameter. However, if the score is low, then the system move will consist solely of a request for confirmation of the parameter: the request for further information is held over until confirmation has been obtained. In addition, as the exchanges become more embedded, as in the case of repeated recognition failures, system utterances narrow the range of preferred responses from the user by means of ‘menu’ style interaction, requests for reformulation or the spelling of parameters. Using a fine-grained and flexible dialogue strategy offers a partial, but principled solution, to the problem of maintaining a co-operative interaction with users. When the dialogue is progressing smoothly, system output permits a wide range of responses from the user. As the dialogue becomes more difficult, the system begins to exercise more control over the range of user responses.

3.4 Linguistic Interface Module

The linguistic interface module interfaces the dialogue manager with the linguistic processing module. It is also responsible for maintaining a model of the linguistic structure of system and user turns. In addition, it constructs predictions for the
linguistic processing module when sent an ordered set of dialogue allowances by
the dialogue module (cf. Young et al. 1989; Guyomard et al. 1990).

3.5 Message Planning Module

The message planning module interfaces the dialogue manager with the message
generation module. When it receives a dialogue allowance from the dialogue mod-
ule, the allowance is augmented with rhetorical and semantic structure necessary
for the generation of system messages requesting, if necessary, information from
the belief and linguistic interface modules (cf. Hovy 1988). When the utterance
has been generated, it sends the appropriate updates to the dialogue, belief and
linguistic interface modules.

4 An Illustrative Example

A simple example, shown in Figure 1, illustrates how the Sundial dialogue manager
actually proceeds during a dialogue.

| S₁ | Hello. Sundial reservation system. Can I help you? |
| U₁ | I’d like a ticket from London to Paris |
| S₂ | London to Paris. When do you want to travel? |
| U₂ | November the 17th |
| S₃ | Did you say DECEMBER the 17th? |
| U₃ | No, November the 17th |
| S₄ | ok, November the 17th. At what time? |
| U₄ | at 10. |

Figure 1: A dialogue extract

This dialogue extract, with a misunderstanding between the user and the sys-

tem (about the departure date), illustrates both interaction between modules and
how the dialogue strategy modulates system output. We will not describe each
step in detail, due to space limitations, but will focus our attention on U₂ inter-
pretation and S₃ generation.

When uttering S₂ the system knows the following information:
• A task has been established (derived from $U_1$).

• The belief module informs the dialogue module that the departure place and the arrival place received average recognition scores.

• The task module informs the dialogue module that two additional parameters, the departure date and time, need to be requested from the user since they are necessary for resolving the request.

At this stage, the dialogue strategy permits two dialogue allowances to be realized in $S_2$: the first part implicitly seeks confirmation of two averagely scored parameters; and the second is a request for information about the departure time. The state of the system at this stage is shown in Figure 2.

| Module       | Current Goals                                      |
|--------------|----------------------------------------------------|
| TASK MODULE  | Waiting for departure date and time values         |
| BELIEF MODULE| nothing                                             |
| DIALOGUE MODULE | • Trying to confirm departure and arrival places   |
|               | • Trying to obtain the departure date parameter    |
|               | • waiting for a dialogue opportunity to ask the    |
|               | departure time parameter.                           |

Figure 2: System status after $S_2$

Once the user utters $U_2$, dialogue interpretation proceeds as follows:

1. The dialogue module requests the belief module to interpret $U_2$ under the following possible contexts:

   (a) departure city,

   (b) arrival city,

   (c) departure date.

These contexts are established according to the local representation of the dialogue by the dialogue module (cf. Bilange et al. 1990).
2. The belief module replies that there is a possible interpretation under the third context. This results in assigning the (misheard) value ‘December the 17th’ to the departure date parameter. Furthermore, the belief module informs the dialogue module that this value is associated with a poor recognition score.

3. The dialogue module integrates this information about $U_2$ from the belief module into the dialogue model. It then derives the following implicatures:

(a) The user accepts the question about the date. Therefore, an evaluation of the answer is possible. The value is sent to the task module.

(b) Since the question is accepted, there is an implicit acceptance of the previous evaluation – i.e. confirmation of the departure and arrival cities. Therefore, the departure city and arrival city parameters are implicitly confirmed. These two confirmations are sent to the belief module.

4. The dialogue module plans two possible continuations: the first consists of an evaluation of the departure date; and the second an evaluation plus a request for the departure time (a postponed task goal). The first continuation is selected for generation since the dialogue strategy requires that parameters with low scores are explicitly, and individually, evaluated by the user. In this case, planning and generation of this dialogue allowance results in a question ($S_3$) thereby making clear to the user that clarification is being sought by the system. Thus the topic of the conversation is now unique and the system can strongly expect a positive reaction (e.g. “that’s it”) or a correction as in $U_3$.

5 **Implementation**

The Sundial dialogue manager has been implemented in Quintus Prolog and tested on a variety of different hardware platforms. It has been integrated with the rest
of the Sundial system and successfully manages a range of realistic telephone-based spoken dialogues. So far, the dialogue manager has been used to manage dialogues in English, French, and German relating to the tasks of flight enquiries, flight reservations, and train timetable enquiries. Language and task can be changed in the dialogue manager simply by resetting switches which govern the choice of static knowledge bases to consult during dialogue management.

Current average response times for the entire Sundial system (including speech recognition and synthesis) are around 10 seconds, with the dialogue manager taking 1-2 seconds. However, a significant amount of optimization has still to be carried out, and it is expected that response times will approach real time within the next six months. The lexicon currently contains around 300 distinct words.

6 Conclusion

Two problems facing dialogue management in telephone information systems – co-operative interaction and portability across languages and task domains – have been addressed by the dialogue manager in the Sundial system. Our solution has been to distribute the interactional model, as well as functions dealing with co-operative interaction, across a number of semi-independent software modules. Rather than engaging in detailed and computationally expensive modelling of a user’s intentionality, a strategy of offering the user a local part of the system’s distributed interactional model for acceptance or rejection has been employed. Each sub-module is an expert on a local part of the interactional model. The language of spoken interaction and the application task can both be changed by instructing the relevant sub-modules to consult different knowledge bases. In this way the generic Sundial dialogue manager can be customized for specific languages and applications.

References

Allen, J. F. (1983). Recognising intentions from natural language utterances. In M. Brady and R. Berwick, editors, *Computational models of discourse*, pages
107 – 166. MIT.

Bilange, E., M. Guyomard, and J. Siroux (1990). Separating dialogue knowledge from task knowledge for oral dialogue management. In COGNITIVA ’90. Madrid.

Bunt, H. C. (1989). Towards a dynamic interpretation theory of utterances in dialogue. In B. A. Elsendoorn and H. Bouma, editors, Working models of human perception. Academic Press.

Carberry, S. (1989). A pragmatics-based approach to ellipsis resolution. *Computational Linguistics*, 15(2).

Chapman, D. (1987). Planning for conjunctive goals. *Artificial Intelligence*, 32: 333–377.

Ferrari, G. and R. Reilly (1986). A two-level dialogue representation. In COLING ’86. Bonn.

Frankish, C. R. (1989). Conversations with computers: problems of feedback and error correction. In Eurespeech ’89. Paris.

Fraser, N. M. and G. N. Gilbert (1991). Simulating speech systems. *Computer Speech and Language*, 5.

Garrod, S. and A. Anderson (1987). Saying what you mean in dialogue: A study in conceptual and semantic co-ordination. *Cognition*, 27: 181–218.

Gerlach, M. and H. Horacek (1989). Dialog control in a natural language system. In *Proceedings of the 4th Conference of the European Chapter of the Association for Computational Linguistics*. Manchester.

Gilbert, G. N. (1987). Cognitive and social models of the user. In H. J. Bullinger and B. Shackel, editors, *Human-computer interaction - Interact ’87*. North Holland, Amsterdam. Proceedings of the second IFIP conference, Stuttgart.

Grosz, B. J. and C. L. Sidner (1986). Attentions, intentions, and the structure of discourse. *Computational Linguistics*, 12(3): 175–204.
Guyomard, M. and J. Siroux (1988). An approach to co-operation in man machine oral dialogue. In *ERGO-IA*.

Guyomard, M., J. Siroux, and A. Cozannet (1990). Le rôle du dialogue pour la reconnaissance de la parole. Le cas du système des Pages Jaunes. In *Proceedings of 18th JEP, Montreal*.

Hirschberg, J. and J. Pierrehumbert (1986). The intonational structuring of discourse. In *Proceedings of the ACL*.

Hovy, E. H. (1988). *Generating Natural Language Under Pragmatic Constraints*. Lawrence Erlbaum Associates.

Litman, D. J. and J. F. Allen (1984). A plan recognition model for subdialogues in conversations. Technical Report TR 141, University of Rochester.

Lochbaum, K. E., B. J. Grosz, and C. L. Sidner (1990). Models of plans to support communication: An initial report. In *AAAI'90*. Boston.

Luzzati, D. (1989). A dynamic dialog model for human machine communication. In *Eurospeech '89*. Paris.

Luzzati, D. and F. Neel (1989). Dialogue behaviour induced by machine. In *Eurospeech '89*. Paris.

Moeschler, J. (1989). *Modélisation du dialogue, représentation de l’inférence argumentative*. Hermes.

Peckham, J. (in press). Speech understanding and dialogue over the telephone: an overview of the Sundial project. *Acoustic Bulletin*.

Ponamale, M., E. Bilange, K. Choukri, and S. Soudoplatoff (1990). A computer-aided approach to the design of an oral dialogue system. In *Proceedings of Eastern Multiconference, Nashville*.

Sabah, G. (1990). A model for interaction between cognitive processes. In *COLING'90*. Helsinki.
Sadek, M. D. (1990). Logical task modelling for man-machine dialogue. In *AAAI’90*. Boston.

Siroux, J., M. Guyomard, and L. Trilling (1984). Le dialogueur un intermédiaire entre l’utilisateur et l’application. In *Proceedings of Dialogue homme machine a composante orale*. Organised by GRECO.

Suchman, L. A. (1987). *Plans and Situated Action*. Cambridge University Press, Cambridge.

Wachtel, T. (1986). Pragmatic sensitivity in NL interfaces and the structure of conversation. In *COLING ’86*. Bonn.

Young, S. J. and C. E. Proctor (1989). The design and implementation of dialogue control in voice operated database inquiry systems. *Computer Speech and Language*, 3: 329–353.

Young, S. R., W. H. Ward, and A. G. Hauptmann (1989). Layering predictions: flexible use of dialog expectation in speech recognition. In *Proceedings of IJCAI ’89, Detroit.*