Cover page

A Methodology for Diagnostic Evaluation of Spoken Human-Machine Dialogue

Running head: Diagnostic Evaluation of Dialogue

Laila Dybkjær*, Niels Ole Bernsen* and Hans Dybkjær**

*The Maersk Mc-Kinney Moller Institute for Production Technology
Odense University, Campusvej 55, 5230 Odense M, Denmark
emails: laila@mip.ou.dk, nob@mip.ou.dk
phone: (+45) 65 57 35 44 fax: (+45) 66 15 76 97

**Prolog Development Center A/S
H. J. Holst Vej 3-5A, 2605 Brøndby, Denmark
email: dybkjaer@pdc.dk
phone: (+45) 36 72 10 22 fax: (+45) 36 72 02 69

Contact person:

Laila Dybkjær
The Maersk Mc-Kinney Moller Institute for Production Technology

Odense University, Campusvej 55,
DK-5230 Odense M, Denmark
email: laila@mip.ou.dk
phone: (+45) 65 57 35 53
fax: (+45) 66 15 76 97

A Methodology for Diagnostic Evaluation of Spoken Human-Machine Dialogue

Laila Dybkjær*, Niels Ole Bernsen* and Hans Dybkjær**

*The Maersk Mc-Kinney Moller Institute for Production Technology

Odense University, Campusvej 55, 5230 Odense M, Denmark

emails: laila@mip.ou.dk, nob@mip.ou.dk

phone: (+45) 65 57 35 44 fax: (+45) 66 15 76 97

**Prolog Development Center A/S

H. J. Holst Vej 3-5A, 2605 Brøndby, Denmark

email: dybkjaer@pdc.dk

phone: (+45) 36 72 10 22 fax: (+45) 36 72 02 69

Summary

Diagnostic evaluation is an important instrument for the development of high quality spoken language dialogue systems. Yet no rigorous methodology exists for the systematic and exhaustive diagnostic evaluation of all aspects of spoken language interaction: recognition, synthesis, grammar, vocabulary, dialogue etc. The paper addresses part of this problem by presenting a methodology for the diagnostic evaluation of spoken human-machine dialogue. The first part of the me-

thodology is general and supports the detection of any kind of user-system interaction problem. The second part concerns the classification, diagnosis and repair of problems of dialogue interaction. These problems are of two broad kinds: dialogue design errors and user errors. Use of the methodology is demonstrated through analysis of a corpus from the user test of the Danish Dialogue System.

1 Introduction

Diagnostic evaluation of problems of user-system interaction is a major concern in spoken language dialogue systems (SLDSs) development. The process of diagnostic evaluation focuses on the detection, classification, diagnostic analysis and repair of recognition problems, linguistic problems, dialogue interaction problems and any other kind of problem that may affect user-system interaction. When problems have been properly repaired, they are prevented from occurring in future user interactions with the system. However, today's diagnostic evaluation methods primarily build on system developers' craft skills and inadequate ad hoc methods. There is an evident lack of a rigorous methodology in support of systematic and exhaustive diagnostic evaluation.

This paper presents and illustrates a methodology for diagnostic evaluation of user-system interaction in SLDSs, focusing on problems of dialogue interaction. The methodology has two parts. The first part helps system developers detect any kind of user-system interaction problem. The second part supports the classification, diagnosis and repair of dialogue interaction problems.

In diagnostic evaluation of dialogue interaction, the natural focus is on dialogue design errors. Dialogue design errors tend to make user task performance unnecessarily bumpy and generate user dissatisfaction with SLDS technologies. It is the system developer's task to detect and diagnose such errors so that they can be repaired in order to prevent human-machine miscommunication from seriously damaging the user's task performance. Some forms of miscommunication cannot be prevented from occurring and must be handled on-line through meta-communication. *Meta-communication* is communication about the user-system communication itself and is usually being undertaken for purposes of clarification or repair. Meta-communication may be contrasted with *domain communication* which is communication within or about the task domain. In particular the speech recognition capabilities of SLDSs are still fragile and user-initiated repair meta-communication functionality is therefore needed to overcome the effects of system misrecognitions. In addition, users will inevitably provide input which, although recognised and understood by the system, requires system-initiated clarification meta-communication dialogue. It is important to bear in mind, however, that, given current state-of-the-art in speech and language technologies, the possibilities of on-line handling of clarification and repair meta-communication are seriously limited. In particular, most user-initiated clarification meta-communication is difficult or impossible to handle by current systems. Furthermore, miscommunication, even when successfully resolved, always leads to additional user-system exchanges which delay task performance. It follows that diagnostic evaluation with the purpose of reducing the amount of miscommunication that can occur is highly important to successful dialogue design. Reduced meta-communication is a source of SLDS robustness and of increased dialogue quality, smoothness and efficiency.

We want to argue, moreover, that not everything that goes wrong in the dialogue between user and system is due solely to errors made by the dialogue developers. Users make errors, too. Also some dialogue interaction problems are the compound effects of dialogue design errors and user errors. It follows that diagnostic evaluation of dialogue interaction includes user errors within its scope. This raises the tricky issue of how to separate errors made solely by users from compound errors and from pure errors of dialogue design.

There is a rapidly growing body of work on evaluation of NLP systems in general and of dialogue interaction in particular. Overviews are Gallier and Sparck-Jones (1996) and Hirschmann and Thompson (1996). Other recent references are McRoy (1996), Calzolari and McNaught (1996), Gaizauskas (1997), Hirschberg, Kamm and Walker (1997), Gibbon, Moore and Winski (1997).

The methodology for diagnostic evaluation to be presented below was developed in the course of designing, implementing and testing the dialogue model for the Danish Dialogue System. We present the results of using the methodology in a controlled user test of the Danish Dialogue System. Section 2 provides a description of the Danish Dialogue System and the development of its dialogue model. Section 3 describes the user test of the system. In Section 4, the first part of the methodology for diagnostic evaluation is presented, i.e. the detection of interaction problems in general. The second part of the methodology, concerning the classification, diagnostic analysis and repair of dialogue interaction problems, is presented in Sections 5 and 6. Section 5 presents a typology for the classification of dialogue design errors and reports on its use for diagnostic evaluation in the user test. Section 6 discusses the classification, diagnosis and repair of the user errors found in the user test data. Section 7 concludes the paper.

2 Development of the dialogue model for the Danish Dialogue System

The Danish Dialogue System is an application-oriented prototype ticket reservation system for Danish domestic flights. The system is a walk-up-and-use application that is accessed over the telephone. It understands speaker-independent continuous spoken Danish with a vocabulary of about 500 words. The prototype runs in close-to-real-time. It is a representative example of advanced state-of-the-art systems. Comparable SLDSs are found in Aust and Örder (1995) (German train time table information), Cole, Novick, Fanty, Vermeulen, Sutton, Burnett and Schalkwyk (1994) (US Census questionnaire system), Eckert, Nöth, Niemann and Schukat-Talamazzini (1995) (German train time table information), Peckham (1993) (British Airways flight information).

The system has five main modules. The *speech recogniser* uses Hidden Markov Models (HMMs) to produce a 1-best string of words. The *parser* makes a syntactic analysis of the string and extracts the semantic contents which are represented in frame-like structures called semantic objects. The *dialogue handling module* interprets the contents of the semantic objects and decides on the next system action which may be to send a query to the *application database*, send output to the user, or wait for new input. In the latter case predictions on the next user input are sent to the recogniser and the parser. *Output* is produced by concatenating pre-recorded phrases under the control of the dialogue module.

The dialogue model for the Danish Dialogue System was developed by the Wizard of Oz (WOZ) experimental prototyping method in which a human simulates the system to be designed, cf. Fraser and Gilbert (1991). Since the application is accessed over the telephone, real-time performance

was required. In the context of the chosen hardware and software the real-time requirement gave rise to the following additional requirements: at most 100 words can be active in memory at a time; the average user utterance length should not exceed 3-4 words; and the maximum user utterance length should not exceed 10 words. The last two requirements were also intended to maintain the recogniser error rate at an acceptable level. Furthermore, because of limited project resources the system vocabulary size was set to about 500 words.

Apart from real-time performance, the main usability requirements were: sufficient task domain coverage, robustness, intuitively natural forms of language and dialogue, and dialogue flexibility. These requirements had to be traded off against the above-mentioned resource constraints and technological requirements. It was the task of the WOZ experiments to optimise the trade-offs, cf. Dybkjær, Bernsen and Dybkjær (1993). This was done iteratively through seven series of WOZ experiments. These yielded a transcribed corpus of 125 task-oriented human-machine dialogues corresponding to approximately seven hours of spoken dialogue. A total of 24 different subjects were involved in the iterations. Dialogues were based on written task descriptions (scenarios).

The WOZ-derived dialogue model which formed the basis for the implementation satisfied the resource constraints and technological requirements except that subjects' vocabularies failed to show sufficient convergence towards the end of the WOZ phase. This was no surprise, as related ATIS (Air Travel Information Systems) results from other languages suggest a minimum domain vocabulary of 1000-1200 words, cf. Peckham (1993). As regards the dialogue design goals, real-time performance appeared feasible, task domain coverage was acceptable, and the restrictions on

users' language and dialogue behaviour appeared to follow principles that would enable users to comply with them. The latter assumption had to be revised following the user test, as we shall see.

The constraint satisfaction just described was mainly achieved at the expense of dialogue flexibility. The dialogue model is mixed-initiative. Domain communication is system-directed whereas users can initiate repair meta-communication through keywords. Because of the strong limitations on active vocabulary size and user utterance length, it had been necessary to leave the domain communication initiative with the system. The system maintains its dialogue initiative by concluding all its turns by a non-open question to the user, i.e., a question which asks for a well-defined piece of information, such as a choice between binary options, a date or a time, or a destination. System-directed dialogue affords little dialogue flexibility. However, a field study established the task of reservation is a well-structured task. It includes a certain number of sub-tasks most of which must be completed in order to achieve the reservation task, and for many of these sub-tasks there is a natural order in which to carry them out. For such well-structured tasks, system-directedness may allow acceptable dialogue, at least up to a certain level of dialogue complexity, cf. Dybkjær, Bernsen and Dybkjær (1995). Adding flexibility to this structure was mainly done through incorporation of a minimal user model which allows expert users to de-select (i) the introductory instructions for novice users and (ii) the information on discount types (see S2 and S9 in Figure 1, respectively). Figure 1 shows a dialogue with the implemented reservation system.[1]

S1:	Goddag. Dette er DanLuft reservationsservice for indenrigsfly. Kender du dette system? (Hello, this is the DanLuft reservation service for domestic flights. Do you know how to use this system?)

U1: Nej. (No.)

S2: Systemet kan reservere billetter til indenrigsfly i Danmark. Du bruger det ved at besvare systemet spørgsmål. Desuden kan du bruge de to specielle kommandoer 'gentag' og 'ret' for at få gentaget eller rettet den seneste oplysning. Systemet forstår dig kun når spørgsmål besvares kort og et ad gangen. Oplys venligst kundenummer.

(The system can reserve tickets for Danish domestic flights. You use it by answering the system's questions. In addition you may use the two special commands 'repeat' and 'change' to have the most recent information repeated or changed. The system will only understand you when you answer its questions briefly and one at a time. Please state your customer number.)

U2: Nummer fire. (Number four.)

S3: Kundenummer fire. Hvor mange personer skal rejse?

(Customer number four. How many people will travel?)

U3: To personer. (Two people.)

S4: To personer. Angiv id-nummeret på den første person.

(Two people. Please state the id-number of the first person.)

U4: Id-nummer to. (Id-number two.)

S5: Angiv id-nummeret på den anden person. (Please state the id-number of the second person.)

U5: Id-nummer fem. (Id-number five.)

S6: Id-numre to og fem, Rikke Hansen og Anders Hansen. Hvor starter rejsen?

(Id-numbers two and five, Rikke Hansen and Anders Hansen. Where does the journey start?)

U6: København. (Copenhagen.)

S7: Fra København. Hvor går rejsen hen? (From Copenhagen. Where does the journey end?)

U7: Billund. (Billund.)

S8: Til Billund. Vil du reservere returrejse? (To Billund. Do you want a return ticket?)

U8: Ja tak. (Yes, please.)

[1] The WOZ experiments also covered the tasks of changing a reservation and obtaining travel information. However, as these tasks are not well-structured, they do not appear as well-suited for system-directed dialogue as does

S9: Der er mulighed for rød rabat på visse hverdagsafgange og grøn rabat i weekenden. Ved rød rabat kan hjemrejse tidligst ske efter to nætter. Er du specielt interesseret i at udnytte rabatformer?

(Red discount is available on certain weekday departures and green discount is available on week-end departures. In case of red discount the return journey may not take place until two nights after the departure. Are you particularly interested in discount?)

U9: Nej. (No.)

S10: Hvilken dato starter rejsen? (On which date will the journey start?)

U10: Den enogtredivte januar. (On the thirty-first of January.)

S11: Tirsdag den enogtredivte i første. Hvilken tid på dagen? (Tuesday, January 31. At which time?)

U11: Klokken otte. (At eight o'clock.)

S12: Klokken otte nul nul. Hvilken dato rejses retur?

(At eight o'clock. On which date will the return journey start?)

U12: Den enogtredivte januar. (On the thirty-first of January.)

S13: Tirsdag den enogtredivte i første. Hvilken tid på dagen? (Tuesday, January 31. At which time?)

U13: Klokken atten ti. (At 18:10.)

S14: Klokken atten ti. Der er nu reserveret to billetter til id-numre to og fem Rikke Hansen og Anders Hansen fra København til Billund tirsdag den enogtredivte i første klokken otte nul nul og retur tirsdag den enogtredivte i første klokken atten ti. Dit referencenummer er niogfyrre. Den samlede pris er et tusind tre hundrede og tres kroner. Vil du have billetterne tilsendt eller skal de hentes i lufthavnen?

(At 18:10. You have now booked two tickets for id-numbers two and five Rikke Hansen and Anders Hansen from Copenhagen to Billund on Tuesday January 31 at 8:00 returning on Tuesday January 31 at 18:10. Your reference number is 49. The total price is 1360 kroner. Are the tickets to be mailed or will the travellers pick them up at the airport?)

U14: Tilsendes. (Mailed.)

S15: Billetterne bliver tilsendt cirka en uge før afrejsen. Vil du mere?

(The tickets will be mailed about one week before the departure. Do you want more?)

---

the reservation task itself, cf. Dybkjær et al. (1995). This was a main reason for not implementing those two tasks.

U15: Ja. (Yes.)

Figure 1. The dialogue G41a from the user test of the Danish Dialogue System with English translation in parentheses, cf. Dybkjær, Bernsen and Dybkjær (1996). S means system, U means user.

Whenever needed, users may initiate meta-communication to resolve misunderstanding or lack of understanding by using one of the keywords 'change' and 'repeat'. The system initiates meta-communication by saying "Sorry, I did not understand" or by asking the user, after a long pause, "Are you still there?" Use of 'repeat' makes the implemented system repeat its most recent utterance. Use of 'change' allows users to correct the latest piece of information given to the system. 'Change' may also be used recursively to correct information from an arbitrary earlier utterance. If a user says 'change', the system will check with the user if the latest piece of information it received was correctly understood. The user may then accept, reject and provide new information, or say 'change' again. In the latter case the system will check the correctness of the penultimate piece of information with the user. For natural and early error detection, the system provides feedback by echoing the key information in the latest user utterance (see, e.g., S3 in Figure 1). Furthermore, at the end of a reservation task, a summary is provided of the entire reservation made by the user (see S14 in Figure 1).

3 Controlled user test of the Danish Dialogue System

When the reservation system had been implemented and debugged, a controlled user test was carried out. In this test, a simulated speech recogniser was used, cf. Bernsen, Dybkjær and Dybkjær (1995). A wizard keyed in the users' answers into the simulated recogniser. The simulation en-

sured that typos were automatically corrected and that input to the parser corresponded to an input string which could have been recognised by the real recogniser. In this set-up, recognition accuracy would be 100% as long as users expressed themselves in accordance with the vocabulary and grammars known to the system. Otherwise, the simulated recogniser would turn the user input into a string which only contained words and grammatical constructs from the recogniser's vocabulary and rules of grammar.

The test was based on 20 different scenarios which had been designed by the system developers to enable exploration of all aspects of the task structure. Since the flight ticket reservation task is a well-structured task in which a prescribed minimum of information must be exchanged between user and system, it was possible to extract from the task structure a set of sub-task components, such as number of travellers, age of traveller, and discount vs. normal fare, any combination of which should be managed by the dialogue system. The scenarios were generated from systematically combining these components.

Twelve external subjects who had not tried the system before and who represented the target group, mostly professional secretaries, participated in the user test. The subjects conducted the dialogues over the telephone in their normal work environments in order to make their task as realistic as possible. Before interacting with the system each subject received an introductory letter, a leaflet briefly describing the system, four scenarios and a questionnaire. After the experiment they received a telephone interview and filled in the questionnaire. The subjects were given a total of 50 tasks based on 48 individual scenarios, two of which contained two tasks. A *task* consists of ordering one or more tickets for one route. A *route* is a full trip, i.e. either a one-way trip, a two-

way trip or a round-trip. The number of recorded dialogues was 57 because subjects sometimes reiterated a failed dialogue and eventually succeeded with their task. A *dialogue* is one path, whether completed or not, through the dialogue structure. If, at the end of the dialogue, the user selects to do a second reservation without hanging up, the user opens a new dialogue. All dialogues were recorded and all transactions between the individual system modules were logged. The recorded dialogues were transcribed and annotated, and the test data was then subjected to diagnostic evaluation. The transcribed corpus comprises 998 system utterances and 998 user utterances.

4 A methodology for diagnostic evaluation

Our diagnostic evaluation of the user test corpus was particularly aimed at detecting, classifying, diagnosing and repairing dialogue interaction problems. For problem detection we used the methodology described in this section which allows the detection of any kind of interaction problems including recognition problems, linguistic problems, such as insufficient grammar and vocabulary, problems of dialogue interaction, and other problems, such as database errors and cases of system breakdown. In order to classify and diagnose dialogue design errors and provide clues to their repair, we used an independently developed typology of non-cooperative system dialogue behaviour (Figure 2). The typology is based on a set of guidelines for cooperative spoken human-machine dialogue. The guidelines were initially developed on the basis of our WOZ corpus (Section 2) and were then refined through comparison with an already established set of maxims for cooperative human-human dialogue, cf. Bernsen, Dybkjær and Dybkjær (1996), Grice (1975). The underlying assumption is that *any dialogue design error,* apart, perhaps, from problems of system impolite-

ness, *can be seen as a problem of non-cooperative system behaviour.* The typology of (system) cooperativity problems presented in Figure 2 distinguishes seven aspects of dialogue. An *aspect* represents the property of dialogue violated through a particular generic or specific problem. A *generic* cooperativity problem (second column) may subsume one or more *specific* problems (third column) which specialise the generic problem to certain classes of phenomena. GP1, for instance, subsumes SP1 and SP2 whereas GP2 does not subsume any specific problems. Although subsumed by generic problems, we believe that the additional classificatory information provided by the specific problems is useful to SLDS diagnostic evaluation.

COOPERATIVITY PROBLEM		
Dialogue aspect	**Generic problem**	**Specific problem**
Group 1: **Informativeness**	GP1: System provides less information than required.	SP1: System is not fully explicit in communicating to users the commitments they have made.
		SP2: Missing system feedback on user information.
	GP2: System provides more information than required.	
Group 2: **Truth and evidence**	GP3: System provides false information.	
	GP4: System provides information for which it lacks evidence.	
Group 3: **Relevance**	G 5: System provides irrelevant information.	
Group 4:	GP6: Obscure system utterance.	

Manner	GP7: Ambiguous system utterance.	SP3: System does not provide same formulation of the same question to users everywhere in its dialogue turns.
	GP8: Too lengthy expressions provided by system.	
	GP9: System provides disorderly discourse.	
Group 5: **Partner asymmetry**	GP10: System does not inform users of important non-normal characteristics which they should, and are able to, take into account to behave co-operatively in dialogue.	SP4: Missing or unclear information on what the system can and cannot do.
		SP5: Missing or unclear instructions on how to interact with the system.
Group 6: **Background knowledge**	GP11: System does not take users' relevant background knowledge into account.	SP6: Lacking anticipation of domain misunderstanding by analogy.
		SP7: System does not separate when possible between the needs of novice and expert users.
	GP12: System does not consider legitimate user expectations as to its own background knowledge.	SP8: Missing system domain knowledge and inference.
Group 7: **Repair and clarification**	GP13: System does not initiate repair or clarification meta-communication in case of communication failure.	SP9: System does not initiate clarification if it has failed to understand the user.
		SP10: Missing clarification of inconsistent user input.
		SP11: Missing clarification of ambiguous user input.

Figure 2. A typology of dialogue cooperativity problems. GP means generic problem. SP means specific problem. GPs subsume SPs.

Detection of interaction problems was done by comparing expected and actual user-system exchanges. Expectations were based on the dialogue task structure and on the scenarios given to users, cf. the example in Figure 3. Actual exchanges were taken from the recorded and transcribed user-system dialogues, cf. Figure 4. Potentially, an interaction problem has been detected if there is a deviation between the expected and actual user-system exchanges. Each such deviation, or problem symptom, is marked up by inserting a brief and temporary description of what the problem seems to be. Following the mark-up process, a diagnostic analysis of the marked-up deviations is carried out. This analysis consists in (i) verification of the symptom observed, including check of the scenario interpretation underlying the presumed deviation; (ii) problem diagnosis through use of dialogue transcriptions and system logs as necessary; (iii) identification of one or several system or dialogue re-design cures, or measures to prevent the occurrence of user errors, including evaluation of their feasibility; and (iv) final classification of the problem. The diagnostic analysis may show that a marked-up utterance contains more, or other, problems of interaction than temporarily indicated to begin with. Similarly, what was initially believed to be a user error may be shown through analysis to be a dialogue design problem, and what was initially believed to be a dialogue design problem may turn out to be something else instead, such as a linguistic problem. In Figures 4 and 5, only dialogue interaction problems have been indicated since other kinds of interaction problems are not discussed here. All dialogues in the user test corpus were annotated, analysed and discussed by at least two different coders.

Anders Bækgaard (ID-number 6), Paul Dalsgaard (ID-number 3) and Børge Lindberg (ID-number 4) work in a department in Aalborg that has customer number 3. They are all going to Copenhagen on the first weekend in February. They want to depart by the earliest flight on Saturday at 7:20 and return by the latest flight on Sunday at 22:40.

Figure 3. Scenario T32.

S11: Are you particularly interested in discount?

U11: Yes.

S12: On which date will the journey start?

U12: The first weekend in February.

S13: Friday February 10th. At which time?

U13: It must be on Saturday at 7:20. **E2, GP10, GP10**

S14: There is no departure at 7:20. The closest other red departure is at 10:50. Do you want this departure? **GP1, SP10**

U14: No.

S15: At which time?

U15: Saturday at 7:20. **GP10, GP10**

S16: There is no departure at 7:20. The closest other red departure is at 10:50. Do you want this departure? **GP1, SP10**

U16: Yes.

S17: On which date will the return journey start?

U17: On Sunday February 5th.

S18: Sunday February 12th. At which time?

U18: 22:40. **E2**

S19: 22:40. You have now booked 3 tickets for Anders Bækgaard, Paul Dalsgaard and Børge Lindberg from Aalborg to Copenhagen on Friday February 10th at 10:50 returning on Sunday February 12th at 22:40. The tickets are on green conditions. Your reference number is 51. The total price is 2820 kroner. Are the tickets to be mailed or will the travellers pick them up at the airport?

U19: They should be mailed. **E2**

Figure 4. Excerpt from dialogue T32a from the user test of the Danish Dialogue System (translated from the Danish), cf. Dybkjær et al. (1996). The dialogue is based on Scenario T32 in Figure 2. S means system, U means user. E means user error, GP means generic dialogue design problem and SP means specific dialogue design problem.

Symptom verification consists in checking, for each marked-up deviation between the expected and actual user-system exchanges, if the expectation was correct. In particular the scenario interpretation should be checked. As we found, a scenario sometimes may have other legitimate interpretations than the one foreseen by the system designers and reflected in the expected user input. If this is the case, the definition of the expected user-system exchange is modified. If, on the other hand, the expectation was correct or if the modified expectation still leaves a deviation, an actual problem of user-system interaction has been detected and steps (ii), (iii) and (iv) above should be performed. Each identified problem was analysed as regards what went wrong and whether the problem was caused by the system or by the user. After a first round of analysis in which a rough and preliminary annotation of each problem was added, a more thorough analysis was carried out. In a final iteration, all results were reconsidered and part of the user errors reclassified as dialogue design errors. A more detailed discussion of dialogue design errors and user errors is provided in Sections 5 and 6, respectively. In the remainder of this section, we illustrate the methodology up to and including step (i).

Figure 5 shows problem detection and final mark-up of dialogue interaction problems in a dialogue from the user test of the Danish Dialogue System. Based on the ticket reservation task structure, Column 1 shows the system's questions in the order in which they would be addressed by users. For system-directed dialogue an ordered list of system questions can be constructed because sub-

tasks are being addressed in a fixed order. For mixed-initiative dialogue and user-directed dialogue the order in which sub-tasks are being addressed may depend on the scenario and on user preferences. A diagnostic evaluation template for such dialogues may therefore be less general and may have to be adapted to each concrete scenario and dialogue. Column 2 in Figure 5 shows the key contents of the expected user input which were filled into the template prior to the user test. In Figure 5 the normative user answers reflect Scenario T32 shown in Figure 3.

Scenario: T32a User: 6 Date: 16 January 1995			
System questions	**Normative user answers**	**Actual system-user exchanges**	**Problems**
System already known	no / yes / -	-	
Customer number	3	S: Is it customer number 4? U: No, 3. S: Number 10 does not exist. Which number? U: 3.	
Number of travellers	3	S: Number of travellers? U: 3.	
ID-numbers	6, 3, 4	S: ID-numbers? U: 6, 3, 4.	
Departure airport	Aalborg	S: From which airport? U: Aalborg.	
Arrival airport	Copenhagen	S: To which airport? U: Copenhagen.	
Return journey	yes	S: Return journey? U: Yes.	

Interested in discount	no / yes	S: Interested in discount? U: Yes.	
Date of departure (out)	February 4	S: Date of departure? U: First weekend in February S: Friday February 10.	
Hour of departure (out)	7:20	S: At which time? U: Saturday at 7:20. S: No departure at 7:20. Closest other red departure at 10:50. Do you want this? U: No. S: At which time? U: Saturday at 7:20. S: No departure at 7:20. Closest other red departure at 10:50. Do you want this? U: Yes.	E2, GP10x2 GP1, SP10 GP10x2 GP1, SP10
Date of departure (home)	February 5	S: Date of return? U: Sunday February 5. S: Sunday February 12.	
Hour of departure (home)	22:40	S: At which time? U: 22:40.	E2
Delivery	airport / mail	S: Confirms the reservation. Should tickets be mailed or picked up at the airport? U: Mailed.	E2

More	yes / no	S: Do you want more? U: Yes.	

Figure 5. The system's questions, key contents of the expected (normative) user answers, actual user-system exchanges, and dialogue interaction problems identified in Dialogue T32a. E means user error, GP means generic dialogue design problem and SP means specific dialogue design problem.

After the user test, the key contents of each actual user and system utterance were entered into Column 3 of the template, cf. Dybkjær et al. (1996). Note that 'key contents' means the semantics of the core message rather than the surface language of user and system utterances. As can be seen from Figure 5, system feedback is only indicated in case of misunderstandings or other problems. In all other cases system feedback is omitted because it merely repeats what the user just said. Pointers from Column 3 to the transcribed corpus allow access to the complete surface language of the user-system exchanges and to a log of internal system module communication whenever needed during diagnostic analysis.

Finally, Column 4 in the template in Figure 5 serves the markup of interaction problems that have been detected through comparison between normative and actual user-system exchanges. Column 4 shows the *detection* of where in the dialogue a dialogue interaction problem occurred and a *final classification* of that problem which indicates what went wrong, cf. Sections 5 and 6.

5 Dialogue design errors

In the diagnostic analysis of the user test data each identified dialogue design error was (a) characterised with respect to its *symptom*, (b) a *diagnosis* was made, and (c) one or several *cures* were proposed. The 'cure' part of the diagnostic analysis may suggest several ways in which to improve system dialogue behaviour in order to prevent detected problems from occurring in the future. For an interaction problem to count as a dialogue design error, at least one of the proposed cures should specify revised dialogue design. This is why problem classification, although used from initial problem detection onwards, can be finally made only when possible cures for the problem have been identified. Furthermore, the 'cure' part of the diagnostic analysis may show that a new *type* of dialogue design error should be added to the typology in Figure 2. What we found was that virtually all dialogue design errors in the user test could be classified according to this typology. Only two new underlying specific guidelines, and hence two new problem types (SP10 and SP11), were added. These had to do with the fact that system-initiated repair meta-communication had not been simulated during WOZ, cf. Bernsen et al. (1996).

A single utterance may represent several dialogue design problems as illustrated, e.g., in S14 in Figure 4. In such cases each individual problem must be separately analysed. In the first part of S14 the system provides insufficient information in that it does not tell that there is a blue departure at 7:20 on Fridays (GP1). Secondly, the departure at 7:20 exists but it does not offer discount and the system implicitly assigns priority to discount over departure time without sufficient reason (SP10). The classifications GP1 and SP10 respectively of the problems in S14 were inserted in the template as shown in Column 4 of Figure 5 to replace the temporary markup of unclassified problems.

In addition to problem classification, the typology also provides cues to problem repair, for instance by pointing to the fact that the system does not provide sufficient information. The 'cure' part of the analysis provides a more detailed proposal for how to repair the problem. For instance the GP1 problem in S14 in Figure 4 could be solved by telling the user that there is no red (i.e. discount) departure but that there is a blue (i.e. normal fare) departure at the chosen hour. It may be discussed if the first part of S14 manifests a problem of type GP1 or of type GP3 (false information). Viewed in isolation, the information provided by the system is false but viewed together with the rest of S14, the information rather seems to be insufficient.

Figure 6 presents an overview of the types of dialogue design error that were identified in the user test. For each identified GP and SP, the observed dialogue design errors are briefly characterised with global indications of their causes and repair. The figure only contains 12 GPs and SPs. However, although the generic problems GP11, GP12 and GP13 were not directly observed in the user test data, these problems are still regarded as having occurred because cases of one or more of the specific problems subsumed by each of them were found in the data.

COOPERATIVITY PROBLEM	No.	TF	CAUSE/REPAIR
GP1: System provides less information than required (final question too open; withholding important information, requested or not).	19		System question design (4). System response design (15).
SP2: Missing system feedback on user information (system misunderstandings only show up later in the dialogue)**.**	2	1	System response feedback design.

GP5: System provides irrelevant information (irrelevant error message produced by grammar failure).	2	1	Grammar design *or* speech recognition design coupled with improved repair design.
GP6: Obscure system utterance (grammatically incorrect response; obscure departure information).	7		System response grammar design (1). System response design (6).
GP7: Ambiguous system utterance (question on point of departure).	2		System question design.
GP10: System does not inform users of important non-normal characteristics which they should, and are able to, take into account to behave co-operatively in dialogue (indirect response, change through comments, asking questions, answering several questions at a time).	33		Unreasonable system demands on users. Improve the system to handle the violations.
SP4: Missing or unclear information on what the system can and cannot do (system does not listen during its own dialogue turns).	33	1	Speech prompt design.
SP5: Missing or unclear instructions on how to interact with the system (under-supported user navigation: use of 'change'; round-trip reservations).	2	1	User instruction design.
SP6: Lacking anticipation of domain misunderstanding by analogy (user is unaware that discount is only possible on return fares).	3		User information design.
SP8: Missing system domain knowledge and inference (temporal inference; inference from negated binary option).	4		System inference design.
SP10: Missing clarification of inconsistent user input (system jumps to wrong conclusion).	5		System clarification question design.

SP11: Missing clarification of ambiguous user input (system jumps to wrong conclusion).	5	2	System clarification question design.

Figure 6. Typology of the 117 particular dialogue design errors identified in the user test. The number (No.) of occurrences of each problem is shown as are the responsibilities for transaction failure (TF) per problem type. The rightmost column shows the global cause(s) of the problems and hence what needs to be repaired to prevent them from reoccurring.

Representative examples of all dialogue design errors identified in the user test can be found in Bernsen, Dybkjær and Dybkjær (1998). Examples of errors of types GP1, SP10 and GP10 were already shown above in Figure 4. All identified errors of type SP10 were very similar whereas the errors of type GP1 were of three different kinds. The examples in Figure 4 show cases in which the system withholds non-requested information. In other cases the system withheld requested information. This happened, for instance, in the following situation: the system had informed that there was no departure at the hour indicated by the user, offering instead a list of possible departures. The user asked the system to repeat. The system, however, only repeated part of its previous turn leaving out the list of possible departures which was precisely the information which the user wanted to have repeated. The third kind of GP1 error occurred in the final system question which is too open. The system asks "Do you want more?" (cf. Figure 1) but the intended meaning is the much more limited question "Do you want to make another reservation?".

Errors of type GP10 were of four different kinds: users asked questions; provided indirect answers, e.g. by answering 'cheap' to the question of hour of departure; answered several questions at a time, often through providing two temporal expressions in the same utterance (U15 in Figure

4); and attempted to make changes through full-sentence expressions rather than by saying 'change' (U13 in Figure 4). The most frequent cases were changes through comments and answering several questions at a time. Almost all of these cases led to misunderstanding or non-understanding.

It is significant that the large majority of dialogue design errors could be straightforwardly classified. It is only to be expected, however, that some errors are borderline cases which may alternatively be classified as being of different types. An example was mentioned above in relation to the analysis of the first part of S14 in Figure 4.

Nine of the 24 problem types in the typology were not observed in the user test. Figure 7 explores why. Most of the problems in question are either easy to avoid during dialogue design once this has been decided (SP1, SP3, SP9); or it is difficult to tell from observed cooperativity problems whether or not concrete cases occur because these types of design error must be massively present for a concrete cooperativity problem to be observed (GP2, GP8, SP7). If less massively present, users tend to suffer in silence during the dialogue and complain afterwards. An example of this was found in the WOZ experiments. The fact that problems GP2 (system provides more information than required) and GP8 (system is too verbose) had occurred became apparent from users' complaints that the system talked too much. The problems were solved by removing superfluous information and constructing more succinct system utterances.

COOPERATIVITY PROBLEM	COMMENTS

SP1: System is not fully explicit in communicating to users the commitments they have made.	Easy to ensure once this has been decided.
GP2: System provides more information than required.	Difficult to test through identified cooperativity problems.
GP3: System provides false information.	Great care taken during dialogue design to avoid this problem.
GP4: System provides information for which it lacks evidence.	The system cannot directly commit this error. Problems SP10 and SP11 indirectly raise issues of this kind.
SP3: System does not provide same formulation of the same question to users everywhere in its dialogue turns.	Easy to ensure once this has been decided.
GP8: System is too verbose.	Difficult to test through identified cooperativity problems.
GP9: System provides disorderly discourse.	Great care taken during dialogue design to avoid this problem.
SP7: System does not separate when possible between the needs of novice and expert users.	Difficult to test through identified cooperativity problems.
SP9: System does not initiate clarification when it has failed to understand the user.	Clarification ability is easy to provide once this has been decided.

Figure 7. Why some dialogue design errors were not observed in the user test.

6 User Errors

The following discussion of the user errors identified in the user test of the Danish Dialogue System is less principled than the discussion in Section 5 of dialogue design errors in the sense that we did not have an independent classification of user error types to depart from. The typology to be presented below thus has been established empirically based on the user test itself.

Some of the dialogue interaction problems detected in the user test corpus were user errors. However, the concept of a 'user error' is a complex one, as we shall see. Our initial definition of a user error, and one which may appear plausible to many system developers, was something like the following: a user error is *a case in which a user does not behave in accordance with the full normative model of the dialogue.* In controlled user testing, the full normative model of the dialogue is defined by (i) explicit designer instructions to users, i.e. the scenarios that users have to carry out in dialogue with the system; (ii) explicit system instructions to users, such as the system's introduction to itself, cf. Figure 1; (iii) explicit system utterances during dialogue; and (iv) implicit system 'instructions' to users. (i) through (iv) will be illustrated and discussed below. (i) is absent in field testing of systems and in practical system use. In such cases, the full normative dialogue model reduces to (ii)-(iv). (ii) is very important to SLDS design and use because SLDSs are vastly inferior dialogue partners when compared to humans. This means that humans must be told how to interact with an SLDS lest they treat the system as a human dialogue partner. Users, of course, should react appropriately to the system's utterances during dialogue (iii), given their scenarios and the dialogue context. The need to include (iv) follows from the important fact that it is impractical to explicitly inform users about all the types of behaviour they should avoid during dialo-

gue with the system. Rather, users should be made to grasp the general fact that the system is a severely restricted dialogue partner which should be treated accordingly.

According to the above definition of 'user error', 102 individual user errors were found in the user test corpus. A more thorough analysis of the user errors that were identified on the basis of the above definition revealed, however, that a significant number of them were *caused* by dialogue design errors. For instance, users responded differently from what they should have responded according to the scenario because of missing system feedback or because a system question was too open and invited users to respond in ways which we had not intended. We shall ignore such cases in what follows because we expect them to disappear once the dialogue design errors discussed above have been fixed. Instead we shall focus on the dialogue errors that were made solely by the users. This leaves 62 individual user errors for discussion in what follows.

Each user error was analysed and (a) characterised with respect to its *symptom*, (b) a *diagnosis* was made, and (c) whenever possible a *preventive measure* was proposed. A single utterance sometimes contained several user errors and sometimes an utterance contained both dialogue design errors and user errors, see e.g. U13 in Figure 4. The analysis of the user error in U13 led to the diagnosis that the user ignores clear system feedback. This error was considered a direct cause of the transaction failure of the dialogue in Figure 4. The user only tries to change the day of the week and not the date although this was also misunderstood by the system. Upon diagnostic analysis, and just like dialogue design error types, user error classification results are inserted in the diagnostic evaluation template to replace the temporary indication of an unclassified problem, cf. Figure 5.

Figure 8 presents an overview of the user error types that were initially identified in the user test. Two error types (E3 and E6) were divided into sub-types. E1 includes the scenario violations, i.e. violations of explicit designer instructions. E2 and E3a include cases in which users did not pay attention to explicit system utterances (feedback and questions). E3b is closely related to E5 (see below). E3b, E4, E5, E6 and E7 represent violations of explicit system instructions provided in the system's introduction (see Figure 1). As we shall see, however, E3b, E4, E5 and E6 turned out to be disguised dialogue design errors. In E8 the user violates implicit system instructions. For each identified user error type in Figure 8, one or more sub-types are indicated which briefly characterise the problem. For each sub-type, preventive measures, if any, are indicated. In the following each error type is discussed in more detail.

Error Types	**Error Sub-Types**	**No. of Cases**	**Preventive Measure**
E1. Misunderstanding of scenario.	a. Careless reading or processing.	14	Use clear scenarios, carefully studied, to reduce errors.
E2. Ignoring clear system feedback.	a. Straight ignorance.	7	Encourage user seriousness to reduce errors.
E3. Responding to a question different from the clear system question.	a. Straight wrong response.	4	Encourage user seriousness to reduce errors.
	b. Indirect response.	3	Disguised dialogue design error.
E4. Change through comments (including 'false' keywords).	a. Cognitive overload.	17	Disguised dialogue design error.

E5. Asking questions.	a. Asking for decision-relevant information.	4	Disguised dialogue design error.
E6. Answering several questions at a time.	a. Natural response 'package'.	10	Disguised dialogue design error.
	b. Slip.	1	None.
E7. Thinking aloud.	a. Natural thinking aloud.	1	None.
E8. Non-cooperativity.	a. Unnecessary complexity.	1	None.

Figure 8. The initially identified user error types and sub-types.

E1. Misunderstanding the scenario

Controlled user testing is important in systems design and it may be worth considering ways of preventing user errors in controlled test environments. It should be noted that scenario misunderstandings cannot give rise to transaction failure. Normally, users just carry out a different scenario. Transaction failure occurs only when users do not obtain the reservation they actually ask for. Carrying out a different scenario may, however, affect the quality of system evaluation. If a scenario is not being carried out an important part of the dialogue model may remain untested. Almost one fourth of the detected 62 user errors were due to users acting against the instructions in the scenarios. E1 raises two issues in the preparation of controlled user testing: (i) to reduce the number of errors, scenarios should be made as clear as possible. Clear scenarios should not be confused with *simple* scenarios. Scenarios should reflect the types of information real users actually have when addressing the system. (ii) Users should be encouraged to carefully prepare themselves on the scenarios they are to complete in conversation with the system. A practical solution is to promise an award to subjects who stick to their scenarios in conversation with the system.

E2. Ignoring clear system feedback

The speech recognition capabilities of most telephone-based systems are still fragile. It is therefore important that users listen carefully to the system's feedback to verify that they have been correctly understood. Of the seven transaction failures in the user test, one was caused by a combination of a dialogue design problem and a user who ignored clear system feedback. A second transaction failure occurred solely because the user did not pay sufficient attention to the system's feedback which made it clear that the user had been misunderstood, cf. Figure 4. Three of the seven detected E2 cases occurred in this dialogue in which the user continuously ignored the system feedback on dates. Thus, four out of the seven detected cases of ignored system feedback had severe implications for the success of the transaction. Moreover, had the user test included a real recogniser, more cases of system misunderstanding would no doubt have occurred and hence more cases in which users would have had to identify system misrecognitions from the system's feedback. E2 raises the issue of encouraging test subjects to 'act' seriously in dialogue with the system and be very attentive to what the system says.

E3. Responding to a question different from a clear system question

E3 has at least two sub-types. The first sub-type, E3a, included four cases in which users gave a straight wrong response to a system question, such as answering "Saturday" to a system question concerning the departure airport. In one case the answer was not understood by the system and in three cases it was misunderstood. E3a raises the same issue as did E2 of encouraging users to seriously pay attention to the system's utterances. The second sub-type, E3b, concerns *indirect* user responses. For instance, a user answered "it must be as cheap as possible" to a system question on

hour of departure. In human-human conversation, this type of indirect answer would be perfectly all right. It indicates that the speaker does not possess the information necessary to provide a direct answer, and a human travel agent would in response list the relevant departures on which discount may be obtained. Our SLDS, however, has limited inferential capabilities and is not able to cope with indirect responses. They will be either not understood or misunderstood. It may be argued that indirect user responses are not user errors at all. They do not conflict with the system's introduction. At best it might be argued that they conflict with the difficult requirement on users which we have called 'implicit instructions' to users**.**

E4. Change through comments

E4 gave rise to numerous (almost 30%) user errors in the test. In 16 out of 17 cases, users tried to make corrections through natural sentences rather than by using the keywords prescribed in the system's introduction. In none of these cases was the requested correction understood as intended. Only in one case did the user achieve the intended correction. In this case, the user used a keyword different from 'change' but meaning the same, which accidentally was recognised as 'change'. The theoretical importance of these findings is that of emphasising the undesirability of including designer-designed user keywords in dialogue design for SLDSs. Such keywords will neither correspond to the keywords preferred by all or most users nor to the natural preference among native speakers to reply in spoken sentence form rather than through keywords. It is furthermore our hypothesis that the more cognitive load a user has at a certain stage during dialogue task performance, the more likely it is that the user will forget about specific keywords to be used. E4 raises the hard issue of allowing users a more natural form of repair meta-communication.

E5. Asking questions

Three out of four E5 cases occurred when the system had asked for an hour of departure, expressing that the user did not have exact information on departure time. What the observed cases show is that reservation dialogue, in its very nature, so to speak, is *informed reservation* dialogue. As with E3b above, it is not clear what the dialogue designer should do about this problem in the short term. Current systems are not likely to be able to understand all possible and relevant user questions in the context of ordering tasks. The optimistic conclusion is that E3b and E5 only constituted 7 errors in total in the user test, and that skilled users of the system will learn other ways of eliciting the system's knowledge about departure times, for instance by answering "in the afternoon" when asked about an hour of departure. However, a principled solution to the problem only seems possible through enabling the system to conduct rather sophisticated mixed-initiative domain dialogue.

E6. Answering several questions at a time

E6 has at least two sub-types. The first sub-type, E6a, gave rise to many (about 16%) user errors in the test. In 7 of the 10 cases, only the part of the user's response which answered the system's question was understood. In the remaining 3 cases the entire user response was misunderstood. What this error type suggests is that (i) users naturally store information in 'packages' consisting of several pieces of information, and (ii) users have stereotypical linguistic response patterns, such as prefixing a 'change' keyword with a 'no'. Dialogue designers should be aware of these natural stereotypes and enable the system to understand them. This problem appears solvable by today's technology. The second sub-type, E6b, illustrates a phenomenon which no feat of dialogue design

is likely to remove, i.e. the naturally occurring slips-of-the-tongue in spontaneous speech. Slips do not appear to be a major problem, however.

E7. Thinking aloud

E7 illustrates another phenomenon which no dialogue design effort is likely to remove, i.e. the naturally occurring thinking-aloud in spontaneous speech. Thinking-aloud does not appear to constitute a major problem, however. Only one case was observed in the entire corpus.

E8. Non-cooperativity

Only one case of deliberate user non-cooperativity was detected in the test corpus. The user replied "the ticket should not be sent" to the system's question of whether the ticket should be sent or would be picked up at the airport. We know that the particular user who caused the problem was deliberately testing the hypothesis that the system would be unable to handle the input because she said so in the telephone interview following her interaction with the system. SLDSs designers have no way of designing dialogues with sufficient robustness to withstand deliberately non-cooperative users. Nor should SLDSs designers attempt to do so, apart, of course, from ensuring that the system will not break down and that deliberately non-cooperative users cannot cause any harm.

Summary on user errors

The E1 errors are of only minor importance as they will disappear when the system is being used in real life. Similarly, E8 errors are of minor importance because users will stop experimenting

with the system when they want the task done. E6b and E7 can hardly be prevented but, at least according to our test data, they are infrequent and do not cause severe problems of interaction.

E2 and E3a seem to have a much larger effect on dialogue transaction success. Although they can hardly be completely avoided, it is likely that their number can be reduced by clearly making users aware of the importance of paying attention to system feedback and system questions. Real-life users are likely to be more attentive.

E3b, E4, E5 and E6a would all be perfectly acceptable in human-human dialogue. However, because of the limited dialogue capabilities of our SLDS, it is clearly stated in the system's introduction how users should interact with it in order to prevent these errors. Whereas E3b is less clear, the E4, E5 and E6a errors all violate the system's explicit instructions. The important question is why so many users violate exactly these instructions. A likely explanation is that, at least for many users, it is not *cognitively feasible* to follow the system's explicit instructions. This reveals a fundamental shortcoming in our initial concept of 'user error'. It is not sufficient to provide clear and explicit instructions to users on how to interact with the system. *It must also be possible for users, such as they are, to follow these instructions in practice,* cf. cooperativity problem GP10 in Figure 2. The conclusion is that E3b, E4, E5 and E6a are *not* user errors at all but rather constitute more or less difficult problems of dialogue design. A simple revision of our initial definition of 'user error' provided at the start of the present section is: a user error is *a case in which a user does not behave in accordance with the full, and cognitively feasible, normative model of the dialogue.*

7 Conclusion

We have presented a two-part methodology for diagnostic evaluation of spoken language dialogue systems. Use of the methodology was demonstrated on the dialogue corpus from the user test of the Danish Dialogue System. The dialogue interaction problems which were detected in the corpus included dialogue design errors as well as user errors. Based on an independent typology of dialogue cooperativity problems, dialogue design errors were classified and cues to their repair provided. The fact that the typology was sufficient for classifying virtually all the detected dialogue design errors is worth repeating here. This suggests that the typology may be useful to other developers of SLDSs when performing diagnostic evaluation of their systems. The identified user errors were classified on an empirical basis, preventive measures to avoid them were proposed when possible, and the concept of a 'user error' was refined in the process.

We are currently investigating the generality and transferability of the guidelines underlying the typology of dialogue cooperativity problems presented in Figure 2. *Generality* is being tested by applying the guidelines to corpora from interactions with (i) *systems that are different* from the Danish Dialogue System and which (ii) cover *different task domains.* Moreover (iii), the guidelines are being applied as a *design guide* prior to implementation rather than to the diagnostic evaluation of an implemented system. Finally (iv), the guidelines are being applied in *less controlled circumstances* compared to those that obtain in a controlled user test, such as when dealing with field test data. *Transferability* is being tested by investigating (a) what it takes for a novice SLDSs developer to learn to master the guidelines; and (b) how the required learning steps may be supported and "packaged" for transfer to other developers so that they can easily learn how to use the guidelines. Preliminary results are presented in Bernsen, Dybkjær, Dybkjær and Zinkevicius

(1997). The results demonstrate that the guidelines can be used as a design guide during early design in much the same way as the typology in Figure 2 is used for diagnostic evaluation of implemented systems. We therefore believe that the guidelines can support the design of a first interaction model no matter whether the model is intended for simulation-prior-to-implementation or straight implementation.

Acknowledgements. The Danish Dialogue System was developed in collaboration between the Center for PersonKommunikation, Aalborg University (speech recognition, grammar), the Centre for Language Technology, Copenhagen (grammar, parsing), and the authors (dialogue and application design and implementation, human-machine aspects). The project was generously supported by the Danish Research Councils for the Technical and the Natural Sciences.

References

Aust, H. & Örder, M. (1995). Dialogue control in automatic inquiry systems. *Proceedings of the ESCA Workshop on Spoken Dialogue Systems,* Vigsø, 121-124.

Bernsen, N. O., Dybkjær, H. & Dybkjær, L. (1995). Exploring the limits of system-directed dialogue. Dialogue evaluation of the Danish Dialogue System. *Proceedings of Eurospeech '95,* Madrid, 1457-60.

Bernsen, N. O., Dybkjær, H. & Dybkjær, L. (1996). Cooperativity in human-machine and human-human spoken dialogue. *Discourse Processes,* Vol. 21, No. 2, 213-236.

Bernsen, N. O., Dybkjær, H. & Dybkjær, L. (1998). *Designing interactive speech systems. From first ideas to user testing.* Springer Verlag (to appear).

Bernsen, N. O., Dybkjær, H., Dybkjær, L. & V. Zinkevicius (1997). Generality and transferability. Two issues in putting a dialogue evaluation tool into practical use. *Proceedings of Eurospeech '97*, (to appear).

Calzolari, N. & McNaught, J. (1996). EAGLES—Expert Advisory Group on Language Engineering Standards. URL: http://www.ilc.pi.cnr.it/EAGLES/home.html

Cole, R., Novick, D. G., Fanty, M., Vermeulen, P., Sutton, S., Burnett, D. & Schalkwyk, J. (1994). A prototype voice-response questionnaire for the US Census. *Proceedings of the ICSLP '94,* Yokohama, 683-686.

Dybkjær, H., Bernsen, N. O. & Dybkjær, L. (1993). Wizard-of-Oz and the trade-off between naturalness and recogniser constraints. *Proceedings of Eurospeech '93*, Berlin, 947-50.

Dybkjær, L., Bernsen, N. O. & Dybkjær, H. (1995). Different spoken language dialogues for different tasks. A task-oriented dialogue theory. In Pfleger, S., Gonçalves, J. and Varghese, K. C., Eds. *Advances in Human-Computer Interaction. Human Comfort and Security*, Springer Research Reports, 46-61.

Dybkjær, L., Bernsen, N. O. & Dybkjær, H. (1996). Evaluation of spoken dialogues. User test with a simulated speech recogniser. *Report 9b from the Danish Project in Spoken Language Dialogue Systems*. Roskilde University. 3 volumes of 18 pages, 265 pages, and 109 pages, respectively.

Eckert, W., Nöth, E., Niemann, H. & Schukat-Talamazzini, E. (1995). Real users behave weird - Experiences made collecting large human-machine-dialog corpora. *Proceedings of the ESCA Workshop on Spoken Dialogue Systems*, Vigsø, 193-196.

Fraser, N. M. & Gilbert, G. N. (1991). Simulating speech systems. *Computer Speech and Language* 5, 81-99.

Gallier, J. R. & Sparck-Jones, S. (1996). *Evaluating Natural Language Processing systems.* Springer.

Gaizauskas, R., Ed. (1997). *Proceedings of the SALT Club Workshop on Evaluation in Speech and Language Technology*, Sheffield.

Gibbon, D., Moore, R. & Winski, R., Eds. (1997). *Handbook of standards and resources for spoken language systems*. Mouton de Gruyter, Berlin, New York.

Grice, P. (1975). Logic and conversation. In Cole, P. and Morgan, J. L., Eds. *Syntax and Semantics,* Vol. 3, *Speech Acts,* New York, Academic Press, 41-58. Reprinted in P. Grice, *Studies in the Way of Words,* Harvard University Press, Cambridge MA, 1989.

Hirschberg, J., Kamm , C. & Walker, M., Eds. (1997). *Proceedings of the ACL Workshop on Interactive Spoken Dialog Systems: Bringing Speech and NLP Together in Real Applications.* Madrid.

Hirschmann, L. & Thompson, H. S. (1996). Overview of evaluation in speech and natural language processing. In Cole, R. A., Mariani, J., Uszkoreit, H., Zaenen, A. & Zue, V., Editorial Board, Varile, G. & Zampolli, A., Managing Editors. *Survey of the State of the Art in Human Language Technology,* Section 13.1. URL: http://www.cse.ogi.edu/CSLU/HLTsurvey/.

McRoy, S., Ed. *Proceedings of the AAAI Workshop on Detecting, Repairing, and Preventing Human-Machine Miscommunication,* 13th National Conference on Artificial Intelligence. Portland, Oregon.

Peckham, J. (1993). A new generation of spoken dialogue systems: Results and lessons from the SUNDIAL project. *Proceedings of Eurospeech '93*, Berlin, 33-40.