INFERENTIAL ISSUES IN A SPOKEN LANGUAGE DIALOGUE SYSTEM

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

{dybkjaer,nob,laila}@cog.ruc.dk

Abstract

Spoken language dialogue systems (SLDSs) need inferential mechanisms to act as natural dialogue partners. This paper presents and discusses the inferential needs of a task-oriented SLDS. The discussion is based on dialogue model and database development and implementation of the Danish dialogue system[‡] and on a user test of the system. The presentation is richly illustrated by transcriptions from the user test.

1 INTRODUCTION

The ability to make inferences is a necessary condition of natural human dialogue performance. Without inference each single piece of information to be communicated in dialogue would have to be stated explicitly. This may be useful for, e.g., some kinds of instruction dialogues but in most cases it would be cumbersome and unnatural. The inferential capabilities required depend on a number of parameters such as the topic of the dialogue, how much the interlocutors know about the topic and how well they know each other. In extreme cases, human-human dialogue can be almost incomprehensible to outsiders because the dialogue builds heavily on the ability of the interlocutor to infer large chunks of information which are not being explicitly stated.

Also in task-oriented human-machine dialogue some degree of inferential power is needed to enable the machine to conduct natural dialogue. It is today far from possible to equip spoken language dialogue systems (SLDSs) with an inferential capacity corresponding to that of humans. To some extent this problem may be solved by stating explicitly how the users should interact with the system and by making clear what the system can and cannot do. However, even then an SLDS must be able to make a number of basic inferences to be an acceptable interlocutor. Missing inferential power will easily lead to communication failure.

---

[*] Centre for Cognitive Science, Roskilde University,
P.O. Box 260, DK-4000 Roskilde, Denmark.
Phone +45 46 75 77 11 Fax +45 46 75 45 02 Url http://www.cog.ruc.dk/

[‡] The Danish dialogue system is developed in collaboration with Center for Person-Kommunikation, Ålborg, and Centre for Language Technology, Copenhagen.
The work is supported by the Danish Research Councils.

S3-52	Hello! This is the DanLuft reservation service for domestic flights. Do you know this system?
U3-52	Yes I do.
S3-53	Please state your customer number!
U3-53	Customer number 3.
S3-54	Customer number 3. How many people will travel?
U3-54	One person.
S3-55	One person. Please state the id-number of the person!
U3-55	Number 2.
S3-56	Id-number 2, Jan Kristensen. Where does the journey start?
U3-56	It starts in Ålborg.
S3-57	From Ålborg. Where does the journey end?
U3-57	It goes to Copenhagen.
S3-58	To Copenhagen. Do you want a return ticket?
U3-58	Yes, I would like that.
S3-59	Are you particularly(!) interested in discount?
U3-59	No.
S3-60	At which date will the journey start?
U3-60	On Monday.
S3-61	Monday the 16/1. At which time of the day?
U3-61	A quarter past 7.
S3-62	There is no departure at 7:15. The closest other departure is at 7:20. Do you want this departure?
U3-62	Yes, I take that.
S3-63	On which date will the return journey start?
U3-63	Same day.
S3-64	Monday the 16/1. At which time of the day?
U3-64	A quarter to 6.
S3-65	The home journey must not be before the out journey! At which time of the day?
U3-65	17:45.
S3-66	17:45. You have now booked 1 ticket for id-number 2 Jan Kristensen from Ålborg to Copenhagen on Monday the 16th of January at 7:30 returning on Monday the 16th of January at 17:45. Your reference number is 52. The total price is 1410 kroner. Is the ticket to be mailed or will it be picked up in the airport?
U3-66	It should be mailed.
S3-67	The ticket will be sent to the airport's ticket office. Tickets should be picked up no later than 20 minutes before departure. Do you want more?
U3-67	No.
S3-68	Goodbye!

Transcription 1. A complete dialogue from the user test. The text is translated from the Danish. Utterance numbers refer to the transcriptions [Dybkjær et al. 1995]. For instance, "S3-52" is the 52nd system utterance in conversation with user number 3, and the user answer is presented in "U3-52".

This paper discusses the needs for inferential mechanisms in SLDSs and the problems which may occur when the system's inferential power is insufficient. Inferences will be considered at the phenomenological level, i.e. we shall focus on their consequences at the dialogue level and not on the underlying formal constructions.

The discussion is based on experience from dialogue model and database development and implementation and from a user test of the Danish dialogue system which is a state-of-the-art SLDS for Danish domestic flight ticket reservation. A typical dialogue from the user test is shown in Trancription 1 (hereafter referred to as T-1). As will be seen from the examples, the problem typically is to determine and specify *which* inferences are needed rather than *how* they can be made.

Section 2 describes the implemented dialogue system, the dialogue model and the user test of the implemented system. Section 3 discusses inferential issues in the system, in particular the problems detected in the corpus of dialogues from the user test. Section 4 concludes the paper.

2 The Danish dialogue system

The Danish dialogue system [Baekgaard et al. 1995] is a walk-up-and-use PC-application accessed over the telephone and understanding a continuously spoken fragment of Danish runs on a PC. The system architecture, the design of the dialogue model and its evaluation are briefly described.

The dialogue system has five main components: the *speech recogniser* receives the user's speech signals as input. The speech recogniser is speaker-independent and uses hidden Markov models to produce a 1-best string of words. The *parser* analyses this string and extracts the semantic contents into frame-like structures called semantic objects. The *dialogue handling module* interprets the semantic objects and decides on the next system action which may be to send a query to the 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. The *database* contains information on timetables, flights, reservations, customers and rules for managing the information and queries it receives. System *output* is produced by concatenating pre-recorded phrases.

The *dialogue model* for the Danish dialogue system was iteratively designed by means of the Wizard of Oz (WOZ) method. WOZ is an iterative simulation technique which is well suited to the testing of dialogue models and the adjustment of design goals and design constraints prior to implementation. During each iteration, a human (the wizard) simulates the system in dialogue with subjects (users) who should preferably believe that they are speaking to a real system [Fraser and Gilbert 1991].

The dialogue model had to satisfy technological constraints imposed by the speech recogniser: to ensure real-time performance, at most 100 words could be active in memory at a time; to ensure an acceptable recognition rate, an average and a maximum user utterance length of 3-4 words and 10 words, respectively, were imposed. Other design goals, such as linguistic naturalness, dialogue naturalness and dialogue flexibility had to be traded off against these constraints [Dybkjær et al. 1993]. Finally, limited project resources set the total system vocabulary size to about 500 words.

The WOZ dialogue model development was iterated until the model satisfied the design constraints. In each iteration, the dialogues were recorded, transcribed, analysed and used as a basis for improvements on the dialogue model. We performed seven WOZ iterations yielding 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 seven iterations. Dialogues were based on written descriptions of reservation tasks (scenarios).

Due to the hard constraints on the active vocabulary size, the dialogue model resulting from the last WOZ iteration uses system-directed *domain communication* (concerns the task domain) but allows users to initiate *meta-communication* (concerns the user-system communication itself and is usually undertaken for purposes of clarification or repair) through use of the keywords 'change' and 'repeat'.

When the dialogue model and the other system components had been implemented and debugged, a user test was carried out. In the user test [Bernsen et al. 1995, Dybkjær et al. 1995], the implemented system was used with a simulated recogniser. A wizard keyed in the users' answers to the simulated recogniser which ensured that typos were automatically corrected and that input to the parser corresponded to an input string which could have been recognised by the real speech recogniser. The recognition accuracy would be 100% as long as users remained within the vocabulary and grammars known to the system. Otherwise, the simulated recogniser would turn the input into a string which only contained words and grammatical constructions that were within the recogniser's vocabulary and which conformed to the recogniser's grammar rules.

Ten external and two in-house subjects participated in the user test. Most of them were professional secretaries, i.e. they had a background corresponding to that of the intended main end-user group. As in the WOZ experiments the user test dialogues were based on scenarios and conducted over the telephone. The user test produced a corpus of 57 dialogues. Subjects sometimes repeated a scenario if they did not succeed the first time.

A full dialogue from the user test is shown in (T-1). The typical exchange consists of a system question followed by a user answer followed by system confirmation feedback and a new question (see, e.g., T-1: S3-53, U3-53, S3-54). Variations mainly occur when dialogue interaction problems arise (T-1: S3-62, S3-65). In the next section, inferential issues related to such problems are discussed and illustrated with examples from the user test corpus.

3 INFERENTIAL ISSUES

The inferences that are needed in an SLDS depend on such aspects as task, domain, language, user model and dialogue history. *Task* and *domain* determine the complexity of the inferences necessary. In the Danish dialogue system, the issues about hour, day, fare and their relationships generate the most complex problems. The input *language*—as determined by recognition and parsing—imposes a clear limit to the possible inferences: the system cannot reason about what is simply not available. Finally, the availability of internal data structures such as a *user model* and a *dialogue history* (T-2) may either enable inferences or reduce them to simple information retrieval; on the other hand, the run-time construction of such structures may require much inferential power.

S7-1	Hello! This is the DanLuft reservation service for domestic flights. Do you know this system?
U7-1	Nope.
S7-2	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.
	- - - *(The user completes the present task and starts the next reservation task)*
S7-14	Do you want more?
U7-14	Yes
S7-15	Customer number 2 is that correct?
	- - - *(the user inputs the travellers and the route and starts to input date and hour)*
S7-21	Do you want a return travel?
U7-21	Yes.
S7-22	Red discount can be obtained on certain week days, green discount on weekends. In the case of red discount the home journey can only start after two nights. Are you particularly! interested in discount?

Transcription 2. *User model*.. The system only presents the introduction (S7-2) and discount information (S7-22) if the user does not know the system (U7-1). Contrast this with (T-1: U3-52, S3-59). Also, in subsequent tasks the system assumes to be dealing with the same customer (S7-15).

In this section, we first address groups of inferential problems that are closely related to the current spoken dialogue reservation system. We then take a brief look at the problems that will arise from increased complexity of task and language. Examples are provided of each problem. The examples are characterised with respect to the aspects of task, domain, language, user model and dialogue history, and illustrated with transcriptions from the user test and other experiments with the reservation system.

3.1 The current dialogue system

The problems addressed in this subsection are solved in the current dialogue system or could be so with no change in the language and with minor change in the dialogue representation.

Categorisation and identification

The largest group of inferences includes categorisations and identifications that can be made through straightforward lookups in the database.

Categorisation is used to distinguish between certain user groups. E.g., users are categorised as novices or experienced users (T-1: U3-52), and certain information is only given to novice users (T-2: S7-2, S7-22).

Identification includes the checking and retrieval of values, such as checking the existence of a certain id-number and retrieval of the name corresponding to the number (T-1: S3-54, S3-56), or checking the existence of a certain airport name and route (T-1: S3-58).

Categorisation and identification are closely related to the domain (i.e. which identifications can be made) and to the language (i.e. which synonyms can occur, such as ‘Kastrup’ and ‘Copenhagen airport’).

Saving turns

In a number of cases the system can make simple inferences that save turns. The system is presently able to perform the first and second types of inference described below. The third type has not been included yet but could be so fairly easily. The inferences depend on the task and domain aspects and help improve the dialogue structure.

Delivery: if the date of departure is less than three days from the date of reservation, the choice of having the tickets sent by mail (T-3) will not be offered to the user. Instead the system will inform the user that the tickets must be picked up in the airport.

Binary choice: when there is a choice between two values, a rejection of one value can be assumed to be an acceptance of the other. In (T-3: S9-59) the system fails to make this inference but the problem has been fixed after the user test

Repeated checking: when a user wants to change information, the system must check the information already provided for the current task. The general rule used in the current system is that if the departure or the arrival airport has been changed by the user, the system will check the other airport with the user if already provided. However, in (T-4: S8-45) the repeated checking of the departure airport is superfluous since the user has already confirmed this airport.

S9-56	Do you want the ticket to be mailed or will it be picked up in the airport?
U9-56	It should not be mailed.
S9-57	The ticket will be mailed in a couple of days. *(U9 overlap: Change)* Do you want more?
U9-57	Change.
S9-58	The ticket will be mailed. Is that correct?
U9-58	No.
S9-59	Do you want the ticket to be mailed or will it be picked up in the airport?

Transcription 3. *Negation*. The system does not understand negation (U9-56). No cooperative user applies negation. U9 afterwards claimed to have used a negation deliberately because s/he believed that it would not work.
Binary choice. In (S9-59), instead of asking again the system might infer that a “no” (U9-58) to a binary value is an implicit acceptance of the alternative.

S8-42	The journey starts in København. Is that correct?
U8-42	Yes.
S8-43	The journey ends in Billund. Is that correct?
U8-43	No.
S8-44	Where does the journey end?
U8-44	Odense.
S8-45	To Odense. *(U8 overlap: Yes)* The journey starts in Copenhagen. Is that correct?
U8-45	Yes.

Transcription 4. *Repeated check*. In (S8-45) the repeated check of Copenhagen is unnecessary because (S8-42).

Ambiguity and completion

Sometimes user input is not complete but the missing information can be trivially inferred. The Danish dialogue system is able to infer a date from a number of incomplete or relative date expressions, such as “the 23rd”, “on Saturday” and “same day”. Presently, only the most common date expressions are understood by the system though even they are not trivial, for example:

Date completion: from a day of week, infer the date by finding the firstcoming date identical with that day of the week (T-1: U3-60, T-5). A subtle point is that for the outjourney this date is the firstcoming date after today, but for the homejourney the date is the firstcoming date on or after the outjourney date.

U2-69	Monday.
S2-70	Monday the 16/1.

Transcription 5. *Completion*. The system fills in the most likely value (here the firstcoming Monday after a given point in time).

Which types of date expressions must be included for the system to have sufficient lexical coverage must be discovered through field tests and cannot be predicted in advance. Probably expressions such as “Easter Sunday” will also be needed whereas, e.g., “poetic” designations of morning and evening are unnecessary.

In other cases no unique interpretation can be inferred because the information is inherently ambiguous unless additional constraints are provided by the dialogue context:

S4-63	Thursday the 26/1. Which time of the day?
U4-63	Around 7:30.
S4-64	19:30. On which day will the return journey start?

U4-64	Thursday the 26/1.
S4-65	Thursday the 26/1. Which time of the day?
U4-65	17:20.

Transcription 6. *Ambiguity*. Since there is no departure at 7:30 am the system proposes 7:30 pm which the user appears to accept. However, the user actually meant "in the early morning" as (U4-65) makes clear. Instead, the user should have been offered the alternative 7:20 in (S4-64). See also (T-1: U3-64).

Am or pm: in the Danish a 24-hour time frame is often used for scheduling events. However, quite frequently an hour value between 0 and 12 is implicitly meant to be pm (T-1: U3-64, U3-65). In the current reservation system, if the user indicates an hour value between 0 and 12 with no corresponding departure, which could also be a pm value, then the system checks if there is a departure corresponding to the pm interpretation. If there is, the system infers that the user wants that departure (T-6). However, this inference is often wrong. It should be used only if additional constraints are available. For instance, if the home journey is on the same day as the out journey, this inference may often be preferable to imputing an error to the user (T-1: U3-64, S3-65). In general, whenever the hour input cannot be disambiguated by the system from context, it should ask the user for clarification.

The above inferences depend on the domain and language aspects.

Errors, preferences and conflicts

User input is of course checked against the domain and the current task state. This may result in the detection of errors or conflicts. The current default system behaviour is to state the detected error or conflict and then ask for a new value:

Sold out: if a requested departure (or set of departures) is sold out then the system informs the user and asks for a new hour of departure (T-7: S4-46).

S4-45	Saturday the 21/1. At which time of the day?
U4-45	It should be an outjourney between 7 and 9 in the morning, please.
S4-46	The green departures are sold out in the morning. At which time of the day?
U4-46	Is there something between 9 o'clock and 12 o'clock?
S4-47	There is no departure at 12 o'clock. The closest other departures are at 9:20 and at 22:30.

Transcription 7. *Alternatives*. The system ought to state the departure at 9:20 already when rejecting the morning departures (S4-46). If the user indicates a precise hour of departure, alternatives are presented (S4-47). *Language*. The system does not understand modifiers and coordinated phrases, such as "between …" (U4-46). Also many other time modifiers exist which are not recognised, for instance "before", "after" and "around".

Similar cases exist for each domain constraint that can be violated. However, sometimes the default behaviour is not sufficient, in particular if the solution involves a potential trade-off between user preferences:

Discount departures: when the user has agreed to be particularly interested in discount the system assumes that the user is not interested in departures which do not allow discount. However, not mentioning some departures may confuse the user and, even worse, the user may be more interested in a particular departure than in discount (T-8).

Home before out: in some cases a user may state a time (date or hour) for the home journey which lies before the time for the out journey. Since the time for the out journey has been accepted by the user who did not reject the system's feedback, the default system behaviour would be to treat this as a user error. This is what the current system does. However, in certain contexts, for instance if the user insists (looping, see T-17) or in the above am-or-pm case, other system behaviours would seem more appropriate.

S2-70	Monday the 16/1. At which time of the day?
U2-70	7:20.
S2-71	There is no departure at 7:20. The closest other red departure is at 10:50. Do you want this departure?
U2-71	No thanks.
S2-72	At which time of the day?
U2-72	7:20.
S2-73	There is no departure at 7:20. The closest other red departure is at 10:50. Do you want this departure?

Transcription 8. *Preferences*. The user has previously opted for discount. Now the system only considers discount departures. The user knows from the scenario that there is an ordinary departure at 7:20 and therefore insists on having it. The system persists (S2-73). In (S2-71) the information could be more informative, and at least after (U2-72) the system should have detected a loop and initiated a clarification/repair dialogue. (The user later obtains the departure 7:20 after having cancelled discount through laborious use of "change".)

Meta-communication

Meta-communication inference problems are primarily related to the language. In the current dialogue system users can only initiate meta-communication by means of the keywords "change" and "repeat". However, in certain cases it is possible and desirable to interpret user actions as well-defined attempts to change something. The following case is not included in the current system but could easily be so:

S13-26	Id-number 1, Jens Hansen. Where does the journey start?
U13-26	No, it is not Jens Hansen, id-number 1 is Lars Bo Larsen.
S13-27	Sorry, where does the journey start?

Transcription 9. *Communication failure*. The system actually catches the rejection "no" (U13-26). In general, "no" is insufficient as an answer to focused questions, but in this case, as a reply to a focused question right after feedback, the "no" should be interpreted as a rejection of the feedback. Moreover, the formulation "sorry" is taken by the user to be the system's acceptance of the change whereas the system meant "I did not get anything relevant out of your utterance". (The user later reserves a correct ticket, but the faulty one is never deleted.)

"No" as "change": in isolation, "yes" and "no" are not appropriate responses to questions such as "Please state your customer number". However, in a system turn questions are often preceded by feedback on the previous user answer, and in such cases the "yes" or "no" can be interpreted as referring to this feedback. Thus, in (T-9: U13-26) the system should infer that the "no" (the only thing recognised from this utterance) means "change the information given in the feedback". This inference can only be made if the feedback immediately precedes the system's question, and then only if the system's question is not a Boolean question.

Recognition failure

The system must initiate meta-communication when it fails, at the semantic level, to understand the user input. The system may fail completely to recognise the user utterance in which case the message "no interpretation of semantic input" is sent to the dialogue handler and the output to the user will be "Sorry, I did not understand" (T-10: S7-46).

S7-45	At which time of the day?
U7-45	Around 22 o'clock
S7-46	Sorry, I did not understand.
U7-46	Around 22.
S7-47	Sorry, at which time of the day?
U7-47	22.
S7-48	Sorry, I did not understand.

Transcription 10. *Looping*. In (S7-47) and certainly no later than (S7-48) the system ought to detect that some clarification/repair action is appropriate. The "Around 22 o'clock" and "22" gave no recognition at all whereas "Around 22" yielded the (inappropriate and wrong) "yes" at the semantic level.

In other cases, the system may recognise something which is in-appropriate in the given context, typically "yes" as an answer to a focused question, i.e. a question which invites the user to state a proper name or something similar, such as the name of an airport or an id-number. If the dialogue handler cannot use the input it will answer "Sorry" and repeat its question (T-10: S7-47). In certain cases something better could be done: a "no" to a focused question right after feedback should be interpreted as a "change" command, cf. on meta-communication above (T-9).

3.2 Increasing complexity

The inferential issues described above are either solved in the current version of the dialogue system or could easily be so. However, the user test made us aware of other issues that are not relevant to the existing language model or which would only have to be addressed when more complex tasks were to be addressed.

Language

Many problems in the user test dialogues cannot be solved by merely adding inferential mechanisms because the language model is so limited that the relevant information never reaches the dialogue handler.

Focus: due to the constraint on active vocabulary size the recogniser can only work satisfactorily with a few sub-grammars at a time, corresponding to yes/no, meta-communication keywords, and a single subtask such as date, hour or ticket delivery. The single exception is outjourney and homejourney (T-11) and this is so only because the relevant part of the vocabulary (the airport destinations) are shared between the two. Other cases do not work (T-12).

S10-20	Where does the journey start?
U10-20	From Copenhagen to Esbjerg.
S10-21	From Copenhagen to Esbjerg. Do you want a return ticket?

Transcription 11. *Focus*. The start and end of the journey are in system focus simultaneously.

S2-46	Where does the travel start?
U2-46	Saturday.
S2-47	Sorry! Where does the travel start.

Transcription 12. *Focus*.The user responds to a question different from that asked by the system. With a sufficient language model, the system must infer which subtask is referred to (out date or home date), and if it is a value for a new subtask or a correction to a previous one.

S3-30	How many people will travel?
U3-30	2 adults and 2 children.
S3-30	2 people.

Transcription 13. *Language*. The system does not understand coordinated phrases ("and"), and selects "2" arbitrarily.

Composite phrases: in (T-13) the user does not generalise adults and children to "persons" and instead mentions each of the two sub-species. Only the number of children or the number of adults gets through to the dialogue handler.

S6-78	At which time of the day?
U6-78	They should arrive no later than 9:30.

Transcription 14. *Indirectness*. The system must infer which departures are possible, and inquire whether the user has taken into account, e.g., check-out time and transportation from the airport.

Indirectness and negation: in the current system, user answers must be simple and direct. Indirect expressions such as “the last flight” or “not arriving before 9:30” (T-14) will not work. Also, as already illustrated in (T7: S9-56), the system does not understand negations (“not” is not being recognised). However, this is irrelevant to the current task type since only non-cooperative users are likely to use such constructions.

Informed reservation

Many examples in the user test corpus indicate that the reservation task is not independent or stand-alone (cf. T-15). We hypothesise that a task extension to informed reservation is feasible requiring mainly some additional language capacity. *Informed reservation* is a reservation task in which the user can at any time ask questions related to the current subtask but not to other subtasks. As an additional extension, it might be considered to put closely related subtasks in focus together, such as date and hour which generalise to time, and also the previous subtask for repair purposes (T-16).

S3-24	At which time of the day?
U3-24	Which possibilities exist?
S3-25	Sorry, at which time of the day?

Transcription 15. *Information query*. The current system does not understand user questions.

S6-39	Friday the 10/1. At which time of the day?
U6-39	It should be on Saturday at 7:20.

Transcription 16. *Repair and focus*. The user tries to change “Friday” in addition to anwering the time-of-day question. With a better language model the dialogue handler must infer that the Saturday is the one closest to Friday, which is not necessarily the firstcoming after Friday (though this coincides here), cf. the discussion on date completion (T-5).

We have made some preliminary experiments with informed reservation [Dybkjær and Papazachariou 1995] which indicate that this kind of extension is feasible even with a very crude treatment of speech acts [Papazachariou et al. 1995]. However, there are complex inferential problems in deciding which values to transfer from information subtasks to the main reservation task.

Communication failure

Given its limited linguistic capability the system will often fail to understand the user. The user may use out-of-vocabulary phrases, assume inferences that the system cannot do, or the system may make plain misrecognitions. One problem with recognition failures is that they are hard to detect by the dialogue handler.

Below two measures are described that could be included given the current system's language and task, but at the expense of a much more elaborate dialogue history and input handling. With more complex language and task, however, the handling of communication failure—and meta-communication in general—can be expected to rapidly become more complex.

S-1	To Copenhagen.
	Do you want a return ticket?
U-1	No.
S-2	To Copenhagen.
	Is that correct?
U-2	Yes.
S-3	Do you want a return ticket?
U-3	No.
S-4	To Copenhagen.
	Is that correct?

Transcription 17. *Looping*. The Danish words for "no" and "change" are easily confused, which is what happens after U1 and U3. This transcription is reconstructed from a demonstration with the real recogniser via a standard telephone line.

Failure looping

In the task dialogue structure, looping typically occurs because of communication failure. In the current system the only loop detection is related to timeouts. To do further loop reasoning would require an extended dialogue history. Two variations of communication causing loops are:

Insufficient language: If the user is interested in discount the current system will disregard departures with no discount. In (T-8) the user wishes to override a previously expressed wish for discount and knows that the language does not allow a direct, natural formulation. Instead s/he tries to insist on the existing full-fare departure.

The specific example could be eliminated by other means than loop detection, but the point is that communication failure may happen due to insufficient language even if the recognition is fully correct.

Misrecognition: a frequent cause of loops is misrecognition either because the user uses out-of-vocabulary phrases or because of user dialect or noisy communication channels (T-17).

Score values

The transcription (T-18) shows some of the internal system module exchanges in an abbreviated form. Note that the recogniser assigns score values to the recognised word strings. Even though the scores are only meant for selecting one from a set of possible word strings, it is tempting to consider these scores as indicators of recognition uncertainty.

U4-45 It should be an outjourney between 7 and 9 in the morning, please.

```
keyed [U4-45]  itshoultbeanoutjourneybetweeense
               venandnineinthemorningplease
recognised [U4-45/Hour/score -403.0]
               in the morning
semantics [U4-45] time-of-day "morning"
```

Transcription 18. Scores. The user utterance is keyed in by a wizard (translated and adapted from the Danish including keying errors). The utterance is recognised by the text recogniser with a score of -403.0. Zero (0.0) means perfect match.

After parsing, the remaining semantics is "morning" which is only one, incidentally correct, part of the utterance. This behaviour is typical for the text recogniser when processing long utterances.

Scores	**total**	**ok**		**wrong**		**none**	
interval	#	#	%	#	%	#	%
0	798	796	99.7	0	0.0	2	0.3
[-5, -1]	32	31	96.9	1	3.1	0	0.0
[-846, -6]	157	84	53.5	54	34.4	19	12.1
total	987	911	92.3	55	5.6	21	2.1

Table 1. Scores in the user test corpus versus the semantic contents. The table distinguishes semantics which is *ok*, *wrong*, or *none* (either nothing or irrelevant). In the *ok* and *wrong* cases the semantics may be partial.

In Table 1, intervals of score values are correlated with the recognised semantic contents. We observe that for score values less than -5 the number of (partially) correct recognitions at the semantic level (84) is roughly equivalent to the number of utterances with misrecognised semantics (54) plus those with no appropriate recognised semantics (19). In other words, these cases should be treated very cautiously by the dialogue handler. It should be noted, however, that the numbers are highly system-specific and, in particular, that they were produced by a text recogniser simulating the real recogniser. A real recogniser would hardly generate word strings with scores as high as zero, the perfect match.

4 CONCLUSION

We have addressed a number of inferential issues and problems that were discovered in a user test of the Danish dialogue system. Part of the inference mechanisms described have already been implemented and others could easily be added to the system. Others again would need more complex dialogue management or are not relevant until a more sophisticated treatment of language is available and/or the tasks addressed become more complex.

Many of the inferences are closely related to the domain and are basic to system functionality. This is true of, e.g., categorisation, completion, the examples given of turn saving inferences and indeed of any kind of factual computation. However, the complexity of this sort of domain inference would not seem to bear any direct relation to the language and the task.

The inferences concerning identification and ambiguity are related to the language. In the current system, the extra-lexicon-based inferences mainly depend on the dialogue context. Examples have been given for meta-communication, communication failure, and looping.

With increased task complexity, such as moving from reservation to informed reservation, experience shows that the problems related to meta-communication, dialogue structure and information transfer will significantly increase.

One may ask why the relatively simple inferences which could be added easily, were not implemented to begin with. The answer is that we were not aware of them. Although the WOZ method is probably the most appropriate method for the development and test of dialogue models prior to implementation, it has a number of shortcomings. One of its strengths is that a human simulates the system. It is easier for the human to learn to act more or less like the intended system than it would be to implement the system and then probably have to reimplement large parts of it when the inappropriatenesses and errors are discovered. This strength is also a weakness, however, because it is very difficult for humans to simulate language capabilities at an inferior level. Most of the time, we do not tend to be conscious about our skills in making inferences. This makes it difficult for us to detect and explicitly represent the inferences we actually make during spoken dialogue. Furthermore, it is difficult to plan task scenarios that will trigger situations where the inferences are necessary. A user test or a field test of the implemented system is much more likely to reveal missing inferential mechanisms. This difficulty of detection generally contrasts with the relative ease of representing the needed inferences in a formal system.

It should be emphasised that even though inferences are important and necessary to natural dialogue, they do not by far solve all problems in spoken dialogue systems development. User errors cannot be eliminated through system inference. Note, e.g., how the system misrecognition (T-1: S3-67) passes unnoticed by the user (T-1: U3-67). In addition, many system errors are matters of dialogue design rather than of system inference design, such as ambiguous or obscure system output, missing feedback or insufficient information to users [Dybkjær et al. 1996, Vol. 3].

The categorisations and characterisations of inferences in SLDSs presented in this paper are not yet conclusive. We hope to have provided an outline of the field of problems which may serve as a basis for future work. The interesting task remains of determining more systematically the relationships between i) inferential problems, ii) application aspects such as language, domain, and task, and iii) SLDS aspects such as user model, dialogue history, and meta-communication.

REFERENCES

[Baekgaard et al. 1995) Anders Baekgaard, Niels Ole Bernsen, Tom Brøndsted, Poul Dalsgaard, Hans Dybkjær Laila Dybkjær, Jan Kristiansen, Lars Bol Larsen, Børge Lindberg, Bente Maegaard, Bradley Music, Lene Offersgaard, and Claus Povlsen: The Danish spoken dialogue project. A general overview. *Proceedings of the ESCA Workshop on Spoken Dialogue Systems, Theories and Applications*, Vigsø 30 May-2 June 1995, 89-92.

[Bernsen et al. 1995] Niels Ole Bernsen, Hans Dybkjær and Laila Dybkjær: Exploring the limits of system-directed dialogue. Dialogue evaluation of the Danish dialogue system. *Proceedings of Eurospeech '95,* Madrid, September 1995, 1457-1460.

[Dybkjær and Papazachariou 1995] Hans Dybkjær and Dimitris Papazachariou: Mixed-Initiative Dialogue. Technical report. http://www.cog.ruc.dk/~dybkjaer/mixreport/mix.html

[Dybkjær et al. 1993] Hans Dybkjær, Niels Ole Bernsen and Laila Dybkjær: Wizard-of-Oz and the trade-off between naturalness and recogniser constraints. Proceedings of EUROSPEECH ‘93, Berlin, September 1993, 947-950.

[Dybkjær et al. 1995] Hans Dybkjær, Laila Dybkjær and Niels Ole Bernsen: Markup of transcriptions. Technical report, Centre for Cognitive Science, 1995. http://www.cog.ruc.dk/projects/Dialogue/user-95/. Includes the full user test corpus online.

[Dybkjær et al. 1996] Laila Dybkjær, Niels Ole Bernsen and Hans Dybkjær: Evaluation of Spoken Dialogues. User Test with a Simulated Speech Recogniser. Report 9b from the Danish Project in Spoken Language Dialogue Systems. Roskilde University, February 1996. 3 volumes of 18 pages, 265 pages, and 109 pages, respectively.

[Fraser and Gilbert 1991] Norman M. Fraser and G. Nigel Gilbert: Simulating speech systems. Computer Speech and Language 5, 81-99, 1991.

[Papazachariou et al. 1995] Dimitris Papazachariou, Niels Ole Bernsen, Laila Dybkjær, and Hans Dybkjær: Identification of speaker actions in mixed initiative dialogue. In Laila Dybkjær (ed.): Proceedings of the Second Spoken Language Dialogue and Discourse Workshop, Dublin, April 1995. CCI Topics in Cognitive Science and HCI Vol. 8, Centre for Cognitive Science, Roskilde University, 50-57.