Information-Seeking Chat: Dialogue Management by Topic Structure

Manfred Stede and **David Schlangen**
University of Potsdam
Department of Linguistics
Applied Computational Linguistics
P.O. Box 601553
D-14415 Potsdam — Germany
{stede|das}@ling.uni-potsdam.de

Abstract

In this paper we describe the dialogue sub-genre "information-seeking chat", which is distinguished from other kinds of information-seeking dialogue (e.g. travel information) by its more exploratory and less (single) task-oriented nature. We present an approach to modelling this kind of dialogue, based on the notion of *weighted topic structures* — a single data structure that represents both the domain knowledge and the dialogue history, and we sketch an implementation of this approach in a typed dialogue system.

1 Introduction

Both theoretical analyses of dialogue and implemented dialogue systems have so far mostly focused on two main dialogue genres: strictly task-oriented dialogue (as in the travel agent domain, call routing applications, or collaborative problem solving domains), or tutorial dialogue. In this paper we describe another type of dialogue, which we call "information-seeking chat". This genre is distinguished by its more exploratory and less task-oriented nature, while still being more structured than general free conversation.

Our thesis is that this kind of dialogue can be modelled with a simple taxonomy of dialogue moves and a dialogue management (DM) strategy based on *topic structure*, where the main task of the dialogue manager is to guide the user through the pre-defined topic map. This topic map is a declarative domain model (similar to an ontology) that serves both as a representation of the domain knowledge and as a repository for the discourse history. (The model represents the discourse history insofar as during the course of the dialogue it is annotated with information about which topics have been broached or have been exhausted.) Moreover, it is the only discourse *planning* device the system uses, since it also records the effect of each utterance on the decision of which bit of information to relay, which topic to explore next. This surprisingly simple information structure can successfully model this important kind of dialogue, as we argue here, and it also makes it relatively easy to implement new applications covering other domains in this style—information about companies, for example, or more generally about structured fields of knowledge.

The remainder of this paper is organised as follows. Section 2 elaborates on the peculiarities of "information-seeking chats", and presents our taxonomy of dialogue acts. A discussion of the problems extant approaches to dialogue modelling would have with this kind of dialogue leads over to Section 3, where we describe our approach, and the prototypical implementation. After discussing in Section 4 related attempts to reduce dialogue management to representations of task-knowledge, Section 5 sketches how our dialogue manager fits in with the other modules of the system that is under development. Section 6 finally discusses evaluation issues, and further work.

2 The Dialogue Genre

2.1 Information-Seeking Chat

Imagine that you are planning next year's vacation, and that you want to find a destination that offers both cultural attractions and leisure activities. Once a candidate emerges, you now ideally want to have a conversation with a local representative, ask her a few specific questions about activities you have in mind, but also be alerted by her towards attractions you did not think about. This conversation will wander from one aspect to another, sometimes come back to a topic already mentioned, and possibly sometimes digress to the weather, or local football teams. The purpose in any event is for you to form an impression of an area that previously you knew only very little about, guided by your interests and by what you learn.

This kind of dialogue is rather different from dialogues aiming at finding *one particular* piece of information or executing one particular transaction. While it does fall under the general rubric "Inquiry Oriented Dialogue" defined by (Larsson, 2002) as shown below in (1), it is distinguished from those kinds of inquiry activities by not being driven by specific goals that can easily be decomposed into hierarchically ordered subgoals (like "find a specific train connection", which can be decomposed into "get destination, start date, etc.."). Rather, it is, at least at the beginning of the dialogue, only driven by relatively unspecific high-level goals (e.g., "tell me something about city XY.") that are not as easily decomposed.[1]

(1) "[. . .] the term Inquiry Oriented Dialogue, or IOD, will henceforth be taken to refer to any dialogue whose sole purpose is the transference of information, and which does not involve any DP assuming (or trying to make another DP assume) commitments or obligations concerning any non-communicative actions outside the dialogue." (Larsson, 2002, p. 17)

Hence, this kind of dialogue does not naturally lend itself to an approach of goal-directed hierarchical planning. Instead, it shows similarities to "smalltalk" that drifts from one aspect of the topic to another, while still being more constrained than that by having a specific, albeit very general, purpose. It is also much more a "mixed-initiative dialogue" than for example a dialogue in the well-known travel domain (as modelled in the GoDiS system (Traum and Larsson, 2003) and the DARPA Communicator systems), since both interlocutors can quite freely open up new subtopics, declare one as closed, digress, or hesitate.[2] Accordingly, this kind of dialogue requires implementation strategies quite different from those established for task- or transaction-oriented dialogue. We will propose one such strategy in Section 3, but first we briefly summarise results of a corpus study, which lends further motivation to this choice of strategy.

2.2 Dialogue Flow and Dialogue Acts

We conducted a small corpus study, collecting text dialogues (using a web-based chat tool) between a domain expert and an information-seeker. The only instruction we gave to the expert was to open the dialogue with a standard opening ("Welcome. This is Potsdam Tourist Information. How may I help you?"); the scenario for the inquirer was that they will be in Potsdam for a conference and have to decide whether they should take the weekend after the conference off and stay. All inquirers were non-locals and not familiar with Potsdam. We collected 13 dialogues, of around 17 turns each.[3] An excerpt of one of the dialogues is shown in Figure 1 (annotated with the dialogue-act-types that will be described presently).[4]

[1] Note that it is not excluded that such a dialogue might develop into a more focused, traditional task-oriented dialogue. For example, during an information-seeking chat, the inquirer might become interested in a particular offering and want to book a ticket. In our final system we plan to build in interfaces that can hand over control to other dialogue managers designed for this kind of dialogue.

[2] However, there *are* differences between the dialogue participants, and the boundaries of the roles of inquirer and expert are clear. For example, in our corpus (see next subsection) there is a tendency for the expert adviser not to cut off a topic that is being talked about, and so topic-changes mostly initiate with the user. In principle, however, both interlocutors seem to have the same range of moves available at any point.

[3] Which makes them rather short; this is possibly due to the "simulation"-nature, but hopefully does not affect the representativity of the dialogues.

[4] The collected dialogues are in German, but for reasons of space we only show here an English translation. Since the focus of the paper is on dialogue-management (and not for example on dialogue act recognition), the observations should largely be language-independent.

dialogue 008			
1	E:	Hello. This is Potsdam Tourist Information. How may I help you?	opening
2	U:	Well, I'd like to know whether a weekend in Potsdam is worthwhile?	switch-topic
3	E:	A weekend in Potsdam is always worth your while, there are lots of nice things you can do here.	tell-topic-general
4	U:	Like what?	ask-more-specific
5	E:	For example, you can make a boat trip, or a sightseeing tour. Always popular with tourists is Park Sanssouci.	tell-more-specific
6	U:	Is all of this in the inner city? Can you walk to these things, or do you have to use some kind of transport?	ask-more-attribute
7	E:	It's probably best to take the bus to the park, since it's not really in the centre. By foot it's about 20 minutes. The busses leave from the central station. This is also where the sightseeing tours start. The harbour is also only 5 minutes away from the station.	tell-spec-attribute
8	U:	Sounds good. Are there any reasonably priced hotels near the station?	switch-topic
9	E:	There's the XY-hotel close to the station. But in the centre there are also many other places where you can stay. [...]	tell-topic-general
12	U:	Are there things to do for kids as well?	switch-topic
13	E:	A popular attraction for kids is the filmpark in Babelsberg. You can get there with the tram.	tell-topic-general
14	U:	How far away is that? Which tramline?	ask-more-attribute
15	E:	It's only two stops from the station. You take the S1 to Griebnitzsee, and then you walk, it's only 10 minutes. But you can also take the bus.	tell-spec-attribute
16	U:	Only two stops, great. Is the entrance fee very high?	ask-more-attribute
17	E:	Entrance is 17 Euro for adults and 10 Euro for children.	tell-spec-attribute
18	U:	Not exactly cheap. What's on offer there?	ask-more-specific
19	E:	You can visit the Ufa-filmstudio and take part in the shows. Apart from that, it's like a theme park, with roller-coasters and stuff.	tell-spec-attribute
20	U:	Sounds good! I think I'll stay in Potsdam for the weekend then. Thank you very much for the information.	bye
21	E:	My pleasure. Have fun in Potsdam.	closing

Figure 1: An example dialogue, with dialogue-act annotation

We made the following observations:

• Firstly, the dialogues mostly seem to follow a recursive pattern of dialogue moves: the user asks for (further) information about a topic, which the expert gives, thereby proposing alternative ways of further exploring that topic. Then either one of these alternatives is taken up, and a new sequence is started beginning with this sub-topic, or else the user jumps to a different topic and begins a sequence there. This pattern is illustrated by turns 4 to 8 in Figure 1: turn 5 offers several alternative answers to the question in turn 4, and the user replies by inquiring more details. Then, in turn 8, the user ends this subsequence and jumps to a new sub-topic, which is then briefly explored. Turns 12 to 19 give an example of a sub-topic that is explored in more detail.

• Connected to this pattern is the observation that most questions (and, since this is the preferred device for changing the topic, most topic shifts) are initiated by the user: 89% of all questions come from the user, with a proportion of 35% of all turns being questions.

These are the requirements for modelling this kind of dialogue, then: a) the dialogue manager must allow for systematic exploration of a topic, while b) allowing at all times user-initiated topic shifts. It should be clear that finite-state based approaches (see e.g. (McTear, 1998)) are too rigid for these requirements; they could only model this amount of user-initiative if every state (representing a topic) not only had transitions leading to all alternative sub-topics, but also to *all* other topics as well—resulting in a number of states that is hardly practical.

The flexibility afforded by information-state-update (ISU) approaches (Traum and Larsson, 2003), on the other hand, seems better suited.[5] In

[5]Note that we are talking here about the power of the general approach of ISU-based dialogue management. The extant systems following this approach, such as GoDiS for example (Traum and Larsson, 2003), put some additional constraints on the dialogue management, by being relatively closely oriented towards template-filling and relying on system-initiative to do this, and hence are not capable of han-

this approach, dialogue is modelled as a sequence of *updates* of an (arbitrarily complex) *information state* recording discourse history as well as beliefs and plans, governed by *update-rules* that are triggered by *dialogue acts* and that produce such acts, which abstract over different linguistic realisations. Indeed, the notion of *proposals* as used above fits in nicely with what (Larsson, 2002) calls *issues under negotiation*, which are part of his Information State. In this approach, issues are represented as questions, and proposals are alternative answers to these questions. This mechanism, however, does not say anything about where the required semantic relations between questions (whether two questions are independent—what we would call a topic shift—or whether one further specifies another) is stored.

Our claim is that an underlying hierarchical organisation of topics (much like an ontology modelling the domain) is needed in any case, and that, combined with *weights* representing discourse history, this is indeed the *only* structure that is needed, obviating the need for storing explicit plans. This idea will be explored in the next section, but first we give the full list of dialogue-acts that we derived from our corpus and use in our system (Figure 2). Together with the examples given, the classes should be self-explanatory.[6] Note that we do not claim any general use for this set of acts besides describing this specific dialogue-genre, and that we have devised this set with our approach to dialogue-modelling in mind. Nevertheless, we have tested the coverage and reliability of this mark-up scheme, by getting two naive annotators to code up our dialogues. The achieved coverage of around 98% of all utterances (i.e., only 2% were marked as `other`) and the resulting κ value of .81 indicates the usefulness of the schema.

dling this kind of dialogue. That the *general* approach of ISU should be flexible enough to model it, is perhaps not surprising: it is meant to be a general framework for implementing and comparing dialogue management strategies, after all (Traum and Larsson, 2003).

[6]The acts help and garbage are only used in the implementation (they mark requests for producing a system message and recognition failure, respectively) and not for marking-up the dialogue examples.

3 The Wanderer: Dialogue Management with Topic Structures

3.1 Overview of the approach

The approach to dialogue management proposed here inherits traits from very different traditions: chatbots[7] and ISU-approaches. From the former we import the robustness and the locality of pattern-matching-based dialogue management, while the latter give us a model for abstracting from specific inputs by using dialogue-acts. In the system, we explore the chatbot-like strategy of letting local control decisions drive the dialogue forward—local decisions which, however, need access to an over-arching discourse model. This discourse model in our approach is rather different from earlier notions in that it is very closely related to the *content* model of the system. More specifically, we use a declarative, ontology-like model of domain knowledge as the central repository of information in the system; a repository that holds not only the conceptual knowledge and the associated linguistic forms, but also the information about what has been talked about already and what can or should still be put onto the agenda, represented as numerical *weights* on the topic nodes. Consequently, it is *the content* (together with, or rather, also representing the dialogue history) that is in charge of controlling dialogue flow—as opposed to other dialogue genres, where intentions and goals are in the driver's seat.

The 'information state' used in our system thus is quite simple: it does not contain *explicit* goals or beliefs or partitions; instead it contains an instantiated domain model, dynamically enriched with numerical information representing preferences for discussion.[8] As described in the previous section, we do use dialogue moves as an abstract layer between possible inputs and possi-

[7]Web-based systems for typed dialogue, using just pattern matching, but with quite sophisticated implementations. See, e.g., `http://www.alicebot.org`.

[8]We stress "explicit" here, because the weights can be seen as implicitly storing past intentions, and guiding the overall intention of "staying on topic" and "exploring the topic", and there is the implicit plan for doing the latter, namely by offering information about children nodes. Moreover, the dialogue acts of course represent communicative intentions; the point is that no explicit *planning* beyond the local decision on how to react to the last utterance is needed.

Dialogue Act	Example
ask-more	
ask-more-general	*"Can you tell me more about Potsdam's parks?"*
ask-more-specific	*"Is there a park in Potsdam?"*
ask-more-attribute	*"What are the opening times of the park?"*
reply	
reply-pos	*"OK."*
reply-neg	*"Not really, thanks."*
tell	
tell-topic-general	*"Potsdam has four large public parks."*
tell-spec-attribute	*" 'Sans, Souci.' was built in 1745."*
rule-out-topic	*"I am not that interested in parks."*
switch-topic	*"OK. What about museums in Potsdam? What's on offer there?"*
noncommittal	*"Oh well, I don't know."*
digression	*"Parks are good. I really like a good barbecue in the park."*
bye	*"Thanks. That's enough."*
opening / closing	*"Welcome. This is ..." / "Thank you and goodbye."*
help	*"Help!"*
garbage	

Figure 2: Dialogue Acts used to Describe Information-Seeking Chats

ble updates and also between update effects and possible outputs; our notion of 'update', however, is one of adjusting weights, which reflect the dialogue history as well as user's statements regarding her dis-/interest in particular branches of the domain model.

3.2 The Implementation

We realised The Wanderer using a *description logic* (DL, see e.g. (Baader and Nutt, 2003)), which has several useful properties: Being a 'structured fragment' of first-order logic, it organises knowledge in taxonomies and offers inference mechanisms dedicated to and optimised for taxonomical reasoning—in particular, *subsumption checking*. The knowledge base is split into a terminological part (concepts and relations between them) and an assertional part (specific instances of the concepts/relations in the terminology)—much like the class/instance distinction in object-oriented programming. Subsumption is computed among concept descriptions, and between instance descriptions and concepts. Hence, DL-systems offer sophisticated instance-retrieval: given an arbitrarily complex concept description, the DL finds the set of instances satisfying the description.

In our approach, the terminology part has two components: a model of the domain, and a model of linguistic utterance types. More specifically, the domain model serves as a taxonomy of *topics* that can be addressed in a conversation. In our example application, the root concept `city-topic` is partitioned into entities such as `people`, `buildings`, `parks`, `lakes` and the like, which in turn are decomposed into more specific categories; e.g. `buildings` can be `palaces`, `temples`, `museums` and so forth. Under `people` we assemble personalities of historic interest (the former kings, architects, gardeners, etc.) and those of importance for present-day life. Concepts then can be related. E.g., the domain model offers relations that link buildings to architects, to construction years, to the kings who commissioned the buildings, etc. Besides the historic perspective, buildings can also be related to things like street addresses and entrance fees (relevant for museums or movie theatres). Finally, the most prominent relations spanning different sub-domains are subsumed by a generic relation `sim-topic`, which will be explored when the system initiates a topic shift.

Note that the terminology does not include any Potsdam-specific entity—these all belong into the assertional part of the knowledge base. The terminology thus should be transferable to other cities without too much effort. Instantiations of the concepts then constitute the description of the city in question: *Sanssouci*, *Carlottenhof*, *Marmorpalais* and so on are the `palaces` in Potsdam, linked to their architects *Knobelsdorff*, *Schinkel*, etc. Similarly, specific movie theatres are linked to their respective concrete entrance fees—and so forth for

the entire model of Potsdam-related topics.

So far, we have described just static facts, with no relationship to actual linguistic utterances. In our implemented demonstrator (which aims at investigating dialogue strategy rather than linguistic processing), the system's utterances are largely pre-fabricated. Instead of writing them entirely by hand, however, we perform a semi-automatic mapping from the domain model to utterances that verbalise the facts of the model. Thus our terminology also includes a taxonomy of utterance types (essentially templates for different types of information to be transmitted), and a mapping process traverses the domain model and uses the utterance templates to "compile" the set of system utterances. Importantly, these utterances are again *instances* of the DL; their types are both the domain topic and the kind of utterance. Domain topics need not be leaves of the tree: there are for example utterances about `architects` *(The most profligate classicist Potsdam architects were Schinkel, Persius and Hesse)* as well as about the specific architect `schinkel` *(Schinkel was born in 1781 in Neuruppin)*; the former will be produced when jumping to a new general topic and the latter only when it is explored in more depth.

Turning now to the dialogue manager, the key idea is, as indicated earlier, to associate numerical weights with the topics. The weight of an instance characterises its relative salience for the next step of the conversation.[9] When initialising a dialogue session, weights are by default distributed evenly, unless the content designer has already marked some topics as more prominent than others (e.g., one of the palaces gets higher weight so it will be the starting point when the discussion turns to palaces). Similarly, the "kickoff topic" for beginning the conversation (if the system rather than the user sets the first topic) can be set by the content designer by assigning it the highest overall weight.

After initialisation, the system then moves through the following cycle: (1) get user input (simplified, as described above) — (2) classify this input into a dialogue-act and parameters — (3) choose an output utterance — (4) update the weights. The process stops when either the user ends the conversation (system identifies a BYE-act), or the system has nothing more to say, i.e. when the entire topic range has been exhausted.

The DL we use (LOOM, (MacGregor and Bates, 1987)) offers techniques from object-oriented programming, which we use for realising the response strategy of *The Wanderer*: it is a set of independent *methods* that fire when a particular combination of dialogue-act and parameter is identified in the user's input. This corresponds to the idea of local decision-making (see above) but is realised more flexibly than in chatbots, as more computation can be performed (exploiting the DL's services) to determine the optimal output utterance. The first step consists of constructing a concept description (consisting of domain topic and, if applicable, linguistic types) that is handed to Loom, whose query facility will find the set of candidate utterances. Among these, the selection is made on the basis of the weights, i.e., the highest-ranked utterance is chosen (with random choice in case of a tie). Hence it is the weight update mechanism evoked in step (4) that is responsible for steering utterance selection in order to ensure coherence.

The following types of weight-update-functions are implemented; they are used in the dialogue-act-rules as described in Figure 3.[10]

U1 Increase weight of utterance set by n%;
U2 Reduce weight of utterance set by n%;
U3 Increase weight of utterance set by the amount necessary to just outweigh all others;
U4 Reduce weight of utterance set to almost-zero.

An example shall illustrate this mechanism. In the following, we show a (constructed) user utterance, its dialogue-act-type, the effect on the weights, and the system reply:

[9] To keep the process of weight adaptation more transparent, we chose to ensure that the sum of all weights is kept constant (so it can in fact be interpreted as a probability distribution).

[10] Where 'same node / sister' in column 'SYS UTT FROM NODE' means: if there is a non-exhausted utterance left at same node, chose it; otherwise choose one from the highest-ranked sister node. In row 4: the answer to a specific attribute-question has to be looked up explicitly, and hence the weight-based selection doesn't apply. Rows 5,6: currently, the only system question to which the user REPLIES is "Do you want to learn more about TOPIC?" (the "probe question"). In 5-8, 'sim-topic' refers to the `sim-topic`-relation (mentioned earlier) that abstracts over some specific relations.

	USER DIAL.ACT	SYS UTT FROM NODE	WEIGHT UPDATE
1	ask-more	same node / sister	node and daughters: (U1 10)
2	ask-more-general	mother	mother and all her daughters: (U1 5)
3	ask-more-specific	highest-ranked daughter	all daughters: (U1 5)
4	ask-more-attribute A	response: info on A	none
5	reply-pos	same node / sister	node and daughters, sim-topics: (U1 5)
6	reply-neg	sister / any other	node and daughters: U4 sim-topics: (U2 5)
7	rule-out-topic T	highest-ranked node outside T prefer a probe question	node and daughters: U4 all probe questions: U3 sim-topics: (U2 5)
8	switch-topic to T	new topic T don't choose probe question	(old) node and daughters: U4 nodes below T: U3 intro-q's at T: U4 sim-topics-of-T: (U1 5)
9	noncommittal	same node / sister	none
10	digression	mother	all probe questions U3 node and daughters: (U2 5)
11	bye	bye	n/a
12	help	print info	none
13	garbage	same node / sister	none

Figure 3: The weight-update rules

"What about cinemas?"—(switch-topic(cinema))—(increase weight of cinema-topic and daughters, and related entertainment topics)—"*There are four cinemas in Potsdam, the Thalia, Melodie, UCI, and one in the Filmmuseum.*". | *"Where is the Melodie?"*—(ask-more-attribute(location))—(no effect on weights; retrieve attribute)—"It's on Ebert Street.". | *"Well."*—(noncommittal)—(no effect on weights, stay on topic)—"The entrance is only 4 Euros." (...)

It is clear that this mechanism has its limits. It currently cannot handle conversations where two topics are explored in parallel, and it also cannot handle complex queries like "which of the parks is closer to the central station?" (which however are also beyond the scope of most dialogue systems). While not being implemented at the moment, the use of variables (for example for recording the time of the visit) is possible, and sub-dialogues could be launched to fill them.

4 Related Work

We have already related our approach to finite-state-based and ISU-based approaches above; here we focus on the idea of reducing dialogue control to following a representation of domain knowledge. The Swedish WAXHOLM system (Carlson and Hunnicutt, 1996) goes some way in this direction. It makes use of hierarchical topic-maps as well, which however are only used for computing probabilities during topic spotting (see below for our approach to this). The "construct algebra" approach of Abella and Gorin (Abella and Gorin, 1999), used in the *HMIHY*-system, also explores this reduction-strategy. It uses so-called *constructs* that represent knowledge about the tasks that the system can handle (e.g., call forwarding, or giving billing information), which are organised in an inheritance hierarchy. Dialogue management then consists in creating such constructs and applying *dialogue motivators* on them (for filling in missing information, for example), until the constructs are satisfied (and hence the task is done). In our approach, however, it is not *know-how* that is represented but rather topical knowledge.

Finally, the German SmartKom system has recently promoted the use of ontologies in dialogue systems (Gurevych et al., 2003), mostly for coherence scoring. As we will sketch below, we also use our ontology for this, but in addition, as described above, we use it for the dialogue management as well.

5 Sketch of the other modules

Although the focus of this paper has been on the dialogue manager component of our system, we

shall now briefly describe the context in which this module works.

The matching of input to dialogue-acts is performed by pattern-matching combined with keyword spotting (the keywords being recorded on the topic-nodes). E.g., we specify patterns like "Tell me more about *TOPIC*", where *TOPIC* is a placeholder for a keyword. Together with knowledge about the current topic, a match resolves to either one of the acts from the dialogue-act family ask-more, or to a topic-shift. In case more than one keyword matches, the hypotheses are ranked using the weights, thus making double use of this device.[11] This has certain similarities with the chatbot approach of template matching; we plan to upgrade this in the future to either a linguistic analysis or a statistical model or a combination of both.

6 Summary and Further Work

We have presented a brief study of the genre "information-seeking chat", and have suggested that it has certain features distinguishing it from the kinds of information-seeking dialogues (e.g. travel information) predominantly modelled in dialogue systems, the main difference being that it is less driven by specific task-level goals ("ask about intended departure times", for example) than by the topical structure of the domain. We have proposed a taxonomy of speech acts that can describe the moves in such dialogues; and we have sketched a strategy to model such dialogues, together with an experimental implementation of that strategy. Our prototype implementation relies on a strict division between declarative domain model and dialogue management, so that moving the system to a new domain is a matter of replacing the domain model, not one of re-programming.

Besides further developing the modules of the system, we are also planning a more thorough evaluation of this component, through a Wizard of Oz-study: a three-party dialogue where the wizard classifies the user utterances into dialogue-act plus parameters, and the system produces the replies that are sent to the user. A questionnaire is used to assess the user's (dis-)satisfaction, as well as objective measures such as dialogue length, number of misunderstandings, etc. The base line will be given by a defective version of the dialogue manager (that for example makes random topic shifts), and a gold standard by evaluating human performance on the same task.

[11] While our system is based on written input at the moment, it should be possible to make the move to spoken input, by using these patterns to compile out speech recognition (SR) grammars and using the techniques of topic spotting developed in the spoken language community (see *inter alia* (Myers et al., 2000)). Of course, using automated SR would mean that some sort of error-clarification mechanism would have to be integrated, as for example described in (Schlangen, 2004), complicating the dialogue control mechanism.

References

Alicia Abella and Allen L. Gorin. 1999. Construc algebra: Analytical dialog management. In *Proccedings of the 37th Meeting of the ACL*, Maryland, USA. ACL.

Franz Baader and Wolfgang Nutt. 2003. Basic description logics. In Franz Baader et al, editor, *The Description Logic Handbook*. Cambridge UP, Cambridge.

Rolf Carlson and Sheri Hunnicutt. 1996. Generic and domain-specific aspects of the waxholm NLP and dialog modules. In *Proceedings of the 4th International Conference on Spoken Language Processing*, Philadelphia, USA.

Iryna Gurevych, Robert Porzel, Elena Slinko, Norbert Pfleger, Jan Alexandersson, and Stefan Merten. 2003. Less is more: Using a single knowledge representation in dialogue systems. In *Proceedings of the HLT-NAACL Workshop on Text Meaning*, Edmonton, Canada, May.

Staffan Larsson. 2002. *Issue-based Dialogue Management*. Ph.D. thesis, Goteborg University, Goteborg, Sweden.

Robert MacGregor and Robert Bates. 1987. The LOOM knowledge representation language. TechReport ISI-87-188, USC/ISI, Berkley, California.

Michael McTear. 1998. Modelling spoken dialogues with state transition diagrams: Experiences with the CSLU toolkit. In *Proceedings of the 5th International Conference on Spoken Language Processing (ICSLP-1998)*, Sydney, Australia.

Kary Myers, Michael Kerns, Satinder Singh, and Marilyn A. Walker. 2000. A boosting approach to topic spotting on subdialogues. In *Proceedings of the 17th International Conference on Machine Learning (ICML-2000)*, Stanford, USA.

David Schlangen. 2004. Causes and strategies for requesting clarification in dialogue. In *Proceedings of the 5th Workshop of the ACL SIG on Discourse and Dialogue*, Boston, USA, April.

David Traum and Staffan Larsson. 2003. The information state approach to dialogue management. In Jan van Kuppevelt and Ronnie Smithp, editors, *Current and New Directions in Discourse and Dialogue*, pages 325–353. Kluwer, Dordrecht, The Netherlands.