Evaluating combinations of dialogue acts for generation

Simon Keizer & Harry Bunt
Department of Communication and Information Sciences
Faculty of Humanities
Tilburg University, The Netherlands
{s.keizer,h.bunt}@uvt.nl

Abstract

We will discuss an approach to dialogue act generation that reflects the multidimensionality of communication. Dialogue acts from different dimensions in the taxonomy used are generated in parallel, resulting in a buffer of candidates. The main focus of the paper will be on an additional process of evaluating these candidates, resulting in definitive combinations of dialogue acts to be generated. This evaluation process is required to deal with the interdependencies there still are between dimensions, and involves logical, strategic and pragmatic considerations.

1 Introduction

In natural language dialogue, participants have to take into account several aspects of the communicative process in interpreting and generating utterances. Besides asking questions, giving instructions, and putting requests, related to some underlying task, the dialogue partners should also keep track of the status of processing each other's utterances, deal with interaction management issues such as turn-taking and topic management, and with social aspects of communication like greeting and apologising.

These different aspects of communication are reflected in the dialogue act taxonomy[1] as developed within Dynamic Interpretation Theory (DIT) (Bunt, 2000). This taxonomy consists of currently 10 *dimensions*, each containing communicative functions addressing one of the aspects. The taxonomy allows for multifunctional utterances, in the sense that every utterance in a dialogue gets assigned at most one function from each dimension.

This multidimensionality suggests that in generating dialogue behaviour, participants select dialogue acts from different dimensions simultaneously and independently, and then combine them into multifunctional utterances. However, this combination process is not straightforward. There are dependencies between dialogue acts from different dimensions that have to be taken into account. For example, dialogue acts may be in conflict with each other, so only one of them can be generated, or a dialogue act may already be implied by another and a decision has to be made whether or not to explicitly generate it. Moreover, not every combination of dialogue acts can be realised in a multifunctional natural language utterance, but only in subsequent utterances.

In this paper, we will discuss an approach to dialogue act generation that both acknowledges the multidimensionality of communication and deals with the problem of interdependencies between dimensions. In this approach, we distinguish a separate process of evaluating candidate dialogue acts that have been generated on the basis of dimensions in isolation. In general, the evaluation involves 1) resolving logical conflicts between dialogue acts, 2) strategic and pragmatic considerations for prioritising dialogue acts, and 3) language generation and non-verbal aspects of realising dialogue acts. Here, we will particularly focus on the first and second phases of the evaluation.

We have developed a dialogue manager using

[1] http://let.uvt.nl/general/people/bunt/docs/dit-schema2.html

Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue, pages 158–165,
Antwerp, September 2007.

the abovementioned multidimensional taxonomy in generating dialogue behaviour (Keizer and Bunt, 2006). Dialogue acts from different dimensions are generated in parallel through several *Dialogue Act Agents* operating on the system's information state. Each agent is associated with one specific dimension, and generates contributions related to that dimension only. An additional *Evaluation Agent* takes care of constructing combinations of dialogue acts for actual generation in system utterances. This multi-agent design allows to experiment with different dialogue strategies and styles of communication, having their specification concentrated in the Evaluation agent. A similar argument is used in (Stent, 2002), discussing a dialogue manager consisting of three independent agents operating in parallel. The 'organisation of conversation acts into coherent and natural dialogue contributions' is taken care of by one of these agents, called the Generation Manager. The distinction between the processes of 'contribution planning' and 'contribution structuring' has some similarity with our distinction between the dialogue act agents (over-)generating dialogue acts and the Evaluation agent selecting and combining the resulting candidates. However, contribution structuring deals with interrelationships between the *levels* of conversation acts, whereas our Evaluation agent operates on the basis of interdependencies between *dimensions* of dialogue acts.

Another multi-agent approach to dialogue management is taken in JASPIS (Turunen et al., 2005), a speech application architecture for adaptive and flexible human-computer interaction. The system uses so-called 'Evaluators' that determine which agents should be selected for different interaction tasks, based on evaluation scores. Part of an Evaluator's task may be to decide on a particular dialogue strategy by selecting a corresponding dialogue agent. Our approach of evaluation also involves issues of dialogue strategy, but this is not carried out by selecting between contributions from alternative agents for the same task.

2 Dynamic Interpretation Theory

In Dynamic Interpretation Theory (DIT) (Bunt, 2000), utterances in a dialogue are modelled in terms of combinations of dialogue acts that operate on the information state of the dialogue participants. A dialogue act has a *semantic content*, expressing what the act is about, and a *communicative function* specifying how the semantic content is to be taken to change the information state of the addressee. Communicative functions are organised in a 10-dimensional taxonomy, in which the dimensions reflect different aspects of communication that speakers may address simultaneously in their dialogue behaviour. In each utterance, several dialogue acts can be performed, each dialogue act from a different dimension. The overview below shows a layered structure in which the dimensions are given in boldface italic. So, besides the *Task* dimension, the taxonomy provides for several *Dialogue Control* dimensions, organised into the layers of *Feedback*, *Interaction Management (IM)* and *Social Obligations Management (SOM)*.

- ***Task/domain***: acts that concern the specific underlying task and/or domain;
- **Dialogue Control**
  - **Feedback**
    - ***Auto-Feedback***: acts dealing with the speaker's processing of the addressee's utterances; contains positive and negative feedback acts on different levels of understanding (see below);
    - ***Allo-Feedback***: acts dealing with the addressee's processing of the speaker's previous utterances (as viewed by the speaker); contains positive and negative feedback-giving acts and feedback elicitation acts on different levels of understanding (see below);
  - **Interaction management**
    - ***Turn Management***: turn accepting, giving, grabbing, keeping;
    - ***Time Management***: stalling, pausing;
    - ***Partner Processing Management***: completion, correct-misspeaking;
    - ***Own Processing Management***: error signalling, retraction, self-correction;
    - ***Contact Management***: contact check, indication;
    - ***Topic Management***: topic introduction, closing, shift, shift announcement;
  - ***Social Obligations Management***: initiative and response acts for salutation, self-introduction, gratitude, apology, and valediction.

A participant's information state in DIT is called his *context model*, and contains all information considered relevant for his interpretation and generation of dialogue acts. A context model is structured into several components:

1. *Linguistic Context*: linguistic information about the utterances produced so far (an

extended dialogue history); information about planned system dialogue acts (dialogue future);

2. *Semantic Context*: contains current information about the task/domain, including assumptions about the dialogue partner's information;
3. *Cognitive Context*: the current processing states of both participants, expressed in terms of a level of understanding reached (see below)
4. *Physical and Perceptual Context*: the perceptible aspects of the communication process and the task/domain;
5. *Social Context*: current communicative pressures.

In keeping track of the participants' processing states in the cognitive context, four levels of understanding are distinguished: 1) *perception*: the system was able to hear the utterance (successful speech recognition), 2) *interpretation*: the system understood what was meant by the utterance (successful dialogue act recognition), 3) *evaluation*: the information presented in the utterance did not conflict with the system's context (successful consistency checking), and 4) *execution*: the system could act upon, do something with, the utterance (for example, answering a question, adopting the information given, carrying out a request, etcetera).

These levels of understanding are also used in distinguishing different types of auto- and allo-feedback dialogue acts, each for signalling processing problems on a specific level.

3 Dialogue act generation

The architecture of the dialogue manager is given in Figure 1. Central is the context model, currently containing four of the five components defined in DIT, the Physical & Perceptual Context currently considered to be irrelevant four our purposes. The Context Manager takes care of updating the context model during the dialogue with every new utterance being produced, be it a user or a system utterance. Both for interpretation of user utterances and generation of system utterances, the dialogue manager makes use of the multidimensional taxonomy. User utterances are analysed and eventually interpreted in terms of sets of dialogue acts by the Dialogue Act Recogniser (DAR), the results of which are then written in the dialogue history. The Context Manager then takes care of updating the entire context model and checking it for inconsistencies.

For dialogue act generation, separate *Dialogue Act Agents* are used, that each take care of generating acts from a particular dimension of the taxonomy. These generated acts are recorded as candidates in the dialogue future of the Linguistic Context. Currently, five dialogue act agents have been implemented, covering the five most relevant dimensions for our purposes.

The *Task Agent* is associated with the task/domain dimension: it is responsible for the underlying task itself. In the case of question answering (QA), it basically generates answers to domain questions, where it can turn to either a structured database with domain information, or to a QA module taking self-contained domain questions, to retrieve the information to be contained in the answers it generates. The Task Agent operates primarily on the information in the Semantic Context.

The *Auto-FB Agent* monitors the own processing state as stored in the Cognitive Context, making sure that the system correctly understood the user's utterances. The agent generates negative auto-feedback acts in case of processing problems and occasional positive feedback in case of successful processing.

Similarly, the *Allo-FB Agent* monitors the partner's processing state, also in the Cognitive Context. It generates positive and negative feedback concerning the extent to which the user understood the system correctly.

The *SOM Agent* takes care of the social aspects of the communication. It generates reactive SOM acts to release reactive pressures in the social context (created by initiative SOM acts by the user). It can occasionally generate initiative SOM acts such as apologies (for example, after repeated processing problems).

Finally, the *TimeM Agent* generates pausing acts in case the system wants to gain time in order to perform some task, like retrieving information for answering a domain question.

Although the dimensions of the taxonomy are supposed to be orthogonal, i.e., dialogue acts in a dimension are selected independently, there are still

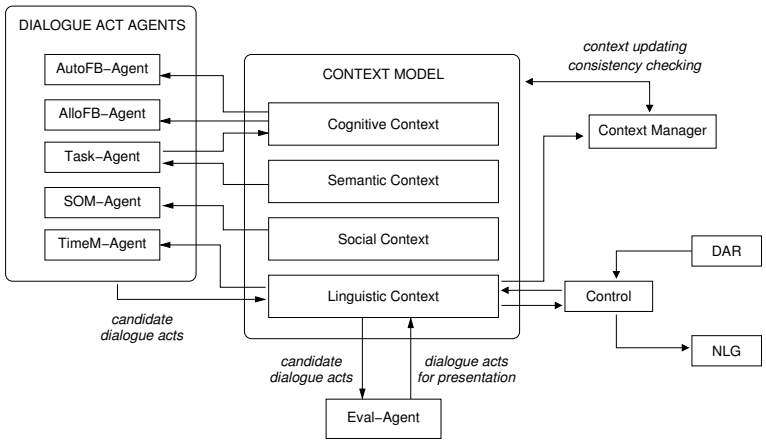

Figure 1: Architecture of the dialogue manager.

dependencies between the dialogue acts that have to be taken into account when combining them for actual generation in multifunctional system utterances. Therefore, an addition *Evaluation agent* is introduced that takes care of evaluating these candidate dialogue acts and decide on definitive combinations of dialogue acts for generation.

3.1 Design of the Evaluation Agent

The procedure for the Evaluation Agent to schedule combinations of dialogue acts from the list of candidates is subdivided into three phases. In the first phase, the dialogue act candidates are evaluated for any inherent dependencies among them. Dialogue acts from different dimensions may be in conflict with each other, so only one of them can be selected for generation, and the other has to be cancelled. The choice of which one to select and which one to cancel is based on some priority ordering among dialogue acts.

The occurrence of dialogue acts having a logical conflict implies that there is some inconsistency in the context model. Although this is undesirable and should be avoided in the design of the dialogue manager, it is nevertheless preferable to design the Evaluation Agent in such a way that it can deal with *any* combination of dialogue acts, irrespective of how the candidates were generated on the basis of the context model.

Moreover, the context model does allow for some type of inconsistency, and therefore, the generation of conflicting candidate dialogue acts. During the updating of the context model with new utterances, new information that has not been successfully integrated in the context model yet, gets stored in what is called the *pending context*, and therefore might be conflicting with the definitive context. Once the context manager has detected such an inconsistency, it records an evaluation level processing problem in the cognitive context, which could trigger the generation of a corresponding auto-feedback act. Inconsistencies might also occur within the pending context itself.

In the second phase, the remaining list of non-conflicting candidates is evaluated from a pragmatic and dialogue strategic point of view. In some circumstances, depending on the nature of the underlying task and the communicative setting, it makes more sense to postpone certain dialogue acts and give priority to others.

One type of strategic consideration is related to the planning of task acts and does not involve the combination of dialogue acts from different dimensions. For example, if the system has several questions to ask to the user, it has to decide whether to combine these questions in a single turn, or to ask them one at a time, in a particular order, distributed over several turns. The latter strategy is the more conservative one, and is used in situations where the risk of misunderstandings is higher, like in noisy environments or where the quality of speech recognition is limited.

Another strategic issue involves the choice whether or not to explicitly produce a dialogue act

that is already implied by the other candidates. For example, a positive auto-feedback act does not need to be generated explicitly, if an answer gets generated that already implies this feedback. However, in some circumstances, there might be good reasons for explicitly performing the implied dialogue act anyway.

A third evaluation issue related to strategic considerations is that of dialogue acts being triggered by reactive pressures, e.g., a thanking down-player ("you're welcome"). Such dialogue acts have to be either generated 'immediately', or not at all: they cannot be postponed. If the system is to behave very socially and there is less need for efficiency, it should generate these response social acts more frequently. This is also true for the generation of initiative social acts, like apologies and thanking.

Finally, in the third phase, combinations of dialogue acts are selected that can actually be realised in multifunctional system utterances. Some combinations of dialogue acts may not carry any logical conflicts, but the particular natural language may not provide a multifunctional utterance for the dialogue acts to be realised. For example, a question in one dimension cannot be combined with an inform in another dimension using one single utterance, because the question requires an interrogative and the inform a declarative sentence.

Besides the construction of multifunctional utterances, some of the dialogue acts can also be realised in a non-verbal manner, for example by means of animations on the graphical user interface of the system, or by means of gestures made by the system if it is an embodied virtual agent.

4 Logical conflicts

Suppose the list of candidates contains both a dialogue act with an answer function (WH-Answer, YN-Answer, etc.) and a negative auto-feedback act on the level of perception or interpretation. Clearly, it would be absurd to generate both dialogue acts, as the answer also implies (overall) positive feedback. One exception would be that the negative feedback act would be about a different utterance in the dialogue history than the question the answer referred to. In that case, either the feedback act or the answer has to be specific about which utterance in the history it responds to. The dialogue act combination should be realised in an utterance following the pattern "<wh-answer>, but <neg-feedback>".

Combining a negative auto-feedback act on execution level and an answer dialogue act also leads to a logical conflict, since an answer implies successful execution(-level processing) of the corresponding question. Giving an answer and at the same time signalling it could not find an answer is inconsistent. Note that it might be the case that the Task Agent found one or more answers to a question, but decided they were not reliable enough to present them to the user. Such answers might be stored somewhere by the Task agent, and possibly generated after all later in the dialogue, but initially they are not candidate dialogue acts.

As in the case of answers, all dialogue acts that have an aspect of referring back to some previous utterance (or, in terms of the DAMSL dialogue act annotation scheme (Allen and Core, 1997), that are 'backward-looking'), imply overall positive feedback and hence conflict with negative auto-feedback in the ways indicated above. In the DIT taxonomy, this is the case for any type of allo-feedback, for reactive SOM acts (react-greet, apology-downplay, etc.) and for dialogue acts with a communicative function such as Agreement, Correction, and Address Request.

5 Strategic issues

Given a list of dialogue act candidates that have no logical conflicts among them, it is still not just a matter of simply mapping them onto a multifunctional utterance. Depending on the situation, it might be strategically or pragmatically preferable to give priority to some dialogue acts and postpone or even cancel others.

Whereas the relative priorities of dialogue acts for dealing with conflicts are strict in the sense that they should ensure rational behaviour (which means not producing conflicting dialogue acts, nor giving priority to a dialogue act implying positive feedback over a negative auto-feedback act), those of non-conflicting dialogue acts can be adjusted for implementing different dialogue strategies and styles of communication.

5.1 Negative auto-feedback

If the system encounters processing problems during the dialogue, it should try to solve these problems, before attending to any other aspects. So in general, negative auto-feedback acts should be given priority over all other dialogue acts.

As we have seen in the previous section, combinations of answers and negative auto-feedback on the level of either perception or interpretation give a logical conflict. However, combinations of answers and negative feedback on the level of evaluation do not. The Task Agent can be triggered by a new user goal, even if this is part of the pending context only. This is the case if the dialogue act recogniser was able to detect a question about the domain in the user utterance, but the context manager did not check this new information for consistency with the context model yet, or already detected an inconsistency (i.e., an evaluation problem was encountered). Now, the candidates list could contain both an answer to the question and a negative auto-feedback act on the level of evaluation.

The example dialogue fragment below illustrates such a situation in which only the feedback act is selected for the eventual system utterance. The dialogue acts indicated reflect the system's interpretation of the user's utterances. The system encountered a conflict in his context model, because it believes that the user can see the 'send button' (after U0), and therefore knows where it is, but it should also believe that the user wants to know where the send button is (after U2). This conflict makes the Auto-feedback agent generate a negative auto-feedback act on evaluation level, whereas the recognised user goal in U2 triggers the Task agent to construct an answer. In the example, the Evaluation agent selects only the feedback act for generation.

- U0: *I see the send button.* INFORM(see_sbutton)
- S1: *okay.* POS-AUTO-FEEDBACK-EXE
- U2: *where is the send button?* WHQ(loc,sbutton)
- S3: *but you just told me you saw the send button!* NEG-AUTO-FEEDBACK-EVAL

The answer to U2 is kept in the candidates list until it is clear whether the system had misinterpreted U0 or U2. In the following dialogue continuation, the user in U4 corrects the system in his interpretation of U0, and hence, the answer can be generated:

- U0: *I see the send button.* INFORM(see_sbutton) [user intended INF(need_sbutton)]
- S1: *okay.* POS-AUTO-FEEDBACK-EXE
- U2: *where is the send button?* WHQ(loc,sbutton) [user intended WHQ(loc,sbutton)]
- S3: *but you just told me that you saw the send button!* NEG-AUTO-FEEDBACK-EVAL
- U4: *no, I told you that I* needed *it.* NEG-ALLO-FB-INT; INF
- S5: *oh, hold on ... the send button is on the bottom right.* POS-AUTO-FB-EXE; PAUSE; WHA(loc,sbutton,bottomr)

Alternatively, the system misinterpreted U2, in which case the answer can be cancelled. In the dialogue continuation below, the user in U4 corrects the system in his interpretation of U2, and hence, the answer has to be replaced:

- U0: *I see the send button.* INFORM(see_button) [user intended INF(see_sbutton)]
- S1: *okay.* POS-AUTO-FEEDBACK-EXE
- U2: *where is the send button?* WHQ(loc,button) [user intended WHQ(loc,pbutton)]
- S3: *but you just told me that you saw the send button!* NEG-AUTO-FEEDBACK-EVAL
- U4: *no, I wanted to know where the* print *button is.* NEG-ALLO-FB-INT; IND-WHQ(loc,pbutton)
- S5: *oh, hold on ... the print button is on the bottom left.* POS-AUTO-FB-EXE; PAUSE; WHA(loc,pbutton,bottoml)

In the above examples, only the negative feedback act was selected for generating S3 and the answer was cancelled. However, the system could also follow an alternative strategy of generating both the negative feedback evaluation and the answer, which would result in something like "the 'send button' is on the bottom right, but didn't you just tell me you saw it?".

5.2 Negative allo-feedback

If the system, after processing a user utterance, has reason to believe that the user did not correctly understand the system's previous utterance, then this last user utterance may not be so relevant anymore. Any candidate dialogue acts triggered by changes in the context model due to this user utterance will not be so relevant either. Therefore, dealing with the user's processing problems should get priority

over any other aspects. In general, negative allo-feedback acts should be given priority over other dialogue acts, except for negative auto-feedback acts.

In the example dialogue fragment below, user and system are discussing a music concert by the Borodin Quartet. The system asks a question and the user responds with a return question which, to the system, seems unrelated. After processing U1, the system could conclude that he misinterpreted the user, because it expects some answer in the form of numerical information only. In that case, no answer would be generated as a candidate dialogue act. Only a negative auto-feedback act on interpretation level, possibly in combination with a request in the task dimension would be generated, resulting in system utterance (S2).

- S0: *how many tickets do you want?* WHQ
- U1: *how much is the Kronos Quartet concert?* WHQ
- S2: *Sorry, I do not understand what you mean.* APO;NEG-AUTO-FB-INT
  Please indicate the number of tickets you want REQ

  S2a: *No, I would like to know the number of tickets you want* NEG-ALLO-FB-INT

  S2b: *The Kronos Quartet concert is 30 euro,* POS-AUTO-FB-INT; WHA
  but I asked about the Borodin Quartet. allo-fb:INF

  S2c: *The Kronos Quartet concert is 30 euro.* WHA

Another scenario would be that the system successfully interpreted U1 as a domain question, but concludes that the user must have misinterpreted S0. This causes the generation of two candidate dialogue acts: a negative allo-feedback act on interpretation level, and an answer to U1. The particular strategy of the system will determine whether only the feedback act will be generated (S2a), both the feedback act and the answer (S2b), or even only the answer (S2c).

5.3 Scheduling task acts

After dealing with any processing problems, the underlying task should be the most important thing to attend to, so dialogue acts in the task dimension should get the highest priority, after negative auto- and allo-feedback acts.

On the basis of the user's input, the generation of several task-oriented dialogue acts can be triggered at once. Some user question or request could trigger several questions the system needs the user to answer before he can answer the question or carry out the request. In the case of several task-oriented dialogue acts, the relative priorities of these candidates are based on task-specific considerations. This could be based on some preferred, logical order in which subtasks should be carried out; in route-planning for example, it might be preferable to ask for the destination location before asking for the date on which the user wants to travel.

5.4 Positive auto-feedback

Every time the system reaches some level of successful processing a user utterance, a positive auto-feedback act signalling this to the user can be triggered. However, actually generating this dialogue act in all of these cases leads to a kind of communicative behaviour that can be experienced by the user as rather annoying. Instead, positive feedback should be generated only occasionally, with a frequency depending on the specific communicative setting.

In the case of dialogues involving the transfer of important information such as credit card numbers, it is desirable to give more positive feedback, but in the case of more informal dialogues, too much positive feedback should be avoided. Although the extent to which positive auto-feedback is given can be taken care of by the Auto-feedback Agent in generating candidates, it is also a matter of evaluating such acts against the other dialogue act candidates. In particular, positive feedback is often already implied by other acts, and therefore does not necessarily have to be generated explicitly.

Also in the case of train table information, like in the dialogue fragment below, giving positive feedback can be a good strategy. After U2, the system has gathered enough information from the user in order to answer his original (Indirect WH-)Question U0. In S3, the system generates this answer, thereby implying positive feedback about U2. However, successful processing of U2 also results in an auto-feedback candidate act that might be generated explicitly as well, as is the case in S3' or S3". In these cases, generating the feedback act reflects a strategy of implicit verification.

- U0: *I'd like to know when the next train to Amsterdam is*

leaving. IWHQ

- S1: *From where are you travelling?* WHQ
- U2: *From Tilburg.* WHA
- S3: *The next train leaves at 10:30h from platform 1.* WHA

S3': *So you want to go from Tilburg to Amsterdam. The next train leaves at 10:30h from platform 1.* POS-AUTO-FB-EXE; WHA

S3": *The next train from Tilburg to Amsterdam leaves at 10:30h from platform 1.* POS-AUTO-FB-EXE WHA

Another typical example of generating positive auto-feedback in combination with other acts is in the dialogue fragment below. The system asks the user a question (S0), but he is not happy about the answer given by the user (U1):

- S0: *When do you want to go?* WHQ
- U1: *I want to go to Amsterdam.* INF
- S2: *Okay, but* when*?* POS-AUTO-FB-EXE NEG-ALLO-FB-INT; WHQ

In S2, the system gives negative allo-feedback about the user's interpretation of S0 and positive auto-feedback about U1. Only after successful interpretation of the previous utterance (U1) as an answer to his question, the system may conclude that the user did not correctly understand the original question (S0).

5.5 Styles of communicative behaviour

The extent to which positive feedback acts are generated, is also a matter of communication style, besides the strategic considerations behind it. Communication style is also reflected through the generation of both initiative and reactive SOM acts. In more formal, task-oriented dialogues, the generation of apologies for example should be kept to a minimum, whereas in more informal dialogues, apologies can make the system's behaviour more natural and therefore, pleasant to the user.

Again, the dialogue act agent responsible for the generation of these acts could take care of the frequency in which these are actually generated, but this also depends on the other available candidates, making it an issue for the Evaluation Agent as well. For example, apologies can be used in combination with negative feedback acts, but their impact in utterances like "sorry?" is not as high as in utterances like "I'm sorry, I did not hear what you were saying".

6 Conclusion and future work

We have discussed an approach to dialogue act generation reflecting the multidimensionality of communication. Particular focus was on the problem of dealing with interdependencies between dialogue acts from different dimensions that have been constructed independently. Giving priority to some dialogue acts and postponing or cancelling others involved logical, strategic and pragmatic considerations, besides specific language generation issues that we did not discuss. A separate process of evaluating candidate dialogue acts allows for implementing different dialogue strategies and communication styles in the dialogue manager.

An interesting topic for future research would be to look at the possibility to assess the (relative) priorities among candidate dialogue acts from data. An advantage of this would be that one could easily adjust the dialogue manager for different types of dialogue (both in terms of the underlying task and style of communication) by reassessing the priorities with appropriate data.

Acknowledgement

This research is carried out in the IMIX-PARADIME project, funded by the Dutch national research foundation NWO.

References

J. Allen and M. Core. 1997. Draft of DAMSL: Dialog Act Markup in Several Layers. Dagstuhl Workshop. `http://www.cs.rochester.edu/research/cisd/resources/damsl/`.

H. Bunt. 2000. Dialogue pragmatics and context specification. In H. Bunt and W. Black, editors, *Abduction, Belief and Context in Dialogue*, Studies in Computational Pragmatics, pages 81–150. John Benjamins.

S. Keizer and H. Bunt. 2006. Multidimensional dialogue management. In *Proceedings of the SIGdial Workshop on Discourse and Dialogue*, pages 37–45.

A.J. Stent. 2002. A conversation acts model for generating spoken dialogue contributions. *Computer Speech and Language, Special Issue on Spoken Language Generation*, 16(3–4):313–352.

M. Turunen, J. Hakulinen, K.-J. Räihä, E.-P. Salonen, A. Kainulainen, and P. Prusi. 2005. An architecture and applications for speech-based accessibility systems. *IBM Systems Journal*, 44(3):485–504.