Dialogue Simulation and Context Dynamics for Dialogue Management

Simon Keizer and **Roser Morante**
Department of Communication and Information Sciences
Faculty of Humanities
Tilburg University, The Netherlands
{s.keizer,r.morante}@uvt.nl

Abstract

In this paper we describe DISCUS, a research tool for developing a context model and update algorithm for dialogue management. The model builds on Dynamic Interpretation Theory (DIT), in which dialogue is modelled in terms of dialogue acts operating on the information state of the dialogue participants. On the basis of dialogue act specifications of both system and user utterances, DISCUS performs the update of the system's context model. The context model is structured into several components and contains complex elements involving the beliefs and goals of both system and user. We will present simulations of two dialogues, one for demonstrating the context update model, and another in which the system utterances are generated automatically.

1 Introduction

DISCUS (Dialogue Simulation and Context Update System) is a research tool for simulating dialogues between a user and a dialogue system. On the basis of dialogue act specifications of both system and user utterances, DISCUS executes an algorithm for updating the system's context model and displays the results on the screen. The tool is used to test, experiment with, and further develop the context model and update algorithm, by abstracting away from the processes of dialogue act recognition and generation in the dialogue system and focusing on the context dynamics. The model is integrated in the dialogue manager of an interactive question answering system.

The context update algorithm is built on Dynamic Interpretation Theory (DIT), (Bunt, 2000), in which dialogue utterances are interpreted as having intended context–changing effects that are determined by the dialogue act(s) being performed with the utterance. So, generally speaking, we follow an approach that fits in the Information State Update paradigm of dialogue modelling (Traum and Larsson, 2003), but with a strong emphasis on dialogue acts and an information state representation that goes beyond a dialogue history and task-specific information.

Dialogue acts in DIT are organised in a multidimensional dialogue act taxonomy, which means that an utterance gets at most one dialogue act from each dimension. The dimensions reflect different aspects of communication that can be addressed simultaneously, such as the underlying task itself, but also the aspect of how the participants were able to process each other's utterances (auto- and allo-feedback), or aspects of interaction-management like turn-taking and topic-management, or social aspects like greetings and apologies. Following this multidimensional organisation of the taxonomy, DISCUS allows to select several dialogue acts from different dimensions to represent a single, multifunctional, utterance.

The starting point for the model for context update are the preconditions of the dialogue acts, which represent the motivation and ability for an agent to perform a dialogue act. These preconditions are represented in terms of beliefs and goals in the information state of the speaker; in that sense we follow an approach that is similar to the BDI (Beliefs, Desires, Intentions) paradigm (Allen and Perrault, 1980). The information state of a dialogue system is represented in its context model, which is structured into different components, representing different kinds of information, such as an extended dialogue history and future (the 'linguistic context'), information about the underlying task (the 'semantic context'), the participants' states of processing each other's utterances (the 'cognitive context'), and information about communicative pressures (the 'social context').

The structure of the context model can be employed in the process of dialogue act generation: for the generation of dialogue acts in specific dimensions, only specific components in the context are relevant.

After discussing the theoretical background of DIT (Section 2) and the concrete specification of the context model used and how it is updated (Section 3), we will discuss the simulation in DISCUS of an example dialogue (Section 4). In Section 5, we discuss a second simulation, but now one in which the system acts are no longer simulated, but generated automatically.

2 DIT

In Dynamic Interpretation Theory (DIT) (Bunt, 2000), a dialogue is modelled as a sequence of utterances expressing sets of *dialogue acts*. These are semantic units, operating on the information states of the participants. Formally, a dialogue act in DIT consists of a *semantic content* and a *communicative function*, the latter specifying how the information state of the addressee is to be updated with the former upon understanding the corresponding utterance. Communicative functions are organised in a taxonomy[1] consisting of 10 *dimensions* (Bunt, 2006) that reflect different aspects of communication speakers may address 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/domain* 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;
    - ***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;
  - **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, contact indication;
    - ***Topic Management***: topic introduction, closing, shift, shift announcement;
  - ***Social Obligations Management***: salutation, self-introduction, gratitude, apology, 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 in the dialogue so far (a kind of 'extended dialogue history'); information about planned system dialogue acts (a 'dialogue future');

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

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 Context specification and update model

The context model we propose follows the general structure according to DIT as described in the previous section. In Figure 1, a feature structure representation is given of our context model. Currently, information about the physical and perceptual context is not relevant for the type of dialogues and the underlying task we consider, and therefore is left out.

The Linguistic Context contains features for storing dialogue acts performed in the dialogue so far: *user_utts* and *system_utts*. In addition, *topic_struct* and *conv_state* contain information about the topical and conversational structure. The other two features in the linguistic context are related to the generation of dialogue acts (see Section 5). The feature *candidate_dial_acts* stores the dialogue acts that are generated by separate agents responsible for dialogue acts from a specific dimension in the taxonomy. The feature *dial_acts_pres* stores the current combination of dialogue acts available for direct presentation as a multifunctional system utterance.

The Semantic Context contains information related to the underlying task, in our case interactive question answering. The feature *task_progress* allows to distinguish between different stages of performing the task. In the case of an interactive question answering system containing separate QA modules that take self-contained questions as input, we distinguish the states of composing a self-contained question for the QA modules (*comp_quest*), waiting for QA results after submitting a question (*quest_qa*), evaluating the QA results (*answ_eval*), and discussing the results with the user (*user_sat*). Besides this task-specific feature, there is a feature *user_model* containing information about the user's beliefs and goals concerning the task-domain. For question answering, this information can be interpreted as a specification of the user's information needs (as built up through the questions asked by the user), and of the user's current knowledge about the domain (as built up through the answers and other information given by the system).

The Cognitive Context contains two features, representing the processing states of the system (*own_proc_state*) and the user (*partner_proc_state*) as viewed by the system. Both contain two features: one indicating whether or not a processing problem was encountered, and if so, on which level of processing this happened, and one containing information about the user's beliefs and goals related to the processing state. The Cognitive Context also has a feature *common_ground*, containing beliefs the system believes to be mutually believed. Finally, a feature *belief_model* is used, containing all (non-mutual) beliefs in the context model. This feature is used to have a reference to all beliefs in one place, making part of the update mechanism more convenient. These beliefs have cross-links to other parts of the context model, wherever appropriate (e.g., to the user model of the Semantic Context).

The Social Context is specified in terms of communicative pressures; currently, we only use one feature indicating whether or not a reactive pressure exists for performing a social obligations

$$\left[\begin{array}{ll}
\mathit{LingContext} : & \left[\begin{array}{l}
\mathit{user\_utts} : \langle \mathit{last\_user\_dial\_act} = \mathit{uda}_0, \mathit{uda}_{-1}, \mathit{uda}_{-2}, \ldots \rangle \\
\mathit{system\_utts} : \langle \mathit{last\_system\_dial\_act} = \mathit{sda}_0, \mathit{sda}_{-1}, \mathit{sda}_{-2}, \ldots \rangle \\
\mathit{topic\_struct} : \langle \mathit{referents} \rangle \\
\mathit{conv\_state} : \mathit{opening} | \mathit{body} | \mathit{pre} - \mathit{closing} | \mathit{closing} \\
\mathit{candidate\_dial\_acts} : \ldots \\
\mathit{dial\_acts\_pres} : \ldots
\end{array}\right] \\
\mathit{SemContext} : & \left[\begin{array}{l}
\mathit{task\_progress} : \mathit{comp\_quest} | \mathit{quest\_qa} | \mathit{answ\_eval} | \mathit{user\_sat} \\
\mathit{user\_model} : \langle \mathit{beliefs} \rangle
\end{array}\right] \\
\mathit{CogContext} : & \left[\begin{array}{ll}
\mathit{own\_proc\_state} : & \left[\begin{array}{l}
\mathit{proc\_problem} : \mathit{perc} | \mathit{int} | \mathit{eval} | \mathit{exec} | \mathit{none} \\
\mathit{user\_model} : \langle \mathit{beliefs} \rangle
\end{array}\right] \\
\mathit{partner\_proc\_state} : & \left[\begin{array}{l}
\mathit{proc\_problem} : \mathit{perc} | \mathit{int} | \mathit{eval} | \mathit{exec} | \mathit{none} \\
\mathit{user\_model} : \langle \mathit{beliefs} \rangle
\end{array}\right] \\
\mathit{belief\_model} : \langle \mathit{beliefs} \rangle & \\
\mathit{common\_ground} : \langle \mathit{mutual\_beliefs} \rangle &
\end{array}\right] \\
\mathit{SocContext} : & \left[\begin{array}{l}
\mathit{comm\_pressure} : \mathit{none} | \mathit{grt} | \mathit{apo} | \mathit{thk} | \mathit{valed}
\end{array}\right]
\end{array}\right]$$

Figure 1: Feature structure representation of the context model used.

management act, and if so, for which one (e.g., *comm_pressure: grt* indicates a pressure for the system to respond to a greeting).

3.1 Context update model

The model for updating the context makes explicit how every dialogue act contributes to changing the information state, it defines the types of effects that an utterance provokes in dialogue participants, and it establishes the operations that cause the change of state in the context.

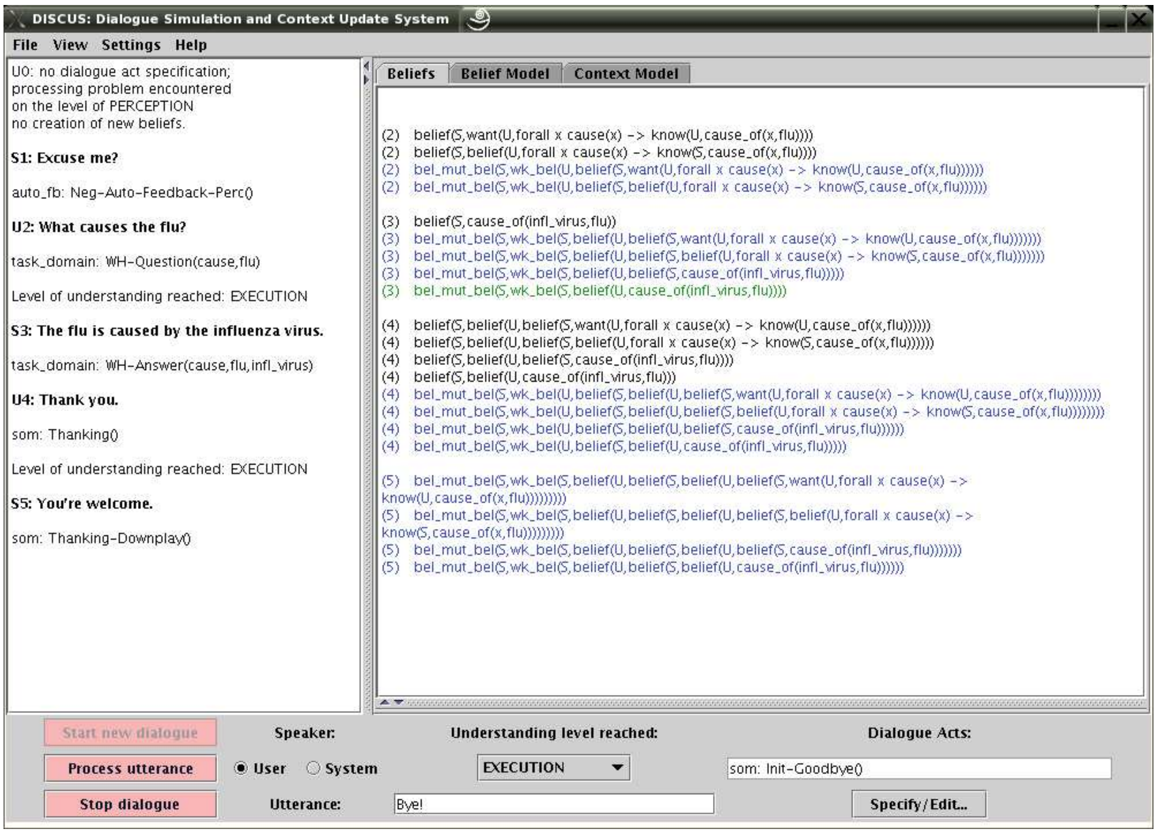
The aspect of the context model related to information transfer and grounding is represented in terms of beliefs. Basically, the types of beliefs we distinguish are represented by means of several operators (weak belief, strong belief, mutual belief, knows value of, wants), that allow to represent the meaning of dialogue acts. As a dialogue evolves, new beliefs are created and existing beliefs may change or be cancelled. Those changes are modelled by means of the operations of *creation, strengthening, cancellation* and *adoption*. Dialogue acts have different types of effects on dialogue participants: effects of understanding and adopting information in the addressee, and effects of expectations of understanding and adoption, and strengthening in speaker and addressee. For more details about this aspect of the context update model, see (Morante et al., 2007).

4 Simulating a QA dialogue

In this section we will discuss the simulation of a dialogue in which the user (U) asks a question that gets answered by the system (S):

U0:	*what causes the flu?*
S1:	*excuse me?*
U2:	*what causes the flu?*
S3:	*the flu is caused by the influenza virus*
U4:	*thank you*
S5:	*you're welcome*

With DISCUS, we can simulate dialogues from the perspective of one of the participants. In this case, we simulate the above dialogue from the system's perspective. In Figure 2, a screenshot of the interface is given. The components in the bottom part of the GUI can be used to specify who is the speaker of the utterance simulated, the system's understanding level reached (in case of a user utterance), a literal text representation of the utterance, and the communicative function (CF) and semantic content (SC) of the dialogue act performed in the utterance. A dialogue act is specified in terms of its *Communicative Function* and its *Semantic Content*, the last of which could be very complex. Here it is specified in terms of at most three arguments. It is also possible to simulate processing problems encountered by the system by identifying a specific level of understanding reached.



```
U0: no dialogue act specification;
processing problem encountered
on the level of PERCEPTION
no creation of new beliefs.

S1: Excuse me?

auto_fb: Neg-Auto-Feedback-Perc()

U2: What causes the flu?

task_domain: WH-Question(cause,flu)

Level of understanding reached: EXECUTION

S3: The flu is caused by the influenza virus.

task_domain: WH-Answer(cause,flu,infl_virus)

U4: Thank you.

som: Thanking()

Level of understanding reached: EXECUTION

S5: You're welcome.

som: Thanking-Downplay()
```

```
(2)  belief(S,want(U,forall x cause(x) -> know(U,cause_of(x,flu))))
(2)  belief(S,belief(U,forall x cause(x) -> know(S,cause_of(x,flu))))
(2)  bel_mut_bel(S,wk_bel(U,belief(S,want(U,forall x cause(x) -> know(U,cause_of(x,flu))))))
(2)  bel_mut_bel(S,wk_bel(U,belief(S,belief(U,forall x cause(x) -> know(S,cause_of(x,flu))))))

(3)  belief(S,cause_of(infl_virus,flu))
(3)  bel_mut_bel(S,wk_bel(S,belief(U,belief(S,want(U,forall x cause(x) -> know(U,cause_of(x,flu)))))))
(3)  bel_mut_bel(S,wk_bel(S,belief(U,belief(S,belief(U,forall x cause(x) -> know(S,cause_of(x,flu)))))))
(3)  bel_mut_bel(S,wk_bel(S,belief(U,belief(S,cause_of(infl_virus,flu)))))
(3)  bel_mut_bel(S,wk_bel(S,belief(U,cause_of(infl_virus,flu))))

(4)  belief(S,belief(U,belief(S,want(U,forall x cause(x) -> know(U,cause_of(x,flu))))))
(4)  belief(S,belief(U,belief(S,belief(U,forall x cause(x) -> know(S,cause_of(x,flu))))))
(4)  belief(S,belief(U,belief(S,cause_of(infl_virus,flu))))
(4)  belief(S,belief(U,cause_of(infl_virus,flu)))
(4)  bel_mut_bel(S,wk_bel(U,belief(S,belief(U,belief(S,want(U,forall x cause(x) -> know(U,cause_of(x,flu))))))))
(4)  bel_mut_bel(S,wk_bel(U,belief(S,belief(U,belief(S,belief(U,forall x cause(x) -> know(S,cause_of(x,flu))))))))
(4)  bel_mut_bel(S,wk_bel(U,belief(S,belief(U,belief(S,cause_of(infl_virus,flu))))))
(4)  bel_mut_bel(S,wk_bel(U,belief(S,belief(U,cause_of(infl_virus,flu)))))

(5)  bel_mut_bel(S,wk_bel(S,belief(U,belief(S,belief(U,belief(S,want(U,forall x cause(x) ->
know(U,cause_of(x,flu)))))))))
(5)  bel_mut_bel(S,wk_bel(S,belief(U,belief(S,belief(U,belief(S,belief(U,forall x cause(x) ->
know(S,cause_of(x,flu)))))))))
(5)  bel_mut_bel(S,wk_bel(S,belief(U,belief(S,belief(U,belief(S,cause_of(infl_virus,flu)))))))
(5)  bel_mut_bel(S,wk_bel(S,belief(U,belief(S,belief(U,cause_of(infl_virus,flu))))))
```

Speaker: (● User) ○ System
Understanding level reached: EXECUTION
Dialogue Acts: som: Init-Goodbye()
Utterance: Bye!

Figure 2: Screenshot of the DISCUS interface.

The text panel in the top left part of the GUI displays the simulated dialogue (Dialogue History). For each utterance in the dialogue, the literal utterance, the speaker (S or U), the dialogue act information along the dimensions of the taxonomy, and in case of user utterances, the processing level reached by the system, can be indicated.

The text panel in the top right part of the GUI displays the beliefs and goals in the context model (Information State). The various kinds of beliefs are displayed in different colours; cancelled beliefs get a 'strike-through' font. Not visible in the screenshot is a separate tab for displaying the full context model.

In processing the first utterance, the system encounters speech recognition problems. This is simulated by specifying a processing level of value 'none' in the interface. Hence, there will be no dialogue acts to specify for this utterance. Updating the context with U1 will create a processing problem on the level of perception being recorded in the own processing state of the cognitive context.

Next, the system signals this processing problem to the user in S1 by means of a negative auto-feedback dialogue act on the level of perception.

In U2, the user repeats his question, and now the system is able to perceive the user's utterance and interpret it as a question about the domain. So, we simulate that the system has reached interpretation level understanding. The interpretation result consists of a dialogue act in the task-domain dimension with a communicative function WH-QUESTION. The semantic content of WH-Questions is specified by means of two arguments, indicating the domain and specific property of the elements from that domain that are asked for by the speaker. In the case of U2, these arguments are *cause* and *flu* respectively.

Updating the system's context model with a dialogue act results in a number of beliefs, based on the preconditions of the dialogue act. As indicated in Section 3.1, these beliefs are the result of different types of effects.

The beliefs in 1 and 2 form the effects of understanding the WH-QUESTION in U2.

$$belief(S, want(U, \forall x\ cause(x) \rightarrow know(U, cause\_of(x, flu)))) \quad (1)$$
$$belief(S, belief(U, \forall x\ cause(x) \rightarrow know(S, cause\_of(x, flu)))) \quad (2)$$

Belief 1 is about a new user goal, i.e., the user wants to know something; belief 2 is about the user believing that the system has the information the user wants.

The user goal is recorded in the *Semantic Context*, because it is related to the underlying task/domain. This is based on the fact that the belief stems from one of the two preconditions associated with the user's WH-QUESTION, a dialogue act in the task-domain dimension.

The beliefs in 3 and 4 form the effects of expected understanding.

$$mut\_bel(S, wk\_bel(U, belief(S, want(U, \forall x\ cause(x) \rightarrow know(U, cause\_of(x, flu)))))) \quad (3)$$
$$mut\_bel(S, wk\_bel(U, belief(S, belief(U, \forall x\ cause(x) \rightarrow know(S, cause\_of(x, flu)))))) \quad (4)$$

These are mutual beliefs that are recorded in the cognitive context as part of the common ground.

Next, we can simulate the system's response, assuming it is cooperative and has been able to find the information requested. This information is represented in the semantic context as the following belief:

$$belief(S, cause\_of(infl\_virus, flu)) \quad (5)$$

This belief, together with 1 and 2 form the preconditions for S3, which is specified as a WH-ANSWER with a semantic content represented by three arguments, two of which correspond to the user's information need created by the WH-QUESTION. The third argument represents the information the system thinks the user asked for, here, *infl(uenza)_virus*.

$$wk\_bel(S, belief(U, belief(S, want(U, \forall x\ cause(x) \rightarrow know(U, cause\_of(x, flu)))))) \quad (6)$$
$$wk\_bel(S, belief(U, belief(S, belief(U, \forall x\ cause(x) \rightarrow know(S, cause\_of(x, flu)))))) \quad (7)$$

$$wk\_bel(S, belief(U, belief(S, cause\_of(infl\_virus, flu)))) \quad (8)$$
$$wk\_bel(S, belief(U, cause\_of(infl\_virus, flu))) \quad (9)$$

Updating the context model with this system dialogue act results in beliefs about expecting that the user understands the system's reply (6, 7, and 8) and that the user adopts the information given by the system (9). These beliefs are also considered by the system to be mutually believed. Such beliefs about mutual beliefs are placed in the common ground of the cognitive context.

Utterance U4 is represented by a dialogue act in the Social Obligations Management dimension, with the communicative function THANKING. Updating the context with this dialogue act creates a so-called reactive pressure, set in the *Social Context*. The system releases this pressure in utterance S5 by means of a THANKING-DOWNPLAY.

5 Dialogue act generation

A more recent feature of DISCUS is that a dialogue act generator (Keizer and Bunt, 2006) can be connected to the simulator and can take care of generating the system's actions. In this case, system utterances no longer need to be simulated through the GUI. The dialogue act generator also follows the multidimensional organisation of the taxonomy in that it consists of several agents, each dedicated to the generation of dialogue acts from a particular dimension. As illustrated in Figure 3, the dialogue act agents monitor and write to different parts of the context model. Dialogue act candidates produced by these agents are recorded in the so-called 'dialogue future' as part of the *Linguistic Context*. The additional Evaluation agent selects a combination of dialogue acts from the dialogue future to form the next system utterance.

Below is a dialogue in which the user utterances are simulated and the system utterances are

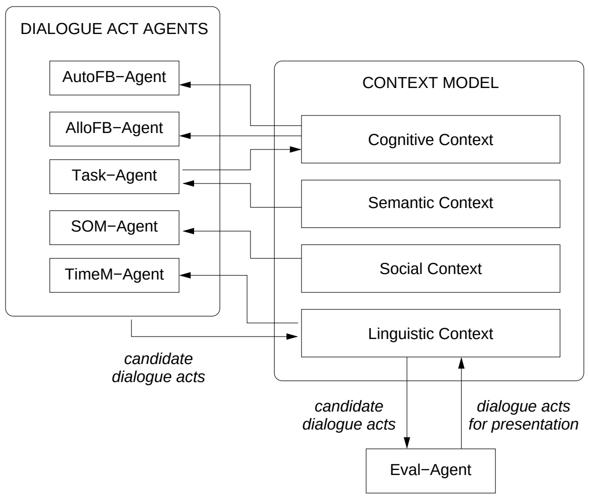


Figure 3: Dialogue act generation architecture.

automatically generated. For this simulation, the task/domain information is contained in a structured database the Task Agent has at its disposal (instead of using separate QA modules, requiring selfcontained, natural language questions as input).

U0: *what are the causes of RSI?*
    WHQ(cause,rsi)

S1: *repetitive movements.*
    WHA(cause,rsi,rep_movs)

U2: *and what are the causes?*
    WHQ(cause,rsi)

S3: *but I just told you!*
    NEG-AUTO-FB-EVAL

U4: *oh sorry, I meant symptoms*
    APO; allo-fb:INF

S5: *so you want to know the symptoms of RSI*
    auto-fb:CHECK

U6: *yes*
    auto-fb:CONF

S7: *just a moment please*
    PAUSE

U8: *ok*
    POS-AUTO-FB-EXE

S9: *unfortunately, I could not find any information*
    APO; NEG-AUTO-FB-EXE

U10: *okay, thanks anyway*
    POS-AUTO-FB-EXE; APO-DP; THK

S11: *you're welcome*
    THK-DP

In U0, the user asks a question about the domain, i.e., the system interprets the user utterance as a WH-QUESTION in the task-domain dimension. Updating the context with this dialogue act results in a number of beliefs (see Section 4), one of which involves a new user goal, i.e., the user wants to know something. This new user goal, that is recorded in the *Semantic Context*, triggers the Task Agent that will try to satisfy it by finding the information requested by the user in the database with domain information.

In this case, the information is found and the Task Agent can construct a WH-ANSWER with that information as the semantic content. This dialogue act is recorded in the dialogue future of the *Linguistic Context*, which results in generating S1.

Next, the user asks another question about the domain, but mistakenly says "causes" instead of the intended "symptoms". The system interprets U2 again as a WH-QUESTION with the same semantic content as in U0. After updating his context with this dialogue act, the system detects an inconsistency: the user cannot have the goal of wanting to know the causes of RSI and at the same time believe that the causes of RSI are 'repetitive movements', as was established in the previous utterances. Therefore, an evaluation level processing problem is recorded in the *Cognitive Context*, causing the Auto-feedback Agent to be triggered and generate a negative auto-feedback act on the level of evaluation. For S3, only this negative feedback act is selected for generation by the Evaluation Agent; any answer to the user's question that might have been generated by the Task Agent is ignored.

In U4, the user realises he made a mistake, apologises and makes a correction to his earlier question. The system interprets this as a dialogue act in the allo-feedback dimension with a communicative function INFORM. In updating his context, the system corrects the effects from his earlier interpretation, including the replacement of the user goal regarding the causes of RSI with a new user goal regarding the symptoms of RSI. Now, the Auto-feedback Agent is triggered to make sure that the system understood the user's question correctly and constructs a CHECK, leading to S5. At the same time, the Task Agent may have been triggered to find the answer to the (corrected) question and already have generated that answer as a candidate.

After the user's confirmation in U6, the system can proceed to produce an answer. In this case however, the Task Agent was not able to return any results within reasonable time. This 'time-out event' is recored in the *Linguistic Context* and triggers the Time-management Agent to generate a PAUSE dialogue act, leading to S7. While the user responds with an overall positive feedback in U8, the Task Agent has finished his attempt to retrieve the required information, but was not successful, and therefore recorded an execution level processing problem in the *Cognitive Context*. The Auto-feedback Agent gets triggered by this new information and generates a negative auto-feedback act on the level of execution. The occurrence of processing problems also triggers the SOM Agent to generate an apology. These two dialogue acts are combined to generate system utterance S9.

Finally, in U10, the user gives overall positive feedback, downplays the apology and thanks the system, thereby pre-closing the dialogue. In updating the context, the THANKING dialogue act causes a reactive pressure of type 'thanking' to be recorded in the *Social Context*. This pressure triggers the SOM Agent, which then constructs a THANKING-DOWNPLAY dialogue act, leading to system utterance S11.

6 Conclusions and Future work

We have presented the DISCUS system as a convenient simulation environment for developing a theory of dialogue management. Building on the DIT framework, we have developed a context model and update algorithm that has been integrated in an interactive question answering system. The rich context model and system of dialogue acts allows for the generation of dialogue behaviour involving different kinds of feedback, interaction management, and social obligations. The multi-agent design of the dialogue manager allows for the generation of multifunctional system utterances.

In general, future work will consist of further improving the tool and extending the implementation of the theory. Particular focus will be on the process of selecting and combining candidate dialogue acts from different dimensions. We will also experiment with different task models via different application agents that the Task Agent can turn to, for example, a database agent or a QA engine.

Acknowledgements

This research is partly funded through the PARADIME project, which falls under the IMIX research programme, funded by the Dutch national research foundation NWO. Roser Morante's research is funded by the EX2005–1145 grant awarded by the Spanish *Ministerio de Educación y Ciencia*. We thank the anonymous reviewers for their useful comments.

References

J. F. Allen and C. R. Perrault. 1980. Analyzing intention in dialogues. *Artificial Intelligence*, 15(3):143–178.

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

H. Bunt. 2006. Dimensions in dialogue act annotation. In *Proceedings 5th International Conference on Language Resources and Evaluation (LREC 2006)*, pages 1444–1449, Genova, Italy.

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

R. Morante, S. Keizer, and H. Bunt. 2007. A dialogue act based model for context updating. In *Proceedings of the 2007 Workshop on the Semantics and Pragmatics of Dialogue (DECALOG)*, Rovereto (Italy), May. To appear.

D. R. Traum and S. Larsson. 2003. The information state approach to dialogue management. In Jan van Kuppevelt and Ronnie Smith, editors, *Current and New Directions in Discourse and Dialogue*, pages 325–354. Kluwer, Dordrecht.