IJCLA VOL. 2, NO. 1–2, JAN-DEC 2011, PP. 155–173
RECEIVED 01/11/10 ACCEPTED 02/12/10 FINAL 12/02/11

Evaluation of a Unified Dialogue Model for Human-Computer Interaction

HUI SHI,[1,2] CUI JIAN,[1] AND CARSTEN RACHUY[1]

[1] *Universität Bremen, Germany*
[2] *DFKI Bremen, Germany*

ABSTRACT

This paper reports our work on evaluating the task success of a dialogue model developed by a unified dialogue modeling approach for human-computer interaction, which combines an information state based dialogue theory and a state-transition based modeling approach at the illocutionary level. As an application, the unified dialogue model has been integrated into a multimodal interactive guidance system for hospital visitors. An experiment with 12 subjects has been carried out. Using the collected dialogue data we have evaluated the task success of the dialogue model by the Kappa coefficient. The results show that the unified dialogue model is highly effective and provide several valuable improvements for the further development as well.

KEYWORDS: *human-computer dialogue, dialogue act, illocutionary structure, information state, dialogue system evaluation, formal methods*

1 INTRODUCTION

Generalized Dialogue Modeling (cf. [14, 8, 12]) and *Information State* based dialogue theories (cf. [15, 5, 2, 4, 7, 16]) are the two most important approaches to develop dialogue models. Generalized dialogue models are based on recursive transition networks. These models consist of pattern-based accounts of dialogue structure at the illocutionary level and

therefore, are independent of utterance content or other direct surface indicators. Information state theories, on the other hand, offer a powerful basis for interaction analysis and practical dialogue system construction. However, such information state based dialogue models are difficult to manage, to extend and to reuse. Although it has been suggested that applying generalized dialogue models to information state based accounts could eliminate some of the perceived problems, there have only been preliminary researches to date [18, 8]. In Lewin [8], for example, recursive transition networks were applied to model Conversational Game Theory by combining dialogue grammars with discourse planning.

The unified dialogue modeling approach introduced in this paper combines the information state based dialogue theory discussed in [16, 7] and the generalized dialogue modeling approach proposed in [14, 12]. Specifically, unified dialogue models extend generalized dialogue models by introducing *context-sensitive* transitions, which allow for direct integration with information state management. A unified dialogue model is represented as the traversal of a state-transition network with arcs denoting context-sensitive transitions and nodes denoting dialogue states. In addition to the allowed dialogue action, each context-sensitive transition is associated with a set of *conditions* under which the dialogue action can be taken and a set of *update rules* for updating the information state after performing the dialogue action.

As emphasized in [11, 12], the separation of illocutionary structures from the information state-based modeling enables the formal analysis and comparison of illocutionary structures by applying well-established techniques from the formal methods community of computer science. In this paper, we focus on the evaluation of unified dialogue models. The *Kappa coefficient* [13, 3] has been proposed as a standard measure of reliability and task success ([17]) for evaluating spoken dialogue systems. Therefore, we apply it to evaluate how well human users can be supported by the unified dialogue model implemented in a multimodal dialogue system for guiding visitors in hospital environments. For this purpose we carried out an experiment with 12 people and collected 272 dialogues.

This paper is organized as follows: Section 2 introduces the unified dialogue modeling approach, which has been applied to develop a unified dialogue model for a practical multimodal dialogue system presented in Section 3. Section 4 describes the experiment and the collected dialogue data, which are then used to evaluate the unified dialogue model by the Kappa coefficient in Sections 5 with respect to the measure of task success. The evaluation results and corresponding improvements are

discussed in Section 6. Finally, Section 7 concludes with the outline of future work.

2 A UNIFIED APPROACH FOR DIALOGUE MODELING

The unified modeling approach takes as a starting point existing researches on the generalized dialogue modeling at the illocutionary level using Recursive Transition Networks (RTNs) [14]. Unlike finite state models, the RTNs employed here capture more abstract dialogue models which depict discourse patterns in illocutionary force terms only – without reference to propositional content or other direct surface indicators. Fig. 1(a) depicts a transition diagram named *Assert(A,B)* initiated by a dialogue participant, say A, and responded to by B. The darkened circles denote final states. This generalized transition diagram is initiated by A's dialogue move of type *assert*. The possible responses from B are threefold: B agrees with the assertion (*B.agree*), accepts it (*B.accept*) or rejects it (*B.reject*). To note that, the transition diagrams *Ask(B,A)* and *Assert(B,A)* are used to enable B to ask some question(s) before reacting to A's request, or to give possible reason(s) by a rejection, and are not presented here in detail.

Generalized dialogue models such as the one depicted in Figure 1(a) are non-deterministic models, where more than one dialogue move is able to trigger state transitions starting from one state. The decision as to which transition should be activated naturally depends to a certain extend on B's pragmatic domain knowledge. To take domain knowledge into account, thus to solve such nondeterministic transitions, *conditional transitions* are introduced in unified dialogue models. A conditional transition can be activated only if its conditions are satisfied. Let *checkAssert* be an operation provided by B's domain component, which takes an assertion as a parameter and returns *true* if B's knowledge matches the assertion; or *false* if the assertion conflicts with some of B's knowledge (in that case, the transition diagram *Assert(B,A)* will be activated to explain the reason for B's rejection); or *added*, if the assertion can be added by B as a new element to the knowledge base. A deterministic transition diagram for the example is now shown in Figure 1(b), where a is assumed to be the assertion made by A.

Although conditional transition models as shown in Fig. 1(b) capture the illocutionary structure of dialogues and are deterministic as well, they do not provide mechanisms to integrate dialogue context and history. Therefore, they do not reflect dialogue participants' attitudinal state

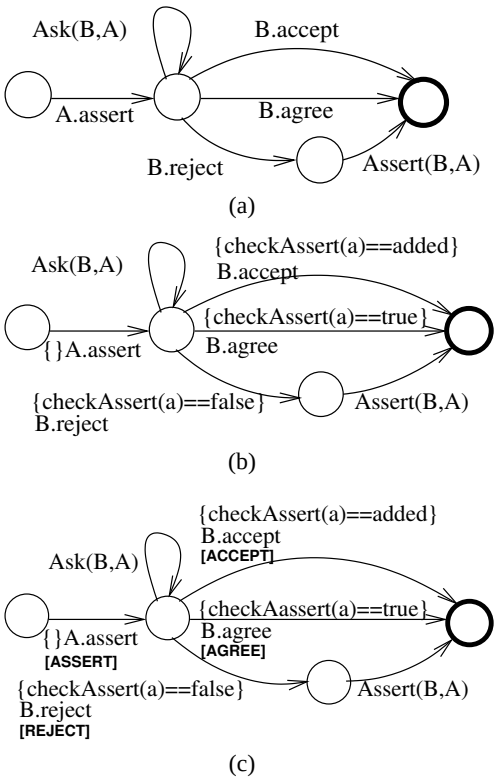

Fig. 1. Three transition diagrams: (a) non-deterministic assertion, (b) deterministic assertion, and (c) deterministic assertion with update rules

along with the behavioral mechanisms for dialogue progression and the dynamic update of attitudinal states over time. As indicated earlier, information state based approaches of dialogue models [9, 15, 4] and dialogue management [16, 7] focus on the modeling of dialogue contexts and participants' attitudinal states, apart from that they do not capture the structural features of dialogues. Thus, merging these two approaches is valuable, so that the basic formalism of the conditional transition models is extended by introducing a mechanism to interface with information state.

Since generalized dialogue models already capture structural features of dialogue moves, some of the typical *structural elements* in the infor-

mation state based accounts, e.g., *AGENDA* for keeping the planed dialogue acts in Ginzburg and Larsson's models, become unnecessary, hence the information model can be simplified considerably. In unified dialogue models, each transition can be associated with one or more update rules for updating the current information state if needed before proceeding to the next state. As usual, an update rule consists of a name, a set of preconditions and a set of operations on information states. To illustrate this model extension, we again take the transition diagram *Assert(A,B)* as an example and show it in Fig. 1(c). After dialogue participant A makes an assertion, the update rule *ASSERT* will be applied to update the information state, such that the new assertion can be integrated into the current information state. Similarly, B's transitions of *accept*, *agree* and *reject* can change the information state by the corresponding update rules.

Finally, a *unified dialogue model* is a pair $\langle \mathcal{G}, G_0 \rangle$ of a transition network $\mathcal{G}$ with a set of extended recursive transition diagrams and a main diagram $G_0 \in \mathcal{G}$. Each transition may contain some conditions and information state update rule(s). Specifically, if a dialogue is in the start state of a transition whose conditions are satisfied, the corresponding dialogue move is then enabled and the information state is updated by its update rules, and the dialogue will move to its goal state.

3 MIGHE: A MULTIMODEL INTERACTIVE GUIDANCE FOR HOSPITAL ENVIRONMENT

MIGHE is a multimodal interaction system developed for guiding people in public areas such as hospitals. Fig. 2 shows the overall MIGHE architecture. This section focuses on the development of a unified dialogue model and its integration into the dialogue system. The unified dialogue model is implemented within the two components: the *dialogue controller* and the *information state manager*. The *clinic database manager* provides the dialogue controller with necessary information about application environment. The dialogue controller manages the communication between various system components, and controls the dialogue process according to the dialogue model together with the information state manager. The guidance system supports both natural language inputs and touch events, but in the experiment presented in Section 4 only the natural language input channel is enabled.

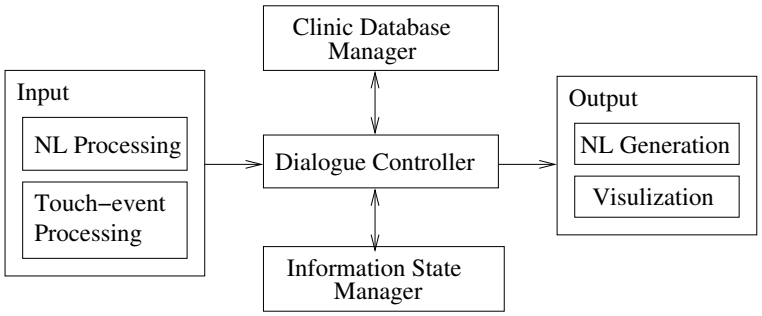

Fig. 2. The overall architecture of the dialogue system

3.1 *The Unified Dialogue Model*

The dialogue model implemented in MIGHE is developed according to the unified dialogue modeling approach introduced in Section 2. In this paper we focus on the task orientated dialogues and disregard communication problems like failures by speech recognition or misunderstanding. Generally, these problems can be treated by extending the dialogue model. The information state structure consists of two parts: *LM* for keeping the latest dialogue move and *CONTEXT* containing a list of contexts of active (sub-)dialogues. In this application, the possible contexts are of the types: *department*, *person*, or *room*, which provide context information for integrating user's dialogue moves, for example, "go to *a room* of *a known department*", or "request for information of *a person* in *a department*".

The unified dialogue model consists of four extended transition diagrams with the main diagram *Dialogue(S,U)*, see Fig. 3. After a system's initializing *request* (Fig. 3(a)), the user can instruct the system to find some visiting goals by utterances with the dialogue act *instruct*, or ask the system to find certain information by *request* (see *Dialogue(U,S)* in Fig. 3(b)). The network *Response(S,U)* (Fig. 3(c)) specifies all deterministic system responses after getting an input from the user according to its domain knowledge and the current information state. If the requested information or instructed goal does not exist, the user's input is *rejected*, probably with a reason if the relevant information is available. If it is found unambiguously, the user is *informed* and asked whether he/she would like to take the found place as a destination in case the last user input is an instruction. However, if more than one possibility are found, a subdialogue

is started by the system for asking the user to make a *choice*. Finally, *Response(U,S)* (Fig. 3(d)) describes possible *nondeterministic* user reactions to a system's request. Moreover, each dialogue move issued by a user in the dialogue model is associated with the name of an update rule.

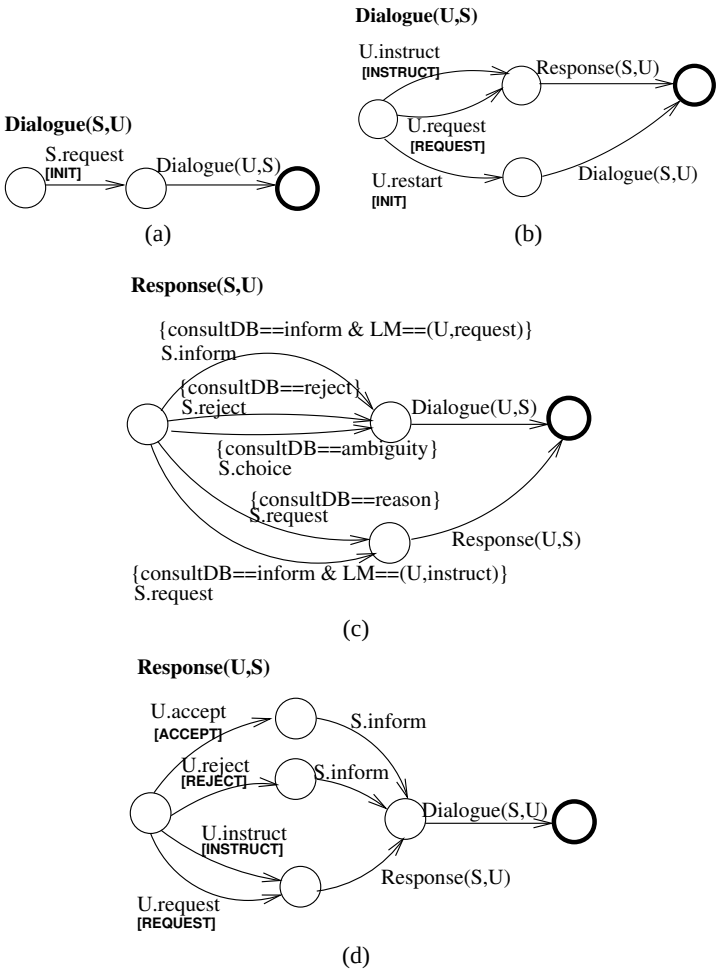

Fig. 3. The unified dialogue model: (a) the main transition diagram, (b) the transitions issued by the user, (c) the system's responses and (d) the user's response

3.2 *Integrating the Unified Dialogue Model into the Dialogue System*

The implementation of the unified dialogue model is carried out in two major steps. In the first step, a set of update rules as required by the dialogue model is implemented for the component *information state manager*. Five update rules are needed in the unified dialogue model (see Fig. 3). The following shows the rule INSTRUCT as an example. Suppose that *context* and *dest* are two operations to identify the context and destination contained in an input, respectively. The context and destination of "I'd like to go to Mrs. Angelika Fromm in Gastroenterology", for example, are "Gastroenterology" and "Mrs. Angelika Fromm". If the current instruction contains context information, i.e., the user gives the context in his/her instruction explicitly, then the new context will be added to *CONTEXT*, otherwise, the most actual context in *CONTEXT* (or *top*(*CONTEXT*)) is used to complete the current instruction. The other rules are defined accordingly.

RULE: INSTRUCT
PRE: if *context*(m)!=*null* then c = *context*(m)
else c = *top*(*CONTEXT*), d = *dest*(m)
EFF *LM* = (U, *instruct*),
if *context*(m)!=*null* then *CONTEXT* = *add*(*CONTEXT*, c)

The second step is the development of the control mechanism of the component *dialogue controller*, which is based on the dialogue state transitions at the illocutionary level specified by the dialogue model. As the unified dialogue model defines a clear illocutionary structure represented by a set of extended recursive transition diagrams, it can be specified with mathematically well-founded methods straightforwardly, e.g., the well-established technique from the formal methods community of computer science *Communicating Sequential Processes* (CSP). The CSP language provides mechanisms for specifying the communication and synchronization of two or more processes consisting of sequential actions. The essential value of CSP is the ability to subject formal specifications that are well founded in mathematical logic to enable powerful analysis using mechanized theorem provers and model checkers (cf. [12]). Although the CSP language, its mathematical foundations and its many possible applications within the Formal Methods Community have been widely investigated [6, 10], applying these techniques to dialogue modeling, specification and analysis builds up a novel area of application. In the following we will briefly introduce the specification of the unified dialogue model presented in Section 3.1 using CSP.

The first CSP process *DialogueUS* in Fig. 4 specifies the transition network *Dialogue(U,S)*, where $\rightarrow$ and [ ] are two CSP operators necessary for the present specification. $\rightarrow$ defines the sequential occurrence of dialogue moves in a process, and [ ] arbitrary selection between several possibilities. The CSP events representing abstract dialogue moves have the form $p.a$, where p is the name of a communication channel and a the dialogue act associated with it. For example, *user.instruct* means getting an input with the dialogue act *instruct* from the user, *is_out.instruct* sending the dialogue act *instruct* to the information state manager, such that the information state can be updated using the context contained in the current input. Obviously, the specification reflects the model structure very well. The second CSP process *ResponseSU* invoked by the first one in Fig. 4 specifies the transition network *Response(S,U)*, in which the *latest dialogue move* kept in the information state is needed. In the specification *ResponseSU* the conditions related to *consultDB* are specified by four database input *db_in* events: *reject*, *reason*, *inform* and *ambiguity*. Also the CSP specification of *Response(U,S)* reflects the network structure straightforwardly.

```
DialogueUS =
    user.restart -> is_out.init -> DialogueSU
 [] user.instruct -> is_out.instruct -> ResponseSU
 [] user.request -> is_out.request -> ResponseSU

ResponseSU = db_out -> (
    db_in.reject -> system.reject -> DialogueUS
 [] db_in.reason -> system.request -> ResponseUS
 [] db_in.inform -> is_in?lm ->
   ( (lm==request) & (system.inform -> DialogueUS)
   [](lm==instruct) & (system.request -> ResponseUS))
 [] db_in.ambiguity -> system.choice -> DialogueUS)
```

Fig. 4. Two CSP specifications

Based on the CSP specifications the model-checker FDR [1] is applied to generate the state machine. After implementing the communication channels between the dialogue controller and the other system components, the state machine can control the state transitions according to communication events.

4 THE EXPERIMENT

In order to explore how well the dialogue interaction between human and the dialogue system is assisted by the unified dialogue model, an evaluation with 12 participants was carried out. Each subject had to undergo two test phases: learning and testing:

- In the learning phase each participant was given a brief introduction to the test procedure, so that they could get to know the way how to dialogue with the system, and what kinds of verbal and textual feedbacks the system provides. Furthermore, they were asked to accomplish several sample tasks.
- In the test phase each participant had to go through three subphases, each of which contains several tasks belonging to a predefined category. In the first subphase, several pieces of information describing a destination (e.g. a person's name, a department or a room number) were given and the participant should tell the system to go there. In the second subphase, pieces of information were given as well, but this time the participant was asked to find out certain information, e.g. where a certain person works or what department a room is in. In the third subphase scenarios like "you are hungry and would like to eat something" were described, and the participant was asked to negotiate with the system on an appropriate destination.

The dialogue system used in the experiment was a networked software application that connected two computers: the *guidance assistant* on one computer and the *input system* on the other. The *input system* was controlled by a human operator who entered the user utterances and acted as a speech recognizer. The *guidance assistant* contains the components *clinic database manager*, *output*, *information state manager* and *dialogue controller*, and the unified dialogue model is the key of the *information state manager* and *dialogue controller*. As a result, the whole test run was simulated as if the participant communicates with the system in natural language directly, but removing possible distractions that might have been introduced by speech recognition, in order to focus on the evaluation of the unified dialogue model. Although a human operator acted as the speech recognizer, our experiment was not a usual "Wizard of Oz" experiment, since the *guidance assistant* ran automatically.

Since the experiment was run with native German-speaking participants, we present in the following the English translations of several example dialogues collected in the experiment. Most of the dialogues turned

out to be unproblematic, as the following example shows. The task of the example contained the destination “Rasmussen” and “Room number 1322”. The room number in the user’s *instruction* (*D_01_01*) was incorrect, thus the system *rejected* it in *D_01_02* with a reason. In *D_01_03* the user provided the room number again, which is interpreted as a new *instruction* integrated with the context *person* provided in the previous instruction and kept in *CONTEXT* of the information state. This time the system found the person and his office, and then responded with a *request* such that the user can confirm or reject it.

D_01_01 U I’d like to go to Mrs. or Mr. Rasmussen in room number 1320.
D_01_02 S I did find Rasmussen, but not in room 1320.
D_01_03 U I’d like to go to room 1322.
D_01_04 S Would you like to go to Mr. Per Rasmussen in room 1322, the office of chief physician?
D_01_05 U Yes.

However, there are dialogues turned out to be problematic. The utterances from *D_02_01* to *D_02_04* fit well to the model, but the utterance *D_02_05* is an user *request*, for which no proper system response is defined. Therefore, the system alternatively offered the *information* about all persons (*D_02_06*). Again the user *requested* the system to name a person with the given name in *D_02_07*, the system found a person with the name, and *requested* the user to confirm, since it is the only person with the last name “Brückner”. The system utterance *U_04_08* did not match the user’s request in *D_02_07*, showing an unnatural dialogue segment.

D_02_01 U I’d like to go to Brückner.
D_02_02 S Would you like to drive to the doctor Mr. Wolfgang Brückner?
D_02_03 U I need more information about him.
D_02_04 S Mr. Wolfgang Brückner is a doctor in the accident surgery. You find him in room 1711.
D_02_05 U Is there an other person with the name Brückner?
D_02_06 S I found all together 32 persons and they are sorted according to their last names.
D_02_07 U Name a person with the name Brückner.
D_02_08 S Would you like to drive to the doctor Mr. Wolfgang Brückner?
D_02_09 U If there is no other person with the name Brückner, then yes.

5 DIALOGUE MODEL EVALUATION CONCERNING TASK SUCCESS

One aim of our experiment was to investigate the task success of the unified dialogue model in a practical dialogue system. Specifically, in this section we are going to evaluate how the system's deterministic behavior (see Fig. 3(c)) influences the task success. Success at the task of a dialogue in our context is measured by how well the dialogue model supports users to complete dialogue tasks and therefore, we apply the Kappa coefficient [13, 3, 17] approach, similar applications can be found in the literature, such as the evaluation of two train timetable information agents in [17].

First, we define a set of *attribute values* for each task. As shown in Fig. 3(d) the unified dialogue model allows a user to make a dialogue move with an *instruction* like "take me to $\cdots$", a *request* like "tell me about $\cdots$", an *accept* like "yes" or a *reject* like "no" after a system's utterance. Each user's dialogue move may contain some content information, also called *attribute values*, of a person's name, a room number and so on. Tab. 1 summarized the set of all relevant attributes.

Table 1. The set of attributes

attribute name	identifier	description	example
first name	FN	first name of a person	Wolfgang
last name	LN	last name of a person	Brückner
gender	G	gender of a person	M
profession	P	profession of a person	Doctor
room number	RNr	number of a room	1711
room type	RT	type of a room	station room
meta room type	MRT	predefined meta type of a room	eating-related
station	F	name of a hospital station	accident surgery

Since different tasks contain different data and have different goals, each task has a set of expected dialogue acts and attribute values, such as the *attribute value matrix* (AVM) in Tab. 2 for the task, in which the participants were asked to go to a person with the last name "Brückner" (see the example dialogue *D_02* in Section 4). Each expected dialogue act-attribute pair is associated with an actual value, which reflects the fact that a unified dialogue model contains a state transition based structure at the illocutionary level and an information state management processes.

With the attribute value matrix we can develop the confusion matrix for the collected dialogue data of that task (see Tab. 3).

Table 2. An example of value matrices for dialogue acts and attribute values

dialogue act	attribute	actual values
instruct	LN	Brückner
	G	M, F
accept	LN	Brückner
	FN	Wolfgang
	P	Doctor
	G	M

Table 3. An example confusion matrix

data		instruct				accept								other	sum
		LN		G		LN		FN		P		G			
		E	NE	E	NE	E	NE	E	NE	E	NE	E	NE		
instruct	LN	12												4	16
	G			9											9
accept	LN					12									12
	FN							11							11
	P									9					9
	G											11			11

The values in the confusion matrix are obtained by comparing the dialogue moves issued by the participants and the expected attribute values of each task specified by a AVM. A user dialogue move may contain expected or unexpected information with respect to the attribute values defined in the AVM for a dialogue task, so we use “E” and “NE” in confusion matrices to denote such situations. Values in the “other” column record the number of undefined dialogue moves occurred in the dialogue data. Hence, these confusion matrices capture not only expected dialogue situations, but also unexpected and undefined situations.

Given a confusion matrix, the success at reaching dialogue goals is measured with the Kappa coefficient [13, 3, 17]: $\kappa = \frac{P(A)-P(E)}{1-P(E)}$, where $P(A)$ is the proportion of times that the dialogue moves agree with the

attribute values and $P(E)$ is the proportion of times that the dialogue moves are expected to be agreed by chance. In our case,

$$P(A) = \frac{\sum_{i=1}^{n} M(i,E)}{T}, \qquad P(E) = \sum_{i=1}^{n} \left(\frac{M(i)}{T} \right)^2$$

where $M(i,E)$ is the value in an expected column of row i, T is the sum of all user dialogue moves, and $M(i)$ the sum of the user dialogue moves in row i.

Since our goal is to find out how well the dialogue model implemented in the dialogue system supports various types of tasks, instead of individual tasks, we first calculate the Kappa coefficient for each type by the confusion matrix combining all the confusion matrices of the tasks in that type. The first type contains 13 tasks with 149 dialogues, the second type 3 tasks with 35 dialogues, the third type 8 tasks with 88 dialogues. Since the third type contains the second type implicitly, only three tasks were taken in the experiment for the second type. Finally, the three confusion matrices of the three individual task types are combined to a single confusion matrix for computing the total Kappa coefficient. The results are presented in Tab. 4.

Table 4. The task type dependent and independent Kappa coefficients

task type	type I	type II	type III	type I, II, III
Kappa coefficient	$\kappa_1 = 0.99$	$\kappa_2 = 0.85$	$\kappa_3 = 0.82$	$\kappa = 0.94$

6 Discussion of Evaluation Results and Improvements

From the Kappa coefficients calculated in table 4, we can see that the unified dialogue model well supports the three types of dialogue tasks: the Kappa coefficients κ_1, κ_2 and κ_3 are greater than 0.8 and the task type independent Kappa coefficient κ is 0.94.

Especially, the achievement of the dialogue model by the first type of tasks is obvious, since the Kappa coefficient almost equals 1. In all the 13 tasks of the first type the participants were asked to instruct the system to a destination described by several pieces of information about a person, a room or a department (see the dialogue example *D_01* in Section 4), and therefore, the dialogue goal of each task was given explicitly. On the

other hand, each dialogue goal in the third type was described implicitly, such as the goal of the task described by "you are hungry and would like to eat something" was "the patient dining-hall". From the fact $\kappa_1 > \kappa_3$ we can conclude that the unified dialogue model supports dialogues with a clearly defined goal much better than those without. Missing the negotiation of users' intention in the unified dialogue model is the key fact leading to this finding.

The analysis of the confusion matrices issues two drawbacks of the unified dialogue model: losing track of the current user dialogue move type and missing some unexpected dialogue move types. For example, the dialogue *D_03* itself was straightforward. The problem was that the system could not continue the dialogue after the utterance *D_03_03*, since it lost the information to decide whether the utterance is a *request* or an *instruct* move and therefore, the dialogue model cannot determine the next transition to be taken (see Fig. 3(c)). The problem can be solved by adding the *move context* into the information state, which keeps the current user dialogue act, in our case *instruct* or *request*.

D_03_01 U Which profession does Mr. Schulte practice?
D_03_02 S I have found several persons who match Mr. Schulte.
Which person do you mean?
D_03_03 U I mean Mr. Tobias Schulte.

Extending the illocutionary structure of the unified dialogue model is another step to improve the dialogue model. *D_04*, for example, contains the user dialogue move *D_04_02*, where the user told the system that he did not know the room number. This type of dialogue moves is not supported by the illocutionary structure of the dialogue model. Adding a new transition *U.inform* to the start state in the transition dialogue of *Response(U,S)* (see Fig. 3(d)) enables the dialogue model to handle such dialogue moves.

D_04_01 S Would you like to drive to room 1262, ECG 2, in the cardiology?
D_04_02 UI don't know the room number.

The refinements of the dialogue model by adding new information state elements and additional transitions have been applied to update the dialogue system. We believe that they will improve the task success of the unified dialogue model throughout. This has to be proved by a follow-up experiment.

Based on the evaluation results, we conclude that the unified dialogue model well supports users to dialogue with the hospital guidance system,

however, they cannot be used to measure the effectiveness of the whole dialogue system, since all the test runs were, with the assistance of a human operator[3], simulated as if the participants were conversing with the system in natural language directly, but removing possible distractions that might have been introduced by speech recognition. Comparing the audio data with the manual input data did not deliver any essential deviation that would affect task successes of any undergone dialogues. Therefore, our focus on evaluation of the unified dialogue model is maintained.

Unified dialogue models are constructed at the illocutionary force level, which naturally enables dealing with diversity situations. However, choosing the appropriate set of communicative acts is one important factor affecting the coverage of a unified dialogue model. Care must be taken on the one hand to avoid over-simplification to the point where the structural model collapses down to a two-state initiate-response network with jumps. Although these over-simplified models capture most dialogue situations, they are not useful for dialogue control or formal analysis of dialogue structure. On the other hand, models, as the one discussed in this paper, well reflect natural dialogue structures at the illocutionary level and still possess the context sensitive information state management that relies on domain specific communication. Diversity problems might occur when people dialogue with a system based on a too excessively designed unified dialogue model, but through appropriate design and careful evaluation possible diversities can be detected and the model can then be improved accordingly.

7 CONCLUSION

In this paper we applied the Kappa coefficient (κ) to evaluate the effectiveness of a unified dialogue model by task success, which combines a generalized dialogic structure at the illocutionary level and an information state based content manager. Specifically, three Kappa coefficients were calculated from the confusion matrices for three types of dialogue tasks using the 272 dialogues collected in an experiment with 12 participants. The results showed that the unified dialogue model well supports those dialogue tasks in general ($\kappa = 0.94$). Especially, tasks with an explicit defined dialogue goal (cf. $\kappa_1 = 0.99$). The experiment results also delivered useful findings for the improvement of the dialogue model. This

[3] We used only one operator in the whole experiment

paper has three major contributions. First, it showed the development of unified dialogue models in general and by an example. Second, we demonstrated how to evaluate unified dialogue models by combining dialogue acts with attribute values. Third, we applied the standard method, the Kappa coefficient, to evaluate a unified dialogue model.

To evaluate the improvement of the unified dialogue model according to the analysis of the experiment results, we are now carrying out a follow-up experiment. The collected dialogue data will also be used for training an automatic speech recognizer, which will then be integrated into the multimodal interactive system for further experimenting. Last but not least, applying reinforcement learning techniques to enhance the existing unified dialogue model centered management system is another research direction we are now concerned with.

ACKNOWLEDGMENTS We gratefully acknowledge the support of the Deutsche Forschungsgemeinschaft (DFG) through the Collaborative Research Center SFB/TR 8 Spatial Cognition – Subproject I3-SharC. We would like to thank Prof. Dr. Nicole von Steinbüchel, Frank Schafmeister and Nadine Sasse from the Department of Medical Psychology and Medical Sociology at Georg-August-Universität Göttingen for helping us to plan and execute the experiment.

REFERENCES

1. Failures diference refinement. FDR2 manual. Technical report, Formal System (Europa) Ltd., 2001.
2. J. Allen. *Natural Language Processing*. Benjamin Cumminings Publishing Company Inc., 1995.
3. J. C. Carletta. Assessing the reliability of subjective codings. *Computational Linguistics*, 22(2):249–254, 1996.
4. J. Ginzburg. Dynamics and the semantics of dialogue. *Language, Logic and Computation*, 1, 1996.
5. B. J. Grosz and C. L. Sidner. Attention, Intentions and the Structure of Discourse. *Computational Linguistics*, 12(3):175–204, 1986.
6. C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.
7. S. Larsson. *Issue-Based Dialogue Management*. PhD thesis, Department of Linguistics, Göteborg University, 2002.
8. I. Lewin. A formal model of conversational game theory. In *The Fourth Workshop on the Semantics & Pragmantics of Dialogue*, 2000.
9. D. K. Lewis. Scorekeeping in a language game. *Journal of Philosophical Logic*, 8, 1979.

10. A. W. Roscoe. *The Theory and Practice of Concurrency*. Prentice-Hall, 1998.
11. H. Shi, R. Ross, and J. Bateman. Formalising Control in Robust Spoken Dialogue Systems. In B. K. Aichernig and B. Beckert, editors, *Proceedings of Software Engineering and Formal Methods 2005*, IEEE, pages 332–341. IEEE Computer Society, 2005.
12. H. Shi, R. J. Ross, T. Tenbrink, and J. Bateman. Modelling illocutionary structure: Combining empirical studies with formal model analysis. In A. Gelbukh, editor, *Proceedings of the 11th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2010)*, volume 6008 of *Lecture Notes in Computer Science*, pages 340–353, 2010.
13. S. Siegel and N. J. Castellan. *Nonparametric Statistics for the Behavioral Sciences*. McGraw Hill, 1988.
14. S. Sitter and A. Stein. Modelling the illocutionary aspects of information-seeking dialogues. *Journal of Information Processing and Management*, 28, 1992.
15. R. Stalnaker. Assertion. *Journal of Syntax and Semantics*, 9, 1979.
16. D. Traum and S. Larsson. The information state approach to dialogue management. In *Current and New Directions in Discourse and Dialogue*. Kluwer, 2003.
17. M. A. Walker, D. J. Litman, C. A. Kamm, and A. Abella. PARADISE: A framework for evaluating spoken dialogue agents. In *Proc. of the Eighth Conference on European Chapter of ACL*, pages 271–280, 1997.
18. W. Xu, B. Xu, T. Huang, and X. Hairong. Bridging the gap between dialogue management and dialogue models. In *Proc. of the Third SIGdial Workshop on Discourse and Dialogue*, 2002.

HUI SHI
SFB/TR8 SPATIAL COGNITION,
UNIVERSITÄT BREMEN,
GERMANY
AND
SAFE AND SECURE COGNITIVE SYSTEMS,
DFKI BREMEN,
GERMANY
E-MAIL: <SHI@INFORMATIK.UNI-BREMEN.DE>

CUI JIAN
SFB/TR8 SPATIAL COGNITION,
UNIVERSITÄT BREMEN,
GERMANY
E-MAIL: <KEN@INFORMATIK.UNI-BREMEN.DE>

CARSTEN RACHUY
SFB/TR8 SPATIAL COGNITION,
UNIVERSITÄT BREMEN,
GERMANY
E-MAIL: <RACHUY@INFORMATIK.UNI-BREMEN.DE>