

Speech Dialogue Systems — A Pragmatics-Guided Approach to Rational Interaction

Günther Görz and Bernd Ludwig

Computer Science Institute

University of Erlangen-Nuremberg, Germany

The general goal in implementing dialogue systems is in most cases to provide some kind of human-computer-interface for rational interaction. For that purpose, an approach is required which enables such a system to satisfy user goals in a given (ideally open) domain by conducting spoken dialogues. Furthermore, it should be possible to augment it by other kinds of multi-modal interaction like gestures or the selection of items from a menu on a screen. Interactions are called “rational” because rationality principles — at the knowledge representation level — should be applied to optimally select appropriate communicative actions. In this article, different approaches to the understanding of dialogues and the implementation of dialogue systems will be presented. As a basis for comparing the approaches, fundamental issues of language understanding, speech act theory, and theories of rational interaction are outlined. To give an impression of their implications for the implementation of dialogue systems, we refer to the system framework developed in Erlangen, which has been applied in various projects, as a show case throughout the paper.

1 Rational Dialogues: General Assumptions

We assume that the satisfaction of user goals within the thematic framework of a particular application domain can be achieved with the help of a dialogue system in close cooperation with a technical application which is called the “domain problem solver”. That may be an information or reservation system, or a system for controlling technical devices, to mention a few examples.

In such settings, Allen et al. [1] characterize “practical systems” as task- or goal-oriented dialogue systems, which conduct dialogue on accomplishing a concrete task. They claim that the conversational competence required for practical dialogues is significantly more easy to achieve than general human conversational competence. We are convinced that this distinction is useful because it provides a realistic starting point and because the latter — general human conversational competence — is hard to define. Hence, we begin with a certain well-defined conversational competence try to augment it with increasingly complex requirements from the application. I.e., we attempt to proceed in an incremental fashion. However, we cannot expect that this can be done in the way of a linear progress: Probably completely different requirements come in which cannot be integrated seamlessly.

Another general underlying assumption is that for the interpretation of dialogue we insist on a clear commitment to a (computational) logic framework. Of course, humans act incoherently and even inconsistently, and common sense reasoning can be understood in terms of logic to a certain extent only, but we are convinced that a coherent and consistent rational reconstruction is the best we can do about it. A constructive perspective like this enables us to begin

with a well understood framework for knowledge representation and reasoning upon which we can attempt to build rule systems for still idealized, but more realistic patterns of argumentation in specific domains. We believe that there is a potential to succeed in a variety of prevalently instrumentalized contexts as is the case with technical applications — that will be discussed in more detail below — or in forensic argumentation, to give another example.

2 On Dialogue Modelling

The question of scalability of natural language dialogue systems has been an issue for quite a long time. It becomes more and more pressing with the increasing processing power and memory size of modern hardware and the progress in software engineering techniques.

2.1 The Structural Approach

For several decades, natural language and speech systems for information dialogues have been developed on the basis of an approach which led from experimental systems to a commercially available technology. Nevertheless, we will argue that those systems, although successful in many domains, have severe limitations which are inherent to the underlying so-called structural approach.

The goal addressed by this first system generation is to accomplish an information task. By conducting more or less strictly guided dialogues they aim at supplying a user with a specific information in a given domain, e.g. train connections, traffic jams, cinema programmes, stock exchange data, or weather information in as few dialogue turns as possible. The technical application they are cooperating with is usually a

static database system; in some cases it has been extended with an application server that might accept orders or reservations. The basic technique used by these systems is the extraction of parameter values for a given schema from user utterances (“slot filling”) under the closed world assumption. In general, they have, if any, a very limited and domain-specific inferencing capability. As for dialogue modelling, they rely on a state-based approach, which is also called *structural*. As speech in speech recognition technology, dialogues are modelled by means of stochastic finite-state machines. The designer of a structural dialogue model is forced to anticipate future admissible dialogue states. If large annotated corpora are available, stochastic training techniques can help, but the general problem of anticipation remains. In this framework, the general view of dialogue in speech communication is understood as controlling and restricting interaction — this is fundamentally different from basing human-computer interaction on human conversation (cf. [1]). This “controlling and restricting” view is to a large extent due to robustness requirements all speech dialogue systems — not only the structural ones — are faced with: recognition errors, speaker adaptation, and “out of vocabulary/domain” problems.

So, which limitations are inherent in the structural approach? Despite various improvements like the introduction of anchoring into the dialogue history, using dialogue act predictions and defaults, and employing sub-dialogue patterns, structural approaches fail to deal with semantic and pragmatic problems in a flexible way. References cannot be resolved, defaults cannot be applied, and subdialogues cannot be initiated.

This assertion is corroborated by the results of an evaluation of more than 1100 recorded train information dialogues with the EVAR system, a structural system developed at our university [4,9]: 68.5% of the dialogues were completed successfully. Of the remaining 31.5% incomplete dialogues no less than 83.1% failed because there was no (proper) integration into the context; 14.7% were cancelled by the user, and 2.2% failed because of a system crash.

To resolve semantic and pragmatic problems which result from a high degree of dependency on the dialogue and application context, a system must be able to draw inferences based on a dialogue model and an application model. This requirement goes far beyond the expressive means provided by a structural model, which usually integrates dialogue and application knowledge in a finite-state machine — for theoretical reasons. At this point, another aspect of the close integration of dialogue and application features need to be addressed, although it is not a fundamental theoretical limitation of the structural approach: Because these systems usually are tailored to a specific task type, mostly static database querying, they lack of a sufficient modularization. The need to factor out proper dialogue features is hardly addressed. Therefore, attempts to build in extensions or to port to new domains within the same task type are faced with a considerable technical effort, which constitutes a bottleneck to scalability. Porting to a new domain with different task types will often result in a complete reimplementations of the

structural dialogue model.

To coming back to the previous argument: The need for reasoning becomes even more apparent if we have to deal with meta-discourse, a phenomenon which is frequent in corpora, and with dynamic domains.

To enable flexible, robust and generic dialogue management, we aim at an approach characterized by a logically precise modelling of the linguistic and application related aspects of the domain in question. The dialogue manager has to derive pragmatics-driven discourse strategies from reasoning on evolving belief structures, both of the user and the system. For the linguistic part, we claim that a “pragmatics-first” view on rational interaction provides an appropriate framework for flexible and scalable dialogue modelling. In particular, the plan-based approach offers the means to conduct task- or goal-oriented dialogues which aim at accomplishing concrete tasks. It enables cooperative response behaviour and the ability for negotiation.

2.2 Reconsidering Basal Decisions

The discussion of the state-based approach to natural language and speech dialogue systems indicated a need to reconsider basal design decisions. That means, we have to provide arguments and reasons which help with the search for solutions, in other words, we are looking for theoretical justifications for those decisions. In the following text, “theory” will address the spheres of language and reasoning. As for system architecture, it still seems to be an art and engineering practice, and we will be happy if we are able to derive at least some constraints on architectural designs.

In the following sections, we will set up a general perspective on dialogue systems that can be characterized as a “pragmatics-first” view on rational interaction¹.

For dialogue modelling, we will follow the **plan-based** approach that has its roots in natural language processing and Artificial Intelligence. It provides the means to conduct task- or goal-oriented dialogues which are focussed on accomplishing concrete tasks as mentioned in the introduction. We claim that only a general planning approach enables cooperative response behaviour (pragmatic adequateness, overanswering) and the ability for negotiation.

For the reasoning part, i.e. knowledge representation and inference for the interpretation of dialogue as well as for planning to satisfy user goals in the application domain, we will refer to a computational logic framework, in particular description logics.

Due to a clear functional separation between the language model, the dialogue model, and the domain model, the feedback cycle between the dialogue manager and the applications is of vital importance for system functionality, as is the close interaction on the linguistic side occurring between the speech parser, the dialogue manager and the text

¹From a theoretical perspective see the pioneering work by Cohen and others, cf. the contributions in the volume [7]. As an example for a system strictly based thereon cf. Sadek’s ARTIMIS, [17–19]

generation module. Furthermore, such a division of labor provides a sufficient condition to address scalability.

To give an example, let us consider the scenarios which the EMBASSI system has to deal with. EMBASSI is a joint national project sponsored by the German Ministry for Research and Technology with 19 partners from industry, research institutes and universities². Its goal is to develop a system which is able to control devices by means of multi-modal dialogues, e.g. an audio-video theatre at home or a radio and navigation device in cars. Particular emphasis is put on flexible assistance concepts in the realization of the EMBASSI system. For the first application area, there is the explicit requirement that the application system is reconfigurable, i.e., that new devices can be integrated in a plug-and-play fashion on the fly.

The technical answer of the EMBASSI consortium to the quest for such a high level of flexibility and scalability was to design a highly modular agent-based system architecture where the modules communicate in a uniform agent communication language (KQML/ACL). In the following, we will address the question of the theoretical foundations and basal decisions underlying any dialogue model.

2.3 Dimensions of Scalability

The requirement for scalability of dialogue systems often is seen primarily as an engineering problem. At a closer look, it becomes apparent that there is more to it: Scalability must be considered in the context of underlying assumptions and specifications, which of course concerns various technical aspects, but also includes the question on how to understand the interaction between user and system.

The following issues have to be addressed. However, the list is open for extensions:

- vocabulary size,
- linguistic coverage and utterance types, in particular varieties of reference, e.g. anaphor and ellipsis resolution,
- multi-linguality,
- mixed-initiative dialogues,
- multi-modality,
- single or multiple simultaneous goals,
- multi-party dialogue,
- size and fundamental properties of the application domain(s)
- extensibility of applications by new subdomains, e.g. train information plus ticket sale plus hotel information and reservation,
- switching to new (sub-)domains,
- reconfigurability of the application,
- system portability to new domains.

Of course, many authors have addressed problems mentioned in this list. In the following, we will briefly discuss theoretical considerations that are underlying any systematic approach to solutions to some problems mentioned above.

²Grant No. 01 IL 904 F 8

3 Theoretical Considerations

The term “theory”, in particular linguistic theory and logic, will be understood in a very general and explicitly non-formalistic sense, which means that foundational issues are part of our theoretical reflection. For the latter, the hardest part lies in the beginnings: We have to become clear about the basal assumptions we are starting with and we have to make them explicit in order to avoid argumentative cycles in the construction of our terminology. Considerations of the actual, developed level of theory should include an understanding of its genesis — how did we get to where we are? —, and in particular where opportunities for alternative development paths had been.

3.1 A Basis for Rational Dialogues

Under the assumption that the “language as action” perspective provides a flexible and extensible framework for rational dialogues, whose aim is to satisfy user goals in a given application context, we need means to identify such goals and to represent them formally within an explicit representation of an initial situation. We also need methods to decompose a goal into subgoals to be satisfied by the application system, and to control the satisfaction process. Interactions are called “rational” because we want to apply rationality principles (at the knowledge representation level) to optimally select appropriate communicative actions. In other words, we formulate a complex planning problem which comprises at least two levels: planning on the dialogue level w.r.t. interactions between the dialogue system and the user, and planning on the level of the application system.

For planning on the dialogue level, we need an explicit representation of dialogue situations which include statements representing what the system could extract from the interaction with the user so far and assumptions about the user’s knowledge about the actual situation as well as on goals, their subgoals and the actual state of their satisfaction. From the system engineering point of view we are dealing with the epistemic level and there is no need for stronger mentalistic claims as far as the user is concerned. The planning process consists in the application of dialogue operations which have preconditions defining their applicability and assertions about their effect, i.e., how the dialogue situation develops when they are applied.

A general logic-based approach for representing and processing dialogue situations on the epistemic level has been developed by Cohen, Levesque and others³. How rationality principles can be integrated in such a framework has been shown by Asher et al. (e.g. in [3]). Grice’s conversational maxims as e.g. cooperativity and sincerity are represented axiomatically in a modal logic formalization. A comprehensive framework for discourse planning has been established by Grosz and Sidner in their pioneering investigations [11, 12], who in fact proposed three levels for modelling task-oriented

³cf. [6] and further contributions in the volume [7]; cf. also [14, 15]

discourse structure⁴. The *intentional* level records the beliefs and intentions of the dialogue partners regarding the tasks and subtasks to be performed. The *attentional* level captures the changing focus of attention in a dialogue using a stack of so-called “focus spaces” organized around the dialogue tasks. The *linguistic* level represents “segments”, i.e. contiguous sequences of utterances, which contribute to a particular task.

These theoretical studies have been very influential for a lot of systems although we are not aware of comprehensive implementations yet. To quote only a few examples, cf. Rich’s et al. COLLAGEN system [16], Allen’s et al. TRIPS [1] or Sadek’s et al. ARTIMIS [17–19]. In ARTIMIS, Sadek uses a modal logic theorem prover, which introduces a high degree of flexibility and extensibility, but — probably — for the cost of losing completeness. Beyond recognizing user intentions Rich et al. show how plans can be recognized by inferring intentions from actions. Of course there is still a huge need for research into dialogue strategies as clarification, negotiation, and other subdialogues, and on metadialogue.

Our own work builds up on the insight that planning in dialogues is based on partial knowledge. Each contribution of a dialogue turn is differential w.r.t. the present dialogue situation. Therefore we use a monotonic partial logic [13] — which allows a certain kind of defaults — for reasoning in dialogue situations, including the dialogue context, in order to establish common knowledge and conduct dialogue action(s). Instead of making explicit use of modal logic we instead, according to a suggestion by John McCarthy, refer to a realization of it as interpretation in context. Rationality principles serve as constraints on the planning process.

Discourse planning, i.e. the determination of a sequence of dialogue steps, has to take into account that the application subsystem influences the sequence of dialogue steps by reacting on preconditions of operations, and generating effects which change the actual state. So, a description of the dialogue step sequence requires representations of time, the “actual state”, the terminology of the application, and the operations, their preconditions and effects.

3.2 Reasoning

As Pat Hayes has remarked a while ago, it does not make sense to speak of “non-logical” representations of knowledge. In one way or another, for any knowledge representation schema there is a corresponding logic calculus, even if its authors are not aware of it. Therefore, we argue for a clear commitment to (computational) logic for the dialogue manager, i.e. for the interpretation of dialogue, as well as for the application system it is connected to, i.e. for domain knowledge representation and reasoning. As the application may be any software system in general as a database system, a robot controlling system, etc., we will need to introduce a logical layer between it and the dialogue manager in which the domain model is represented and in which inferences, e.g. in planning, are drawn.

⁴cf. [16]

If we speak about logic, we do that in its traditional understanding, which is far more extensive than modern formal logic: In particular, we address the traditional branches of concept formation, proposition, and inference. Modern formal logic primarily deals with the latter aspect, but in our field of interest the other two are of equal importance.

3.3 Systems for Reasoning

Referring to computational logic has three aspects: First we have the formal logical language, i.e. its syntax and semantics which define a certain expressive power, secondly the level of the reasoning problem, where we have to deal with decidability and computational complexity, and finally the inference procedure with the properties of soundness and completeness. The latter issue also imposes a need for compromises: Often we want to express more than a well-understood complete and sound reasoner can deal with, but then must know exactly what we are doing and we have to keep in mind to stay always as close as possible to completeness and soundness.

Therefore we decided to use description logics [8] as our representational framework. As we will point out in the next section in more detail, we use it for the representation of lexical concepts, as well as for dialogue and application modelling. Using a uniform representation schema on several system levels has the advantage that we need not translate between different level-specific schemata, but there may be a tradeoff. But as long as we stay compatible with evolving web standards as XML/RDF(S) on the syntactic and DAML+OIL on the semantic level, we have the additional advantage that we can use publicly available resources (thesauri, formal ontologies) and tools.

Computational logic gives a framework for representing and reasoning about the functionality of a domain as well. Knowledge about functions is orthogonal to knowledge about notions. When the application domain requires the dialogue system to be capable of talking about actions and changing states of the environment, it is necessary to have logical language available that can reason about the effects of actions. Experience from various applications indicates that the expressivity of the PDDL planning language is sufficient for formalizing the goals a user can have and the states the system can be in in order to assign a precisely defined meaning to user utterances. This approach does not entail to model all the details of the application (and, in particular, its implementation), but only up to the level of detail needed to understand user requests. The strict modularity of the system architecture helps to hide implementation details users do not talk about or even are not aware of at all.

4 System Architecture

Our fundamental design decision consists of a clear functional separation between the language model, the dialogue model and the domain model⁵, and we claim that it provides a sufficient condition to address scalability. We agree with Allen's domain-independence hypothesis [1]: "Within the genre of practical dialogue, the bulk of complexity in the language interpretation and dialogue management is independent of the task being performed."

In addition to the structural aspect of modularization which lays down the components a system has and how they are connected with one another, we have also to consider the temporal structure of their mutual interaction. Although for analysis there is a clear direction of data flow "bottom up", from signal to action (and back to signal for generation), we already indicated the shortcomings of a purely sequential processing direction. Therefore we will have to introduce feedback loops which allow to make use of predictions from "higher-level" components by "lower-level" ones, hence providing opportunities for incremental processing.

4.1 Functional Separation of Dialogue Management and Application

The decision to introduce a clear functional separation between dialogue management and application implies the following interaction steps:

- The dialogue manager "formulates a task" for the application; based on an analysis of speech act and content, i.e. it attempts to construct a plan, e.g. for a request, in terms of a transactional task.
- The transactional task consists of steps which are planning goals for the application.
- The application executes the task stepwise.
- The application decides whether it is necessary to further inquire the user.
- The application sends task results and further inquiries to the dialogue manager such that it can execute appropriate dialogue operations.
- The dialogue manager decides about the satisfaction of the request (plan).

A division of labor between dialogue manager and application in this way is quite radical, but it allows a transparent separation of application and dialogue functions and control flows⁶. As far as the administration of application-specific user goals, and in particular conflict resolution among them is concerned, it has to be provided by the application — as opposed to the administration of dialogue goals cared for by the dialogue manager. Application and dialogue manager are planning separately. The exchange of data must guarantee consistency between the application and the dialogue

⁵represented as two boxes with the "Dialogue Module" box in the center of fig. 1. The "Problem Solver" box is a placeholder; in fact, of course it is a technical application system which is not a part of the dialogue system, but accessible through a suitable interface.

⁶cf. our first paper on functional separation and coordination [5]

situation which, of course, requires semantic compatibility. This in turn presupposes that both, dialogue manager and application, have access to the same domain model. Another consequence of the separation is that it leads to a classification of utterances w.r.t. their functionality to change the dialogue situation.

Scalability is supported because the functionality of the application can be augmented without requiring changes to the dialogue manager. Furthermore, by separating the linguistic base ("language model"), the dialogue model, and the application model the reuse of resources is facilitated.

4.1.1 Application and Dialogue Knowledge

From a structural point of view, the conceptual knowledge (concepts or classes, and roles, i.e. binary relations, for their properties) about application and dialogue is represented in two separate, but formally similar terminological hierarchies. They must be inserted as parallel, but disjoint branches into the system's global conceptual model, which in our case is represented in a description logic. In particular, the application knowledge, which is used in application situation descriptions, consists of

- concept descriptions of domain objects, and
- concept descriptions of domain actions.

These concepts are instantiated in application situation descriptions that are used to represent which objects of which types currently exist and which actions are possible in the current situation.

So, the application concept hierarchy represents formally reconstructed technical or scientific knowledge, combined with elements of common sense under a technical perspective. In specific application domains it may be the case — as with EMBASSI — that a considerable part of the application concept hierarchy, i.e. the device-specific concepts, can be gained automatically from a source provided by the application engineers, which in this special case was given as a Java class hierarchy implementing the device control system. Analogously, the dialogue knowledge used in dialogue situation descriptions, is built up from

- concept descriptions of dialogue objects (utterances, enumeration of alternatives, dialogue goals), and
- concept descriptions of dialogue actions (speech acts).

Dialogue situation descriptions contain instances of those objects; they are extracted from the DRT representation.

The common roof for both hierarchies consists of a generic base model, for which we chose the SUMO formal ontology⁷, into which both are plugged in. Furthermore, a third branch, which contains lexical concepts, is inserted in this global model. The lexical concepts are derived from a structured lexicon, in our case EuroWordNet⁸, and they are linked via a specialization role with concepts of the application and dialogue subhierarchies. To establish this mapping from lexical to domain concepts is a rather labor-intensive process and has to be taken up anew whenever the system is configured for a new application. Furthermore, for lexical entries case frames (thematic roles) have to be provided;

⁷cf. <http://ontology.teknowledge.com/>

⁸based on WordNet, cf. [10].

due to a lack of an appropriate resource for German, we created a tool to facilitate this task [20]. So, (semi-) automatic knowledge acquisition remains as a big problem to be addressed by future research.

4.2 Challenges for the Dialogue Manager

An important task for the dialogue manager is to process the interaction between semantics and the domain model. The general idea is that discourse referents in the domain of discourse refer to instances in the application domain. The interpretation of thematic roles in terms of roles in the application domain is encoded as the application specific part of the word's case frames describing the language usage in the application domain. This means that ABoxes (which describe situations in a plan how to satisfy a user request) have to be consistent with respect to the given application concept hierarchy (TBox).

Usually, semantic representations of natural language utterances cannot directly be mapped into extensional terms of the formal terminology: The meaning of many utterances is not defined by operations of an application. Some utterance parts are not relevant within the domain ontology as e.g. "I would like to..." or "Shall I...". But they are relevant for the dialogue manager to determine the utterance's intended function. "I would like to..." expresses that the user pursues an intention. The dialogue manager must be able to recognize and process this fact. Indeed, it represents a state which can be valid in a given dialogue situation. Hence, a cooperative system must search for circumstances under which the intention can be satisfied.

To process dialogue situations in a flexible (and also extensible) way the dialogue manager has access to a repository of dialogue operations. They are characterized by preconditions and effects w.r.t. the dialogue situation — in analogy to the operations of an application. Dialogue operations are associated to speech acts (or performatives) as "must", "can", "shall", "may", "want", and those expressing conventions (thanking, greeting), etc. The inventory of defined dialogue operations defines a complexity limit for dialogues. In fact, speech acts are implemented as configurable dialogue operations; new complex dialogue operations can be defined in a GOLOG- inspired script language.

In goal-oriented dialogues, a cooperative system aims at a successful execution of the user's dialogue goals on the basis of a plan computed for the goal. In our view, the *choice-operator* of the "practical syllogism" (cf. [2]), in general left open, can be given a precise meaning in terms of planning. Such a plan determines the conditions to execute an actual dialogue goal. If a plan cannot be computed due to missing information, the system has to ask the user — otherwise satisfaction of the dialogue goal fails. This (missing) information often is a necessary condition for the satisfiability of the user goal. Of course, the ability to initiate and conduct clarification subdialogues is a general requirement to the dialogue manager, for example in cases like misunderstandings, recognition errors, or ambiguities in utterances which may occur on all linguistic levels. In order to implement this ca-

pability, we are currently working on the issue of a unified view of components for acoustic, linguistic, and application specific processing as problem solver modules that communicate success of (elementary) operations or a diagnosis for the failure of their execution to the dialogue manager. The dialogue manager takes over the responsibility for integrating the information from a module into a dialogue.

5 Conclusions

In our opinion, this kind of information exchange is characteristic for rational dialogues. In the search for answers to the question how theory can contribute to the construction of systems that can handle such interactions, we considered theoretical aspects from linguistics and logic.

For the linguistic part, we claimed that "pragmatics-first" view on rational interaction provides an appropriate framework. In particular, the plan-based approach offers the means to conduct task- or goal-oriented dialogues which aim at accomplishing concrete tasks. It enables cooperative response behaviour and the ability for negotiation.

For the reasoning part, i.e. knowledge representation and inference for the interpretation of dialogue as well as for planning to satisfy user goals in the application domain, we argued for a computational logic framework.

There is no doubt that a minimal prerequisite for scalable systems is that they have a modular structure. We argued that a clear functional separation between the language model, the dialogue model, and the domain model provides a sufficient condition to address scalability.

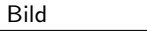
References


- [1] Allen, J. F.; Byron, D. K.; Dzikovska, M.; Ferguson, G.; Galescu, L.; Stent, A.: *Toward Conversational Human-Computer Interaction*, *AI Magazine*, Bd. 22, Nr. 3, 2001, S. 27–37.
- [2] Asher, N.; Lascarides, A.: *Questions in Dialogue*, *Linguistics and Philosophy*, Bd. 21, Nr. 4, 1997, S. 237–309.
- [3] Asher, N.; Lascarides, A.: *Cognitive States, Discourse Structure and the Content of Dialogue*, in *Proceedings of the Amsterdam Dialogue Workshop, Amstologue-99*, Amsterdam, 1999.
- [4] Boros, M.; Wieland, E.; Gallwitz, F.; Görz, G.; Hanrieder, G.; Niemann, H.: *Toward Understanding Spontaneous Speech: Word Accuracy vs. Semantic Accuracy*, in *Proceedings of ICSLP-96 – International Conference on Spoken Language Processing*, Philadelphia, October 1996, S. 1005–1008.
- [5] Brietzmann, A.; Görz, G.: *Pragmatics in Speech Understanding – Revisited*, in Horecky, J. (Hrsgb.): *COLING 82. Proceedings of the Ninth International Conference on Computational Linguistics, Prague, July 5–10, 1982*, Bd. 47 von *North-Holland Linguistic Series*, Academia and North-Holland Publishing Company, Amsterdam, 1982, S. 49–54.

- [6] Cohen, P. R.; Levesque, H. J.: *Persistence, Intention, and Commitment*, in Cohen, P. R.; Morgan, J.; Pollack, M. E. (Hrsgb.): *Intentions in Communication*, System Development Foundation Benchmark Series, Kap. 4, A Bradford Book, The MIT Press, 1990, S. 33–69.
- [7] Cohen, P. R.; Morgan, J.; Pollack, M. E. (Hrsgb.): *Intentions in Communication*, System Development Foundation Benchmark Series, A Bradford Book, The MIT Press, Cambridge, Mass. and London, 1990.
- [8] Donini, F. M.; Lenzerini, M.; Nardi, D.; Schaerf, A.: *Reasoning in Description Logics*, in Brewka, G. (Hrsgb.): *Foundations of Knowledge Representation*, CSLI Publications, Stanford, 1996, S. 191–236.
- [9] Eckert, W.; Görz, G.; Hanrieder, G.; Hiltl, W.; Niemann, H.; Schukat-Talamazzini, G.: *A Robust Information System for Spoken Human-Machine Dialogues*, in *Proceedings of ICASSP-94*, Adelaide, April 1994.
- [10] Fellbaum, C.: *WordNet –An Electronic Lexical Database*, Language, Speech, and Communication, MIT Press, Cambridge, Mass., 1998.
- [11] Grosz, B. L.; Sidner, C. L.: *Attentions, Intentions, and the Structure of Discourse*, *Computational Linguistics*, Bd. 12, Nr. 3, 1986, S. 175–204.
- [12] Grosz, B. L.; Sidner, C. L.: *Plans for Discourse*, in Cohen, P. R.; Morgan, J.; Pollack, M. E. (Hrsgb.): *Intentions in Communication*, System Development Foundation Benchmark Series, Kap. 20, A Bradford Book, The MIT Press, 1990, S. 417–444.
- [13] Nait Abdallah, M.: *The Logic of Partial Information*, Springer, New York, 1995.
- [14] Poesio, M.; Traum, D.: *Conversational Actions and Discourse Situations*, *Computational Intelligence*, Bd. 13, Nr. 3, 1997, S. 309–347.
- [15] Poesio, M.; Traum, D.: *Towards an Axiomatisation of Dialogue Acts*, in Hulstijn, J.; Nijholt, A. (Hrsgb.): *Proceedings of the Twente Workshop on the Formal Semantics and Pragmatics of Dialogues*, Enschede, 1998, S. 207–222.
- [16] Rich, C.; Sidner, C. L.; Lesh, N.: *COLLAGEN – Applying Collaborative Discourse Theory to Human-Computer Interaction*, *AI Magazine*, Bd. 22, Nr. 3, 2001, S. 15–25.
- [17] Sadek, M.: *A Study in the Logic of Intention*, in *Proceedings of ECAI-96*, 1996, S. 462–473.
- [18] Sadek, M.: *Design Considerations on Dialogue Systems: From Theory to Technology – The Case of Artimis –*, in *Proceedings of the ESCA Workshop “Interactive Dialogue in Multi-Modal Systems”*, Kloster Irsee, 1999, S. 173–187.
- [19] Sadek, M.; Bretier, B.; Panaget, F.: *ARTIMIS: Natural Dialogue Meets Rational Agency*, in *Proceedings of IJCAI-97*, 1997, S. 1030–1035.
- [20] Thabet, I.; Ludwig, B.; Schweinberger, F.-P.; Bücher, K.; Görz, G.: *Using EuroWordNet within the Speech Operated System EMBASSI*, in Kunze, C. (Hrsgb.): *GermaNet Workshop: Anwendungen des deutschen Wortnetzes in Theorie und Praxis*, *Proceedings*, Gesellschaft für Linguistische Datenverarbeitung e.V. (GLDV), Tübingen, October 2003.

Kontakt

Günther Görz
 Lehrstuhl Informatik 8 (Künstliche Intelligenz)
 Universität Erlangen-Nürnberg
 Haberstraße 2, 91058 Erlangen
 E-Mail: goerz@informatik.uni-erlangen.de

 Günther Görz studied mathematics, computer science and philosophy at the University of Erlangen-Nürnberg, where he also wrote his Ph.D. thesis on structural analysis of spoken language. From 1972 to 1987 he worked at the university computation center, then until 1989 in the LILOG project of IBM Germany in Stuttgart. Until 1991 he was professor of computer science (Artificial Intelligence) at the university of Hamburg. Since then he has been professor of computer science at the chair of Artificial Intelligence of the University of Erlangen-Nürnberg.

 Bernd Ludwig studied computer science, specializing in theoretical computer science, with a minor in classical greek philology from 1992 to 1997 at the University of Erlangen-Nürnberg. From 1997 to 2004 he worked as a research assistant at the chair of Pattern Recognition of the University of Erlangen-Nürnberg. During that time, he earned his Ph.D. in computer science, specializing in dialogue systems. Since May 2004 he has been assistant professor at the chair of Artificial Intelligence of the University of Erlangen-Nürnberg.