Representing Knowledge of Dialogue, Domain, Task and User in Dialogue Systems - How and Why?

Annika Flycht-Eriksson
Department of Computer and Information Science
Linköping University, SE-581 83, LINKÖPING, SWEDEN
annfl@ida.liu.se

March 22, 2002

Abstract

Dialogue systems utilise a variety of knowledge sources and models. However, the purposes and contributions of specific knowledge sources and the relationships among them are confusing. In this paper, a study of different dialogue systems and the knowledge sources and models they use is presented. The knowledge sources are characterised in terms of what knowledge they contain. The roles of the various knowledge sources and the relations between them are discussed in light of the capabilities needed by dialogue systems to achieve natural interaction with users. Implications for development of dialogue systems are also presented.

1 Introduction

Today much research is carried out within the area of natural language interfaces and dialogue systems, and more and more systems are becoming publicly available. Most of the systems provide information services or assistance in solving a specific task. The systems differ in complexity due to the domain and the approach taken in the design of the system. Some systems are highly knowledge-intensive and contain several interacting knowledge sources and models, while others rely on much simpler models and procedures. The variety of dialogue system architectures that incorporate various knowledge sources and models has led to confusion when it comes to the purpose and contributions of a specific knowledge source or model. The relations between various knowledge sources and models are also diffuse and often several types of knowledge are mixed in one model or knowledge source.

This mixture of knowledge types has a number of drawbacks, especially for research systems and systems not primarily developed for limited dialogue in one application. First of all it is hard and time-consuming to port a system to a new domain or task since all of the knowledge has to be replaced. Another related problem is that it is difficult to experiment with a system's behaviour, for example trying different dialogue strategies, since a change in some item often causes other changes. The lack of clear boundaries between models also makes it difficult to reuse and incorporate previous work done by others. These problems indicate that knowledge of various types should be separated. For this purpose definitions of the different knowledge sources and models for the different types of knowledge used by dialogue systems are desired.

To decide which types of knowledge sources and models to incorporate in a dialogue system, the role of the the knowledge sources and models has to be clarified. One way of doing this is to define the role of a knowledge source or model in terms of the capabilities it supports. Given such a mapping between knowledge sources and capabilities, and a characterisation of a system's intended functionality, the set of required knowledge sources and models in the dialogue system can be derived.

The objective of this paper is twofold; to characterise and categorise the various types of knowledge sources and models used in information-providing and problem-solving dialogue systems, thus clarifying the sometimes confusing terminology; to map the knowledge sources and models to dialogue system capabilities considered useful to achieve natural interaction, thus creating a tool that can be used to facilitate the design of knowledge sources in dialogue system.

The first issue is addressed by a survey of eight information-providing and problem-solving dialogue systems that are examined in light of the knowledge sources and models they use, described in section 2. To address the second issue a set of guidelines and development principles for dialogue systems are used to compile a list of dialogue system capabilities, presented in section 3. Section 4 presents an analysis of how these capabilities are captured in the different knowledge sources is made. This analysis is mainly based on how the knowledge sources and models are used in the surveyed systems. The result of the analysis is summarised in a table that can support design of dialogue systems, which is briefly discussed in section 5

2 Knowledge sources in dialogue systems

To characterise and categorise knowledge sources and models in dialogue systems a study of eight different information-providing or problem-solving dialogue systems has been conducted. The surveyed systems are: CIRCUIT FIX-IT SHOP, GALAXYII, LINLIN, RAILTEL, SUNDIAL, TRAINS, VERBMOBIL, and WAXHOLM[1].

The term **knowledge source** will be used to denote a component in a dialogue system, which consists of a knowledge model and mechanisms used to manipulate and reason about the information held by the model. The types of knowledge sources used for dialogue management in dialogue systems are related to knowledge of dialogue and discourse, task, user and domain. Although there are other models, this selection represents the most common types of knowledge sources and models utilised in information-providing and problem-solving dialogue systems today.

2.1 An overview of the surveyed dialogue systems

Dialogue systems are developed for very different tasks and domains. In this survey the focus is on two types of dialogue systems, *information-providing* systems and *problem-solving* systems[2], leaving out other types of systems

[1]The systems have been chosen to cover most types of dialogue systems: commercial or research, plan-based or grammar-based, general purpose or domain-specific, and natural language only or multi-modal

[2]Since there is no consensus on classifications of dialogue systems, I have made a distinction between information-providing and problem-solving systems, both of which

such as argumentation and tutoring systems. **Information-providing** systems are systems where the user wants some information that is available in one or more background systems. The system helps the user to construct an information request that is specific enough for the system to retrieve the information and present it to the user. Such information-providing dialogue systems are also called *simple service systems* [23]. **Problem-solving** systems collaborates with the user to solve tasks that he or she cannot solve alone. Usually, the human knows about strategies and the system has information about the problem and various details. Thus, it is very important that the system and user co-operate to make the best of their specialties. In this survey the GALAXYII, LINLIN, RAILTEL, SUNDIAL, and WAXHOLM systems belong to the information-providing category, and the CIRCUIT FIX-IT SHOP and TRAINS systems are instances of the problem-solving class of systems.

2.1.1 CIRCUIT FIX-IT SHOP

The CIRCUIT FIX-IT SHOP system is an example of a problem-solving system that assists a user as he or she is fixing an electric circuit. The system is based on the Missing Axiom Theory [37] in which completion of an action is viewed as a theorem and the process of accomplishing an action or a task is the same as proving the theorem. If a task cannot be completed, it is interpreted as a missing axiom which has to be provided by the user through a dialogue with the system.

The system consists of five components: a dialogue controller that is the supervisor of the system, a domain processor that holds information about the application domain, a general reasoning component responsible for theorem proving, a linguistic interface component for interpretation and generation of utterances, and finally a knowledge module which represents knowledge about the dialogue, the user and the actions that can be performed.

2.1.2 GALAXYII

The GALAXY system, followed by the GALAXYII system, developed by the Spoken Language Systems group at MIT, is a distributed multi-modal multi-domain system that provides users with information about, for example, travel and weather [20]. The GALAXYII system consists of several specialised modules, e.g. for speech recognition and understanding, dialogue management and context tracking, application back-ends, generation and speech synthesis. The different modules interact via a hub using a common knowledge representation format, semantic frames [35].

2.1.3 LINLIN

The LINLIN system[3] [2] is a natural language dialogue system that has been customised for various domains, e.g. information on second-hand cars and charter trips to the Greek archipelago. The system includes a generator, a parser, a database interface, an instantiator, and a dialogue manager that is the kernel of the system. The dialogue manager is responsible for controlling the interaction with the user and also controls the other modules.

are sometimes referred to as task-oriented.

[3] LINLIN is strictly speaking a framework that can be used to implement various dialogue systems. The name has, however, also been used to denote some customised systems, and this is how it will be used in this paper.

2.1.4 RailTel

The RailTel system is an example of a practical system that is publicly available and currently in use [7]. It provides information over the telephone about railway journeys. The system is composed of a number of components, such as a speech recogniser and a natural language component, a dialogue manager, and a response generator. The architecture is serial: information represented as semantic frames flows from the speech recogniser to the language understanding module, on to the dialogue manager which accesses the application and forwards the result to the response generator.

2.1.5 sundial

The sundial system has been developed within the SUNDIAL project (Speech UNderstanding in DIALogue) where several dialogue systems for information exchange over the telephone have been created for different languages [24]. The system provides information over the telephone about air and train traffic. The system contains a component for speech recognition, a parser, a dialogue manager, a text generator, and a text-to-speech system. The dialogue manager in the sundial system has a distributed architecture, and consists of five modules: a linguistic interface, a message generator, a dialogue module, a belief module, and a task module, which can be seen as independent agents. Its purpose is to interpret the input that has been analysed by the parser, decide how to continue the dialogue, and plan the system utterances [9].

2.1.6 trains

The trains system is used for mixed-initiative collaborative planning in the transport domain [6]. It is task-oriented, giving the user advice on how to perform a task in the real world outside the system. The system is designed as an agent having a mental state and the dialogue management is plan-based. It is based on the common BDI (Beliefs Desires Intentions) model which have been adapted to conversation between two agents. The first system trains-90 has over the years been redesigned and improved and there are a trains-93 [38], trains-95 [18] as well as a trains-96 [19] system. The trains-96 system consists of a number of modules for interpreting and generating natural language, dialogue processing and domain reasoning. The modules are connected in a star-like fashion and message passing is handled by a specific processor. KQML (cf. [40]) is used as the communication language.

2.1.7 Waxholm

The Waxholm system provides users with information about boat traffic in the Stockholm archipelago. It has much in common with RailTel, both systems are domain-specific spoken dialogue systems originating from the speech research community. However, the Waxholm system also allows for limited multi-modal interaction [10]. The interaction between modules within the system is based on the use of semantic frames. The module for speech recognition and interpretation delivers a semantic frame representation of a request to the dialogue manager, which decides how to respond to the request and updates the semantic frame. The updated frame is then sent to the components responsible for generation of speech and graphics.

2.1.8 VERBMOBIL

The VERBMOBIL system is not a traditional dialogue system as the system is not an active participant in the dialogue. Instead, it listens in on two persons having a conversation in a language that is not their mother tongue, e.g. a German and a Japanese trying to book a meeting speaking English. The system's task is to monitor the dialogue and to give the users translations when required [4].

The system has two different processing modes. When the dialogue participants are speaking English and no translation is necessary, only shallow processing takes place. This means that the system only looks for keywords and tries to identify the speech act performed. In this way, the system knows at what stage the dialogue is. When a translation is requested, the system enters a mode of deep processing where utterances are analysed with respect to prosody, syntax, semantics and pragmatics. Besides modules for keyword spotting, speech recognition and language analysis, there are modules for semantic construction and evaluation, transfer, and generation [3].

2.2 Characteristics of knowledge sources

Even if dialogue system architectures differ all systems need and utilise similar knowledge. However, the terminology used to describe the knowledge sources and models varies between the systems. Therefore a uniform terminology is introduced in this section, and the various knowledge sources and models used by the systems are characterised in terms of the type of knowledge they represent.

2.2.1 Discourse and dialogue knowledge

In dialogue systems two aspects of the dialogue need to be modelled, a generic description of the structure of dialogues that can be used to form a coherent dialogue with the user, and a dynamic representation of the current dialogue. I will refer to the first as the dialogue model and the second as the dialogue history. The dialogue model is often closely connected to and dependent on the information represented in the dialogue history.

Dialogue models have the common purpose of describing how the system should respond to user utterances, for example by accessing a database or asking for clarifications. The general information about the dialogue, held by the dialogue model, is often based on a representation of relations between the constituents of a dialogue. This knowledge is used to control the interaction, i.e. to decide what action to take in a certain situation.

There are several approaches to dialogue modelling used today: the most common are state transition networks, grammar-based, and plan-based approaches.

State transition networks are a rather simple method for modelling and controlling dialogue flow. The system responds to a user utterance by moving from one state to another, thus traversing the network and producing a sequence of exchanges between the user and the system. The advantage of this type of model is its simplicity, given that the task being modelled is well-structured and consists of a limited number of necessary exchanges. If the task is highly complex, the transition networks tend to get unmanageable. This is also true if the model should contain repair strategies and allow the user more unrestricted input [29].

Dialogue grammars have a rather long history and are based on the notion of adjacency pairs [32]. Adjacency pairs express the fact that speech acts typically form a regular sequence, such as a question followed by an answer. Rules in the dialogue grammar capture the sequential and hierarchical constraints of dialogues in the same way that grammar rules describe the syntactic structure of a sentence.

Plan-based models go beyond the utterance and try to model the speaker's intentions and goals [12]. Communication becomes a part of the speaker's overall behaviour. Elements in these models are plans, actions, mental states and mechanisms for recognising a specific plan and for reasoning about the speaker's beliefs, intentions, and actions.

The dialogue models used by the systems in the survey incorporate both grammar-based and plan-based dialogue modelling. The RAILTEL, SUNDIAL, LINLIN, and WAXHOLM systems have a dialogue model consisting of a *dialogue grammar* that models the dialogue's hierarchical structure. In the RAILTEL system the dialogue is partitioned into three phases each consisting of a number of sub-dialogues. It is modelled by a hierarchical structure of sub-dialogues and dialogue acts, represented by a set of rules in a dialogue grammar [7]. A similar approach is taken by SUNDIAL [9] and the LINLIN system [25], which model the dialogue using a dialogue grammar and speech acts.

The dialogue model in the WAXHOLM system is based on the idea that the dialogue should be described as a grammar and at the same time be probabilistic [11]. The WAXHOLM system thus has a dialogue grammar represented as finite state machines but also a *statistical* component for topic prediction, which is part of the dialogue model.

A combination of approaches is also utilised in the VERBMOBIL system. The possible moves for the user and system to make during the interaction are represented in a dialogue model using dialogue acts. The dialogue module in the VERBMOBIL system consists of three sub-modules: a Statistical Module, a Finite State Machine, and a Dialogue Planner. The Statistical Module can, given a dialogue state, predict all possible successive dialogue acts and their likelihood. The Finite State Machine has a similar task: it checks whether a dialogue act is consistent with the underlying dialogue model and which subsequent dialogue acts are possible. The Dialogue Planner is the most sophisticated of the three sub-modules. Plan operators are utilised to plan the dialogue and handle different phases like negotiations, clarifications and repairs. A plan is built up hierarchically with the leaves in the hierarchy being dialogue acts [5].

The TRAINS system takes a distinct *plan-based* approach to dialogue modelling. The system is viewed as a conversational agent having goals, intentions, plans, and obligations. Answers to user utterances are planned by the system in order to reach its goals and at the same time fulfil its obligations [38].

Another way to handle the dialogue is to do it more procedurally as in the GALAXYII system where the dialogue is controlled by a stepwise process. This means that there is no explicit dialogue model as the model lies implicit in the processing algorithm [34].

The CIRCUIT FIX-IT SHOP system also lacks an explicit dialogue model; instead handling the dialogue is tightly coupled to the task model. The task model is used to find missing axioms that result in clarification dialogues and also to insert sub-tasks corresponding to sub-dialogues initiated by the user [37].

The dialogue model is often used together with information about the dialogue state. The variety of ways to model the state of the dialogue has led to a variety of names, e.g. discourse memory, discourse history, dialogue memory, dialogue history, history table, and context model. To avoid confusion only the term dialogue history will be used. Dialogue histories could then be described as partial or full depending on the number of information levels, and the degree of structure could also be used for further classification.

The dialogue history in a dialogue system represents the state of the dialogue, that is, what has been talked about and the current topic of the dialogue. It can be used for dialogue control, disambiguation of context-dependent utterances, and context-sensitive interpretation, e.g. reference resolution and handling of ellipsis.

The representations of dialogue histories range from complex hierarchical structures, containing a variety of information, to much simpler sequential representations. The kind of dialogue history required in a dialogue system depends on the linguistic phenomena that have to be handled, such as misunderstandings, interruptions, and deictic expressions, and the complexity of the task and domain which is reflected in the interaction. If a dialogue is made up of several clearly separated sub-dialogues, a more structured representation may be preferred to a sequential.

In the LINLIN system a dialogue tree consisting of dialogue objects is constructed during the interaction [25]. The tree has three levels which correspond to the whole dialogue, discourse segments called initiative response units, and dialogue moves. A dialogue object contains situation parameters and content parameters. The first holds information about Initiator, Responder and context parameters while the second records the current focus of the dialogue.

The representation of the dialogue history in the SUNDIAL system is distributed over several models. A tree is used to represent the dialogue structure. It resembles the dialogue tree in the LINLIN system but has one more intermediate level to represent a common topic [9]. An interpretation is called a belief and a sequence of beliefs which corresponds to the user's utterances is represented in a contextual model [28]. This model is used to make context-sensitive semantic interpretations of user utterances. Changes in the contextual model are reflected in a flag called status that indicates how the contextual model has changed; the value repeat, for example, means that nothing new has been contributed. This information can be used by the dialogue module to reason about the function of an utterance, e.g. if it is a confirmation or a new request [24].

The dialogue planner in the VERBMOBIL system contains a dialogue history representing intentional, thematic and referential information. Intentional information corresponds to the dialogue acts performed by the participants, and is structured in a tree-like representation. The thematic information refers to relevant information in utterances while referential information is the lexical realisation of utterances. A representation of the proceeding dialogue is constructed dynamically in the dialogue memory during the dialogue [5].

Some of the other systems focus only on the objects that have been mentioned, which could be compared with the attentional level of the discourse using Grosz & Sidner's terminology [22]. The GALAXYII, RAILTEL, and WAXHOLM systems all maintain a history of semantic frames that are used to represent the objects and relations in focus. The discourse module in the GALAXYII system maintains a history table of referable objects to facilitate the interpretation process. Items and requests are represented internally as semantic frames, and the history table consists of a sequence of frames [34]. In the RAILTEL system the context is represented by a dialogue history and a generation history which contain the semantic frame representation of the user's and the system's previous utterances respectively [7]. The utterances are stored sequentially within these. The WAXHOLM system utilises a dialogue history based on the semantic frame representation and the updates of this representation [11].

The dialogue history in the TRAINS system lies somewhere between the complex multi-level and simple frame-based representations, as it models both attentional and intentional features. The discourse is modelled as a stack of discourse units (DUs) which represent utterances and the corresponding speech acts. In the model the initiator and state of DUs are also recorded. Two other contextual features represented in the system are the turn and the local initiative, each of which is held by one of the dialogue participants. The turn indicates who is speaking now, while the local initiative informs which speaker is obliged to speak next [38].

The CIRCUIT FIX-IT SHOP system differs from the other systems as it focuses on what might be said next instead of what has been said earlier. Thus, the attentional state is represented as a set of expectations about the possible responses from the user [36].

2.2.2 Domain knowledge

The amount of domain knowledge needed in a dialogue system differs depending on the domain and the system's task. This means that the sources of domain knowledge can range from rather simple conceptual models to full-fledged world models capable of complex reasoning. Dahlbäck & Jönsson [17] make a distinction between domain models and conceptual models. The domain model represents the structure of the world and usually comprises a subset of general world knowledge, while the conceptual model represents the general and domain-specific concepts. I will also use this distinction, but since reasoning is central to what Dahlbäck and Jönsson call domain models I will instead use the term domain knowledge source.

Information from the conceptual model and the domain knowledge sources is primarily used to identify the relevant items and relations that are discussed, reason about and derive new information from the information provided by the user, to supply default values, etc. In information-providing systems the knowledge represented in a domain knowledge source is often closely connected to the background system, e.g. a database system. In such cases domain knowledge is used to map information in the user's utterances to concepts suitable for database search.

The semantic frames used in the RAILTEL system contain the relevant domain concepts and serve as a simple conceptual model. The domain knowledge also consists of two kinds of rules: *Default value rules* supply default values, e.g. the current or next month for a departure date; *Interpretative rules* transform vague qualitative values into more precise quantitative values used by the system, for example they map the concept "morning" onto the precise interval "between 6 a.m. and noon" [7].

The domain knowledge in the SUNDIAL system is distributed. One module contains a hierarchy of surface-oriented concepts while another module contains a hierarchy of task-oriented concepts. The surface-oriented concepts, for example "at noon", are mapped onto task-related concepts, "at twelve o'clock". The domain knowledge thus consists of two concept hierarchies and inference rules used to map one to the other [24].

Domain knowledge in the VERBMOBIL system is represented in a conceptual hierarchy that represents relations between different categories, entities, and situations. Situations are entities determined by spatial and temporal features like year, month, week, day, location, etc. The hierarchical structure of the model makes it suitable for decisions about possible references. It also allows inheritance; if a temporal object is available for scheduling, the super-ordinate object is regarded as free too, and similarly, if an object is not available, none of the subordinated objects are either [30].

The TRAINS system on the other hand has a more complex representation of domain knowledge. The Domain Plan Reasoner in the system is responsible for the representation of and reasoning about the domain. It maintains a representation of the state of the world, provides new suggestions about routes, and adjusts the current plan given new constraints. Inspection of plans that have been suggested by the user is also done to check whether any unknown conditions need to be considered [18].

The CIRCUIT FIX-IT SHOP system also needs substantial knowledge about the domain. The domain knowledge is divided into a conceptual model and a domain model. The conceptual model represents how different components of a circuit are related to each other and how the different components should function. The domain model consists of a set of axioms that represent the current state of the circuit that is being fixed [36]. The term domain model thus has a different meaning in this system since it is more of a representation of a state than general knowledge about the domain.

In the GALAXYII system conceptual models and domain knowledge sources are combined for some domains. The domain knowledge in the GALAXYII system can be found in two places, declarative tables in the discourse module, and in domain servers that contain the application data. The tables can be seen as conceptual models which describe the possible semantic classes a value can have and some relations between them. The domain servers can contain more specific domain knowledge needed to interpret the semantic frame [34].

In the LINLIN system a conceptual model has been used in the travel domain. It consists of a hierarchy with concepts for charter trips, such as resort, hotels, etc. It is used for focus tracking and to resolve anaphoric expressions [17].

The WAXHOLM system contains no explicit domain knowledge sources; domain knowledge is instead incorporated in the lexicon and the grammar. The parser is based on a semantic feature structure with features of two kinds, basic features and function features. The basic features, e.g. boat and place, provide simple semantic information about a word. They are organised in a hierarchy. The function features, e.g. departure_time, to_place, do not have the same structure and they are associated with actions rather than objects [10].

2.2.3 Task knowledge

The terms task and task model are often used when describing dialogue systems, but they can refer to very different phenomena. It is important to make a clear distinction between the system's task(s) and the user's task(s). A user task is non-linguistic and takes place in the real world outside the system. Models of such tasks represent the user's goals, and knowledge of how they can be achieved. Models of system tasks describe how the system's communicative acts and other tasks, e.g. database access, are carried out.

Dahlbäck [16] uses the term *dialogue-task distance* to describe the degree of connectedness between the user's non-linguistic task and the dialogue structure. For some types of dialogues it is important that the system knows what nonlinguistic task the user is performing or is planning to perform. In these cases the system can often infer the necessary information from the linguistic information. For other types of dialogue, where the dialogue-task distance is long, information about the user's task is less necessary.

Depending on dialogue-task distance different dialogue models are suitable. For advisory or problem-solving dialogues a plan-based model, in which the user's intentions and goals are represented, might be appropriate. For these types of dialogues, there is a close connection between the task and the language, which means that it is possible to infer the non-linguistic intentions behind an utterance. In simple human-computer interaction for information retrieval, the connection between the task and the dialogue is weaker. The user's task is not necessarily reflected in the dialogue that seemingly only consists of information exchanges. This makes it harder to infer the underlying intentions if one wants to model these in the system. On the other hand that type of information is not always necessary for the system to be able to respond appropriately. For this type of interaction a dialogue grammar is sufficient [16].

User task models can vary in complexity depending on the amount of information that has to be exchanged, the structure of the task, and the negotiation necessary. A user task model can consist of a hierarchical representation of sub-tasks which captures the task structure. The structure or lack of structure is important. If a task is ill structured, the system cannot predict what kind of information the user will request and when. In a well-structured task with a predefined, at least partially ordered exchange of information, it is much easier for the system to model the interaction.

In an information-providing system a system task model can assist the system in judging if all the required parameters are present and in cases where they are not, determine what type of information to collect from the user. The use of an explicit system task model makes the system more flexible since it is easier to modify or add new tasks.

In this survey the TRAINS and CIRCUIT FIX-IT SHOP systems are the only systems that are problem-solving and have models of the user's task. The task in the TRAINS system is route planning and knowledge about the task, i.e. the planning process is represented explicitly. The plans that are developed during the interaction with the user are represented as a hierarchy of goals that are expanded into sub-goals. The hierarchy is complemented with a linear history of the planning process. For some sub-problems specialised domain servers are used [19].

In the CIRCUIT FIX-IT SHOP system the task model is prominent, and it contains information about goals, actions and states. Actions can consist of a hierarchy of sub-actions. Actions that lack sub-actions are primitive, and specify physical or sensory actions that users can perform. Goals can be accomplished by performing one or more actions. States can be either physical, representing properties or behaviours, or mental, representing beliefs about the physical state or the user's abilities. The task model can be used to answer questions about the physical state or determine what action to recommend the user to perform [36].

The other systems are all some sort of information-providing system providing information retrieved from databases. In the LINLIN, VERBMOBIL and WAXHOLM systems the system task knowledge is integrated with other knowledge sources and models, and lies implicit in the system. The GALAXYII, and RAILTEL systems have a frame-based specification form that is more or less utilised as a system task model. In the GALAXYII system there is no explicit task model but a frame representation is used to see if all the required slots are filled and, in case some are missing, ask the user for a clarification [34]. In a very similar way task rules in the grammar of the RailTel system are connected to the semantic frame representation of the provided information and sub-dialogues are initiated if information is missing in the frame [7].

The SUNDIAL system utilises a more sophisticated system task model that represents the task structure and keeps track of the status of the task. This often means deciding whether the user has given enough information and if not, what has to be further provided. Another role is to handle situations that arise when the provided information is incorrect. In this case the task model is used to relax the parameters instead of returning a negative answer. For example, if the user has requested a flight at noon and none is available, the system tries to find one in the morning instead [28].

2.2.4 Knowledge of the user

User models represent the user's goals and plans, capabilities, attitudes, and knowledge or beliefs. They vary in complexity, ranging from user stereotypes [31] to complete models of the user's knowledge, intentions, attitudes, etc. [26].

Depending on what kind of information a user model contains it can be used for various purposes. If a user model represents what the user knows about the domain, the system can adapt its answers so that they are informative and easily understandable. User models can also be utilised for dialogue control.

Kass & Finin [26] discuss when user models can be of great assistance and when they can be left out of a system. In simple question-answering systems no user model is necessary. In co-operative question answering systems user models can be more beneficial. For such systems a simple model of the user's goal can be used but a generic model suffices. Co-operative consultation systems on the other hand need an extensive user model.

The TRAINS system is an example of a co-operative system which uses much elaborated models of the and itself. The user model and self model represent the user's and system's mental state and contain information about their beliefs and proposals about the domain. The beliefs can be private to either the system or the user, or they can be shared by both. Mutual beliefs include aspects of the domain plans that are considered to be agreed on by both system and user. An example of private beliefs held by the user are the domain plans that have been proposed but not acknowledged. The system's private beliefs consist of the domain plans the Domain Planner has derived but have not been presented to the user [38].

The user model in the Circuit Fix-it Shop system is a collection of axioms representing the user's knowledge about the state, as it is known by the system. These axioms are used by the system when it tries to prove a theorem and generate missing axioms, i.e. when it decides which sub-actions have to be performed in order to complete a task. For example, if the system knows that the user knows how to perform a sub-task, it can give a higher order instruction instead of explaining and guiding the user through the sub-task [36].

2.2.5 Knowledge sources and dialogue types

To summarise the discussion of the different types of knowledge, information-providing systems and problem-solving systems are contrasted in this section.

Information-providing systems

In information-providing systems the dialogue and the information exchanged through the dialogue are the central features of the system. This is reflected by the fact that the dialogue model and dialogue history are often prominent in such systems.

Information-providing systems are in general less focused on the user's task than problem-solving systems. This is reflected in the lack of user task models. Some systems instead have explicit system task models that represent the information-seeking and information-providing tasks the system. In information-providing systems user models are not necessary, since the execution of the system's tasks is highly independent of the user's knowledge.

The type of domain knowledge needed in information-providing systems also differs from problem-solving systems. If the domain and task are fairly simple, complex domain knowledge sources might be unnecessary when instead a simple conceptual model would suffice.

Problem-solving systems

In problem-solving systems the dialogue between the user and the system is just a means of solving a problem. To identify and co-operate to achieve the user's goals is the primary concern of the system. The focus on the tasks and goals of the user is reflected in the the dialogue model and the dialogue history, which in most cases are intention-based.

Another consequence of the focus on the task is that the user task model is essential, and sometimes takes on some of the responsibilities held by the dialogue model in information-providing systems. For example, the user task model can be used instead of the dialogue model to decide how to deal with sub-goals that are not accomplished.

When a dialogue system is working jointly with the user on a task, it has to build a model of the domain or world that is being altered by actions performed by the user during the dialogue. The domain knowledge sources in problem-solving systems therefore have to be dynamic and in most cases more elaborate than the domain knowledge sources found in information-providing systems. The system also has to keep track of the user's changing knowledge about the domain, thus making a user model a necessity.

2.3 Relations between knowledge sources

To be able to respond properly to a user request in an information providing system, the system has to have knowledge about both the domain and the task. Thus, it can be very tempting to integrate this kind of knowledge in the dialogue model. The task and domain knowledge can also be mixed since the performance of a task is often domain-dependent.

Dialogue systems for information retrieval often lack an explicit system task model, in these systems the representation of task knowledge frequently becomes a part of the dialogue model. This works well in simple domains where the system only performs one or a few similar tasks but if a system is to manage several different tasks explicit task models are preferred.

A typical example of the integration of domain and task knowledge is the use of semantic frames which are part of many information-providing dialogue systems. They represent the relevant domain concepts, and are sometimes coupled to rules for interpretation of domain concepts and rules to fill in default values in a frame. Besides the role of conceptual models, the semantic frames often serve as system task models since they are used to describe what information has to be gathered by the system before a database access. This multiple use of semantic frames is useful as long as the system task is rather simple and well-structured. If the task becomes more complex, separate system task models might be needed.

The relation between user models and dialogue histories has been debated. Opinions range from user models and dialogue histories being completely separate to being two sides of the same coin [27, 33, 13]. The differences in opinion seems to a great extent to depend on how the two types of models are defined. Some researchers focus on the user's goal and compare it to the intentional level of dialogue histories, others look at the user's beliefs and knowledge and compare it to the attentional information in dialogue histories.

3 Capabilities supported by dialogue systems

Choosing natural language as a means for interaction with a system is often motivated by the ease and naturalness of natural language. However, a system needs many capabilities if the dialogue is to be perceived as natural by the user. In this section such capabilities related to dialogue management are presented, and how the capabilities are supported by the knowledge sources and models presented in the section 2 are discussed.

3.1 Graceful and co-operative interaction

Hayes & Reddy [23] describe a set of skills they consider necessary to achieve graceful interaction between a human and an information-providing dialogue system.

The need for domain knowledge[4] is stressed. Two aspects of domain knowledge that should be captured are a priori knowledge about the domain and the services and a specification of how to interact with a user during the formulation of a request. The first is related to three different skills: knowledge about the type and structure of entities in the domain, ability to derive new information from the provided information, and default information. The second involves three different skills: to know what information has been provided so far, what vital information is missing, and the focus of the current conversation.

According to Hayes & Reddy a dialogue system also needs skills for dealing with questions about its capabilities, actions performed by the system, and hypothetical questions. There are also several skills related to the goals and focus of a user interacting with an information-providing dialogue system. The user always has an overall goal with his utterances while the system is assumed to have no other goal than to co-operate with the user. However, on lower levels where the user's goal is divided into sub-goals, the

[4]Hayes & Reddy do not distinguish between domain and task knowledge. Using the terminology from section 2.2 some of the domain-related capabilities would be classified as task-related instead.

system may also have goals. To deal with goals and sub-goals the system must be able to detect and adopt to the user's high level goal, to generate sub-goals, and to know how to achieve a goal. Handling the focus of the conversation requires capabilities to follow shift of focus, and to resolve anaphora and ellipsis.

Finally, since requests in information-providing dialogue systems are specified by providing a set of entities, the system needs skills for identification of entities from descriptions. An attempt to map a description to an entity can have three different outcomes: a unique entity is found, the description is ambiguous and several objects are found, or the description is unsatisfiable and no entity is found.

Abella et al. [1] provide a set of dialogue principles for successful interaction with a dialogue system. These are completion, disambiguation, relaxation, confirmation, augmentation, and user confusion/error correction. Some of these overlap with Hayes & Reddy's skills, for example disambiguation is similar to the skills for dealing with descriptions.

Completion of a request means requesting missing pieces of information from the user. Disambiguation can be needed for two reasons, if the user has said something ambiguous, or if there are too many responses from the application. Relaxation and Confirmation are two ways to deal with requests that cannot be executed due to conflicting information. Relaxation means that the properties of a request that are considered the least important are dropped and the system tries to fulfil the new, less specific, request. Confirmation can be used by a system to check if information that seems incorrect has been understood correctly, for example if a user has provided a date that is out of the range of possible dates. Augmentation is the opposite of relaxation. If a request lacks some constraints, the system should choose the best question to ask in order to resolve the ambiguity. If a user does not know how to answer, a question or gives an erroneous answer the system must be able to find another question to ask. This is called User confusion/Error correction by Abella et al. [1].

Another set of guidelines aiming at the designing of co-operative systems for information-providing dialogues, is presented by Bernsen et al. [8]. The emphasis is on co-operativity and skills that minimise miscommunications. There are also some skills for the clarification and repair meta-communication necessary to handle misscommunications, as such cannot be totally eliminated in dialogue systems due to technical limitations.

Seven aspects of the interaction are captured by the guidelines: informativeness, truth and evidence, relevance, partner assymetry, background knowledge and repair and clarification. Each aspect is represented by one or several generic or specific guidelines, which include some of Grice's maxims [21].

The guidelines overlap the skills presented by Hayes & Reddy [23] and the principles of Abella et al. [1]; for example, to provide feedback, to deal with inconsistent or vague user input, to handle misunderstandings by the system or the user, and to provide sufficient domain knowledge and inference. However, the guidelines also cover some other aspects of the interaction, for example, communication of the commitments made by the user, to use the same formulation of a question everywhere, to communicate what the system can and cannot do, and to provide instructions on how the user should interact with the system. They also promote user-adaptive interaction; the system should differentiate between the needs of novice and expert users whenever possible.

Another aspect of the interaction between a user and a dialogue system which is not explicitly mentioned by the principles and guidelines is initiative. To allow mixed-initiative in dialogues is, however, considered a very important feature of dialogue systems by many. Initiative can be viewed in different ways (see [14] for some approaches), I use the term mixed-initiative to mean that both user and system can control the flow of the dialogue by introducing new topics and initiating clarification sub-dialogues. Skills related to mixed-initiative dialogue include how the user might answer a question from the system, for example by ignoring it or giving too much information, and how clarification sub-dialogues can be initiated.

3.2 Capabilities for dialogue management

The skills, principles and guidelines presented above, which are partially overlapping, deal with features that are considered useful in order for a dialogue system to interact with a user in a co-operative and natural way. In this section I will reformulate them in a uniform format and introduce a classification. When skills, principles and guidelines are similar, the most generic one has been chosen.

3.2.1 Handling tasks and requests

The communication between user and system always has a purpose, to achieve a task. The task can originate from the user or from the system, depending on the service type of the system. In a problem-solving system the user's task is in focus while an information-providing system concentrates on the information requests from the user and the corresponding system task. In an information providing system, a typical sub-task is to specify a parameter in a request. A request can be quite complex and consist of a number of constraints that the user has to provide. There are several capabilities useful for handling tasks and requests:

A1 To identify the task.

A2 To identify sub-tasks and know how they are related to a task.

A3 To reason about how much of a task has been achieved so far.

A4 To decide what action to take in order to achieve a task.

A5 To deal with situations in which no answer can be retrieved from the background system.

A6 To deal with situations in which the answer from the background system includes to much information.

A7 To detect and deal with hypothetical questions.

A8 To explicitly communicate a commitment made by the user in the conversation.

3.2.2 Achieving mixed-initiative dialogue

A key to achieving a natural interaction is to allow mixed-initiative dialogue in which both user and system can take the initiative. The mixed-initiative behaviour of a dialogue system depends on the following capabilities:

A9 To allow the user to over-answer questions.

A10 To allow the user to initiate clarification sub-dialogues.

A11 To allow the user to abandon the current request and pose a new request instead.

3.2.3 Handling focus and discourse

In order to achieve a natural dialogue, a system has to know what the conversation is about. This involves knowledge of what the focus of the current conversation is and other capabilities related to the discourse:

A12 To follow shifts in focus.

A13 To resolve anaphora and ellipsis.

A14 To answer questions on what has been said and done during the conversation

A15 To answer questions about the reason why an action was performed.

3.2.4 Handling domain knowledge

Other capabilities needed to achieve a natural dialogue are related to the system's knowledge about the domain. To be an intelligent conversational partner, the system has to have sufficient domain knowledge and be able to make inferences. The domain knowledge is necessary for the system to be able to handle descriptions and processing of requests:

A16 To map a description to an entity.

A17 To detect ambiguous descriptions and deal with them.

A18 To detect erroneous descriptions and deal with them.

A19 To know the type and structure of the entities in the domain.

A20 To reason about and derive new information from the information provided by the user.

A21 To deal with a user's erroneous inferences or false presuppositions

A22 To have domain-related default information.

A23 To adapt to the user's domain expertise.

A24 To know what the system can and cannot do.

4 Relations between capabilities and knowledge sources

The presented capabilities all depend on the use of various knowledge sources and models. For some of the capabilities knowledge from several knowledge sources or models is required while for others one knowledge source is enough. In this section I will explore how the different capabilities can be achieved. The systems presented in the section 2 will be used as a basis for the analysis.

4.1 Handling tasks and requests

To handle tasks and requests there are primarily four types of knowledge sources involved: dialogue models, dialogue histories system task models and user task models. Often these can be interchanged, for example, an elaborate system task model can take on the responsibilities usually held by a dialogue model or dialogue history.

A1 **To identify the task.** In systems that only perform a specific task, like the RAILTEL and WAXHOLM systems, this capability only requires a dialogue model that holds the information on how the system should behave in order to co-operate with the user to achieve the task at hand. In systems where the task is more complex or several tasks must be accomplished, explicit models of the system's or the user's tasks are used, as is the case in, for example, the SUNDIAL and the TRAINS systems, which utilise system task models and user task models respectively.

A2 **To identify sub-tasks and know how they are related to a task.** For this capability either a dialogue model, a system task model, or a user task model is needed. For the problem-solving systems, TRAINS and CIRCUIT FIX-IT SHOP, the user task model explicitly represents how sub-tasks are related to the user's task. In CIRCUIT FIX-IT SHOP a task that is not accomplished is represented by a theorem that is not yet proven, and missing axioms correspond to sub-tasks. In many of the information-providing systems, for example RAILTEL, SUNDIAL and GALAXYII, frames are used as system task models which implicitly model how sub-tasks are related to the system's task, for example, how specification of a departure location is related to the overall task of providing trip information. In information-providing systems where no explicit system task model is used, e.g. LINLIN, the relation between a sub-task and a task is represented by the dialogue model, for example by a sub-dialogue specification.

A3 **To reason about how much of a task has been achieved so far.** To accomplish this capability either a dialogue history, a system task model or a user task model can be used. The problem-solving systems, e.g. TRAINS and CIRCUIT FIX-IT SHOP, utilise user task models, while the information-providing systems that have system task models, e.g. RAILTEL and SUNDIAL, use these. Information-providing systems that lack explicit system task models, e.g. LINLIN, use a dialogue history instead, which represents what has been accomplished in the dialogue so far.

A4 **To decide what action to take in order to achieve a task.** In simple cases, e.g. systems such as LINLIN that only perform one task, a dialogue model is sufficient for this purpose. For systems that deal with a complex task or more than one task, explicit system task models or user task models are used together with a dialogue model, as in GALAXYII, SUNDIAL, TRAINS and CIRCUIT FIX-IT SHOP.

A5 **To deal with situations in which no answer can be retrieved from the background system.** This capability is primarily needed for information-providing systems where fully specified requests are used to retrieve an answer from some background system. A way to deal with this type of situation is to relax some of the constraints in the request. This approach is taken in, for example, SUNDIAL. Another approach is to present the problem to the user and let her deal with it. This requires a dialogue model that can handle this type of situation.

A6 **To deal with situations in which the answer from the background system includes too much information.** To achieve this capability knowledge from a domain knowledge source or conceptual model can be used to narrow the set of possible answers. The dialogue model can also provide information on how to correct the situation, primarily by entering a sub-dialogue. For instance, RAILTEL has three different dialogue grammar rules to deal with the latter type of situation: If there are less than 3 answers to a request, all information is presented; if there are 3 to 10 answers, partial information is presented and more specific information is asked for in order to narrow down the set; if there are more than 10 answers, more specific information is requested from the user [7].

A7 **To detect and deal with hypothetical questions.** None of the systems in the survey deals with hypothetical questions. To be able to do so the system should be able to differentiate between the task at hand and the new task that the hypothetical question involves. This requires a sophisticated dialogue model, and a dialogue history that records the current task which the system can switch back to later. A system task model or user task model might also be helpful to separate hypothetical from ordinary requests.

A8 **To explicitly communicate a commitment made by the user in the conversation.** This capability requires a dialogue model that can confirm the commitments made, and a system task model, user task model, or dialogue history that serve as a source of this type of information. For example, when the user has specified a trip, the provided constraints can be recorded in a task model or the dialogue history. Before accessing the background system the system can then communicate these commitments to the user by means of one of these models and the dialogue model.

4.2 Achieving mixed-initiative dialogue

Achieving mixed-initiative dialogue mostly rely on a flexible dialogue model and in some cases the existance of a dialogue history.

A9 To allow the user to over-answer questions. Over-answering a question from the system means that the user provides the requested information but also takes the initiative and provides new information. To handle this requires a flexible dialogue model which allows the user several ways to respond to a system utterance. A dialogue history, system task model, or user task model is also needed for storage of the provided information. The LINLIN system accomplish this capability with the use of the dialogue model and dialogue history, while, for example, the RAILTEL system utilises the dialogue model and system task model.

A10 To allow the user to initiate clarification sub-dialogues. This capability requires both a flexible dialogue model and a dialogue history. If the user is to be able to postpone the answer to a question from the system, the system must remember previous states of the dialogue so that it can pick them up later in the interaction.

A11 To allow the user to abandon the current request and pose a new request instead. This is the third capability related to mixed-initiative dialogue, and as previously (A9-A10) this also requires a flexible dialogue model. The user must be allowed to shift topic and sometimes ignore a direct question from the system.

4.3 Handling focus and discourse

The primary type of knowledge source needed to handle focus and discourse is dialogue histories, dialogue models, system task models and user task models can however also be needed for some capabilities.

A12 To follow shifts in focus. Shifts in focus can be detected by the use of a dialogue model. In many systems, for example CIRCUIT FIX-IT SHOP, VERBMOBIL and SUNDIAL, the dialogue model supports the computation of expected types of responses, which can be used to determine whether a shift in focus has occurred. To know what the previous and current foci are also requires a dialogue history that models the attentional state of the dialogue. Such dialogue histories can be found in the GALAXYII and RAILTEL systems, for example.

A13 To resolve anaphora and ellipsis. To resolve anaphora and ellipsis a dialogue history that models the entities that have been talked about is required in order to find the possible referents. For example, a sequence of previous frames as in the GALAXYII and WAXHOLM systems, or a more structured representation like the trees in the SUNDIAL, RAILTEL and LINLIN systems, can be used for this purpose.

A14 To answer questions on what has been said and done during the conversation. For this capability a dialogue history that records the previous states of the dialogue is essential together with a dialogue model that can handle this type of question. User task models of the type in the CIRCUIT FIX-IT SHOP system can also provide information on what has been done.

A15 To answer questions about the reason why an action was performed. If the system is to be able to answer why an action was performed, it has to have a sophisticated model of the tasks that include why they were performed. User task models of the type found in the TRAINS and CIRCUIT FIX-IT SHOP systems could be used for this purpose. Of course, the dialogue model must also know how to deal with this type of question.

4.4 Handling domain knowledge

Handling of domain knowledge mostly rely on the use of domain knowledge sources and conceptual models. The other types of knowledge sources and models can however be useful for some of the capabilities.

A16 To map a description to an entity. Since background and application systems often contain quantitative and precise data, vague qualitative terms have to be transformed to precise entities, for example the expression "this evening" can be transformed to a precise date and time interval. This can be done with a domain knowledge source, for example by the use of interpretative rules like those in the RAILTEL system.

A17 To detect ambiguous descriptions and deal with them. Descriptions can be ambiguous, meaning that there are several matching entities. Domain knowledge sources or conceptual model can be used both to decide if descriptions are ambiguous and to determine how the situation should be handled, for example by providing alternatives or asking for clarifying information. The latter strategy, of course, has to be supported by the dialogue model.

A18 To detect erroneous descriptions and deal with them. Descriptions can also be erroneous if they hold faulty or inconsistent information. In this case a domain knowledge source or conceptual model can be used to find the problem and sometimes suggest other possible interpretations that can be presented to and evaluated by the user.

A19 To know the type and structure of the entities in the domain. A conceptual model is suitable for this. In the CIRCUIT FIX-IT SHOP system a specific model that is separated from the rest of the task and domain knowledge is used for this purpose. A domain knowledge source can also hold this type of information.

A20 **To reason about and derive new information from the information provided by the user.** This capability can be achieved by utilising a domain knowledge source or a conceptual model, given that they are capable of reasoning about entities and concepts. For example, in the VERBMOBIL system the conceptual hierarchy of temporal entities is used to reason about and derive available times for scheduling.

A21 **To deal with a user's erroneous inferences or false presuppositions.** This capability requires a domain knowledge source that can reason about information provided by the user and spot erroneous inferences and false presuppositions, as well as a dialogue model that can give appropriate responses in these situations. A user model of the type found in TRAINS that represents the user's beliefs can also be useful in detecting problems of this type.

A22 **To have domain-related default information.** The domain knowledge sources can incorporate reasoning mechanisms to provide default information. Default values can also be incorporated in a system task model. For example, in RAILTEL default value rules that are linked to the semantic frame that represents a request are used for this purpose.

A23 **To adapt to the user's domain expertise.** In the CIRCUIT FIX-IT SHOP and TRAINS systems the user model holds the information needed to achieve this ability. In TRAINS the user model represents the beliefs held by the user. This information is utilised to decide what new information should be brought to the user's attention.

A24 **To know what the system can and cannot do.** For this capability the system must have meta-knowledge about the tasks and domain it covers. This knowledge could be part of, or be derivable from, a domain knowledge source.

4.5 Summary

An overview of the relations between knowledge sources and capabilities is given in table 1. Knowledge sources that are Required to achieve a specific capability are marked by an R and L means at Least one of the knowledge sources is required.

5 Conclusions and discussion

In this paper knowledge sources and models used for dialogue management in dialogue systems, and the capabilities they support have been described and discussed.

In a survey of eight information-providing or problem-solving dialogue systems four types of knowledge, represented by seven different knowledge sources, were identified. **Dialogue knowledge** is represented in *dialogue models* and *dialogue histories*. **Task knowledge** can be divided in *system task models* and *user task models*. **Domain knowledge** is held by *domain knowledge sources* and *conceptual models*. Finally, **Knowledge of the user**

Table 1: An overview of the relations between capabilities and knowledge sources and models. R stands for required and L for at Least one of.

Model/ Capability	**Dial Mod**	**Dial Hist**	**Sys Task Mod**	**User Task Mod**	**Dom Mod**	**Conc Mod**	**User Mod**
Task and requests							
A1	L		L	L			
A2	L		L	L			
A3		L	L	L			
A4	L		L	L			
A5	L		L		L	L	
A6	L				L	L	
A7	R	R	L	L			
A8	R	L	L	L			
Mixed-initiative dialogue							
A9	R	L	L	L			
A10	R	R					
A11	R						
Focus and discourse							
A12	R	R					
A13		R					
A14	R	L	L	L			
A15	R	L		L			
Domain knowledge							
A16					R		
A17	R				L	L	
A18	R				L	L	
A19					L	L	
A20					L	L	
A21	R	L		L	L		
A22			L		L		
A23							R
A24					L		

can be represented by a *user model*. With this new categorisation the sometimes confusing terminology concerning knowledge sources and models in dialogue systems is hopefully made more clear.

A list of capabilities considered useful for achieving natural and graceful interaction was compiled from a set of guidelines and development principles for dialogue system. The relation between the knowledge sources and the capabilities were mapped out and summarised in table 1. This type of information can be used to support design of dialogue systems. The design of a dialogue system should be based on an analysis of the system's intended functionality and behaviour. This type of characterisation can be based on studies of corpora or other empirical studies which result in requirements that can be expressed in terms of the capabilities described in this section. Given a set of desirable capabilities, the minimum set of required knowledge sources and models can be derived with the help of table 1.

It is, however, important to realise that the set of models that is most suitable for some capabilities might not be sufficient to support implementation of other capabilities. The decision as to which models to use in a dialogue system must therefore be guided by the situation the system will be used in, but it is important that it is an explicit choice and that the restrictions it imposes are considered. By introducing more knowledge sources and models a more flexible dialogue system that can more easily be extended with new capabilities may be designed. It may also facilitate modularisation and clarity of the knowledge sources and models' responsibilities.

Another way to use the capabilities, besides the design of dialogue systems, is to use them is for evaluation of dialogue systems. An approach to evaluation of dialogue systems proposed in the TRINDI project [39] is to examine whether certain characteristic behaviours are manifested by the system. A "Tick-list" consisting of twelve yes-no questions is presented and suggested for this purpose [15]. Some of these questions correspond to the capabilities, for example "Can the system deal with answers to questions that give more information than was requested?" corresponds to capability A9, "To allow the user to over-answer questions" and "Can the system deal with ambiguous designators?" corresponds to capability A17, "To detect ambiguous descriptions and deal with them". Another use of the capabilities, besides design of dialogue systems, could therefore be to create "tick-lists" that can be used for evaluation and comparison of dialogue systems and frameworks.

Applying the capabilities to evaluation of dialogue system is one branch of future work, another is to extend the list and include other types of dialogue systems. The guidelines and development principles used to create the list of capabilities are primarily aimed at information-providing dialogue systems. To make the list complete more work has to be done. For this purpose more guidelines and development principles can be examined, and studies of corpora will also be useful.

6 Acknowledgments

This work is supported by The Swedish Transport & Communications Research Board (KFB). Thanks to Nils Dahlbäck, Lars Degerstedt and Arne Jönsson for stimulating discussions and comments on previous versions of this paper. Thanks also to Ingrid Zukerman and Richard McConachy for the interaction at the ETAI area "Intelligent User Interfaces".

References

[1] Alicia Abella, Michael K. Brown, and Bruce Buntschuh. Development principles for dialog-based interfaces. In Elisabeth Maier, Marion Mast, and Susann Luperfoy, editors, *Dialogue Processing in Spoken Language Systems*, volume 1236 of *Lecture Notes in Computer Science*, pages 141–155. Springer-Verlag, 1997.

[2] Lars Ahrenberg, Arne Jönsson, and Nils Dahlbäck. Discourse representation and discourse management for natural language interfaces. In *Proceedings of the Second Nordic Conference on Text Comprehension in Man and Machine, Täby, Sweden*, 1990.

[3] Jan Alexandersson. Plan recognition in VERBMOBIL. Technical Report 81, DFKI GmbH, URL:
http://www.dfki.uni-sb.de/cgi-bin/verbmobil/htbin/doc-access.cgi, 1995.

[4] Jan Alexandersson, Elisabeth Maier, and Norbert Reithinger. A robust and efficient three-layered dialogue component for speech-to-speech

translation system. Technical Report 50, DFKI GmbH, URL:
http://www.dfki.uni-sb.de/cgi-bin/verbmobil/htbin/doc-access.cgi, 1994.

[5] Jan Alexandersson, Norbert Reithinger, and Elisabeth Maier. Insights into the dialogue processing of VERBMOBIL. Technical Report 191, DFKI GmbH, 1997.

[6] James Allen, Lenhart Schubert, George Ferguson, Peter Heeman, Chung He Hwang, Tsuneaki Kato, Mark Light, Nathaniel Martin, Bradford Miller, Massimo Poesio, and David Traum. The TRAINS project: a case study in building a conversational planning agent. *Journal of Experimental and Theoretical Artificial Intelligence*, 7:7–48, 1995.

[7] S. Bennacef, L. Devillers, S. Rosset, and L. Lamel. Dialog in the RAILTEL telephone-based system. In *Proceedings of International Conference on Spoken Language Processing, ICSLP'96*, volume 1, pages 550–553, Philadelphia, USA, October 1996.

[8] Niels Ole Bernsen, Hans Dybkaer, and Laila Dybkaer. *Designing Interactive Speech Systems: From First Ideas to User Testing.* Springer Verlag, 1998.

[9] Eric Bilange. A task independent oral dialogue model. In *Proceedings of the Fifth Conference of the European Chapter of the Association for Computational Linguistics, EACL'91*, pages 83–88, Berlin, Germany, 1991.

[10] Rolf Carlson and Sheri Hunnicut. Generic and domain-specific aspects of the Waxholm NLP and dialog modules. In *Proceedings of International Conference on Spoken Language Processing, ICSLP'96*, volume 2, pages 677–680, Philadelphia, USA, October 1996.

[11] Rolf Carlson, Sheri Hunnicut, and Joakim Gustafsson. Dialog management in the Waxholm system. In *Papers from the Eighth Swedish Phonetics Conference, Working Papers 43*, pages 46–49, 1994.

[12] Philip. R. Cohen and C. Raymond Perrault. Elements of a plan-based theory of speech acts. *Cognitive Science*, 3:177–212, 1979.

[13] Robin Cohen. On the relationship between user models and discourse models. *Computational Linguistics*, 14(3):88–90, 1988.

[14] Robin Cohen, Coralee Allaby, Christian Cumbaa, Mark Fitzgerald, Konsin Ho, Bowen Hui, Celine Latulipe, Fletcher Lu, Nancy Moussa, David Pooley, Alex Qian, and Saheem Siddiqi. What is initiative. *User Modeling and User-adapted Interaction*, 8(3–4):5–48, 1998.

[15] Robin Cooper, Staffan Larsson, C. Matheson, and David Traum. Coding instructional dialogue for information state. Technical Report D1.1, TRINDI, URL:
http://www.ling.gu.se/research/projects/trindi/publications.html, 2000.

[16] Nils Dahlbäck. Towards a dialogue taxonomy. In Elisabeth Maier, Marion Mast, and Susann LuperFoy, editors, *Dialogue Processing in Spoken Language Systems*, number 1236 in LNAI-Lecture Notes in Artificial Intelligence. Springer Verlag, 1997.

[17] Nils Dahlbäck and Arne Jönsson. Integrating domain specific focusing in dialogue models. In *Proceedings of Eurospeech'97*, volume 4, pages 2215–2218, Rhodes, Greece, 1997.

[18] George Ferguson, James Allen, and Brad Miller. TRAINS-95: Towards a mixed-initiative planning assistant. In *Proceedings of the Third Conference on Artificial Intelligence Planning Systems, AIPS-96*, pages 70–77, 1996.

[19] George M. Ferguson, James F. Allen, Brad W. Miller, and Eric K. Ringger. The design and implementation of the TRAINS-96 system: A prototype mixed-initiative planning assistant. TRAINS Technical Not 96-5, October 1996.

[20] David Goddeau, Eric Brill, James Glass, Christine Pao, Michael Philips, Joseph Polifroni, Stephanie Seneff, and Victor Zue. GALAXY: A human-language interface to on-line travel information. In *Proceedings of International Conference on Spoken Language Processing, ICSLP'94*, pages 707–710, Yokohama, Japan, September 1994.

[21] Paul H. Grice. Logic and conversation. In Peter Cole and Jerry L. Morgan, editors, *Syntax and Semantics (vol. 3) Speech Acts*. Academic Press, 1975.

[22] Barbara J. Grosz and Candace L. Sidner. Attention, intention and the structure of discourse. *Computational Linguistics*, 12(3):175–204, 1986.

[23] Philip J. Hayes and D. Raj Reddy. Steps toward graceful interaction in spoken and written man-machine communication. *International Journal of Man-Machine Studies*, 19:231–284, 1983.

[24] Paul Heisterkamp, Scott McGlashan, and Nick Youd. Dialogue semantics for a spoken dialogue system. In *Proceedings of the International Conference on Spoken Language Processing, ICSLP'92*, Banff, Canada, 1992.

[25] Arne Jönsson. A model for habitable and efficient dialogue management for natural language interaction. *Natural Language Engineering*, 3(2/3):103–122, 1997.

[26] Robert Kass and Tim Finin. Modeling the user in natural language systems. *Computational Linguistics*, 14(3):5–22, 1988.

[27] Alfred Kobsa. User models and discourse models united they stand... *Computational Linguistics*, 14(3):91–94, 1988.

[28] Scott McGlashan, Norman Fraser, Nigel Gilbert, Eric Bilange, Paul Heisterkamp, and Nick Youd. Dialogue management for telephone information systems. In *Proceedings of the International Conference on Applied Language Processing, ICSLP'92*, Trento, Italy, 1992.

[29] Michael McTear. Spoken dialogue technology: Enabling the conversational user interface. URL: http://www.infj.ulst.ac.uk/~cbdg23/survey/spoken_dialogue_technology.html, 2000.

[30] Joachim Quantz, Manfred Gehrke, Uwe Kssner, and Birte Schmitz. The VERBMOBIL domain model version 1.0. Technical Report 29, Technishe Universität Berlin, September 1994.

[31] Elaine Rich. User modelling via stereotypes. In *Readings in Intelligent User Interfaces*, pages 329–341. Morgan Kaufmann, 1998.

[32] Emanuel A. Schegloff and Harvey Sacks. Opening up closings. *Semiotica*, 7:289–327, 1973.

[33] Ethel Schuster. Establishing the relationship between discourse models and user models. *Computational Linguistics*, 14(3):82–85, 1988.

[34] Stephanie Seneff, David Goddeau, Christine Pao, and Joseph Polifroni. Multimodal discourse modelling in a multi-user multi-domain environment. In *Proceedings of International Conference on Spoken Language Processing, ICSLP'96*, pages 192–195, Philadelphia, USA, October 1996.

[35] Stephanie Seneff, Ed Hurley, Raymond Lau, Christine Pao, Philipp Schmid, and Victor Zue. GALAXYII: A reference architecture for conversational system development. In *Proceedings of International Conference on Spoken Language Processing, ICSLP'98*, volume 3, pages 931–934, Sydney, Australia, December 1998.

[36] R. W. Smith and D. R. Hipp. *Spoken Natural Language Dialog Systems: A Practical Approach.* New York: Oxford University Press, 1994.

[37] Ronnie W. Smith. Integration of domain problem solving with natural language dialog: The missing axiom theory. In *Proceedings of Applications of AI X: Knowledge-Based Systems*, pages 270–278, 1992.

[38] David R. Traum. Conversational agency: The TRAINS-93 dialogue manager. In Susann LuperFoy, Anton Nijhholt, and Gert Veldhuijzen van Zanten, editors, *Proceedings of Twente Workshop on Language Technology, TWLT-II*, 1996.

[39] The TRINDI project website. URL: http://www.ling.gu.se/research/projects/trindi/, 2001.

[40] UMBC KQML Web. URL: http://www.cs.umbc.edu/kqml/, 2000.