Building Usable Spoken Dialogue Systems Some Approaches

Niels Ole Bernsen and Laila Dybkjær

Natural Interactive Systems Laboratory, University of Southern Denmark

1 Introduction

From a general HCI (Human-Computer Interaction) point of view, spoken dialogue systems (SDSs) are a kind of speech-based interfaces. Clearly, speech may be used for interface purposes other than spoken dialogue, such as for output-only, as in text-to-speech systems, spoken alarms, spoken directions, and the like, or for input-only, as in systems in which spoken input commands are used to control various kinds of graphical output, such as graphical menus, text time-tables, or other speech-generated search results. However, if the interface is both interactive and really is speech-*based*, then spoken dialogue is a key component in user-system interaction. Even then, spoken dialogue does not have to be the only component. Speech input/output can often be usefully combined with other interactive modalities, producing what might be termed multimodal spoken dialogue interfaces.

The bulk of the present paper presents three approaches to making SDSs usable. The paper does *not* aim to provide a complete overview of the complex field of SDS usability. We believe that the issue of SDS usability today is a target moving as fast as ever, due to the rapid expansion of the field. Rather, the paper reviews some potentially useful results to SDS usability based on work by the authors and colleagues, and looks at some of the challenges ahead. In this context, an *approach* means a way of supporting the building of usable SDSs, a way which is systematic and either based on theory or on substantial empirical research. In HCI, classical *task analysis* is a well-known example of an approach in this sense. Task analysis for SDS development will not be discussed below even though task analysis is essential to the development of usable SDSs and, arguably, its focused application to SDSs development does not seem to have been described in depth in the literature at this point.

Before presenting three approaches to SDS usability, we provide some brief state of the art background in order to establish a frame of reference for discussing the rapidly growing number of usability issues facing SDS developers. Some still believe that speech-based interfaces are mainly used for simple command-and-control applications and that the term 'SDS' is merely a euphemism for those. It is hard to be more wrong that that. In what follows, Section 2 describes the currently dominant SDS application paradigm, i.e. the task-oriented SDS. Section 3 widens the perspective rather dramatically by describing a series of emerging, non-task-oriented aspects of SDSs for which focused usability approaches are mostly lacking at this point. Section 4 presents what might perhaps be claimed to be a "baseline" for the usability evaluation of, primarily, task-oriented SDSs. From an HCI point of view, this baseline could be viewed as a synthesis of classical (but far too general for SDSs) HCI usability theory (or theories) and SDS-specific usability theory and know-how. Section 5 reviews results on when (not) to use speech in human-computer interaction. In a similar way, Section 6 reviews guidelines for how to develop cooperative spoken dialogue interfaces. Section 7 concludes the paper by (re-) emphasising how much we still need to learn.

2 The emergence of the task-oriented SDS paradigm

2.1 A brief history

An SDS is a computer system with which users can have spoken dialogue. Thus, to be successful, an SDS must understand spoken input, produce contextually appropriate spoken output in return, and generate some significant amount of user satisfaction without which user uptake of the technology is

likely to be seriously hampered. Simple as this may appear, from a system development point of view, spoken dialogue is not just spoken dialogue but a dynamic, multi-dimensional space of established achievements, commercial system solutions, research prototype endeavours, and remaining challenges. Probably, this remark has been valid since the first research prototype SDSs were developed in the 1980s, such as APHODEX [Haton 1988] and EVAR [Niemann et al. 1988], and certainly since the first commercial SDS appeared. This was in 1989 when Bell Northern Research deployed "Automated Alternate Billing Services" through local telephone companies in the USA. The system rang customers, told them they had a collect call, and asked whether they would accept the call. Using a very small vocabulary (yes/no and some synonyms), the system successfully completed about 95% of the calls that were candidates for automation [Bossemeyer and Schwab 1991].

In the 1990s, the kind of SDS just exemplified, i.e. the telephone-based, task-oriented SDS, became established as the field's first successful application paradigm [Bernsen et al. 1998]. Today, such systems abound, enabling users to accomplish tasks of many different kinds and in different languages, such as train time-table information systems, switchboard systems, address and phone number information systems, banking systems, polling systems, and frequently asked questions systems. The drivers of these developments were research projects, such as US DARPA ATIS on air travel information [DARPA 1992], EU Sundial on flight and train time-table information [Peckham 1993], the Danish Dialogue Project [Baekgaard et al. 1995] on domestic flight ticket reservation, and US DARPA Communicator on travel planning systems (flight and hotel reservation, car rental) [Walker et al. 2002].

2.2 What's inside?

Today, the task-oriented SDS, whether telephone-based or accessed through an open microphone, is the all-dominant paradigm among commercial SDSs and still, to a very large extent, in research as well. Let us briefly look at some SDS components.

An advanced commercial spoken dialogue system includes the following main components: (1) a *speech recogniser* which converts the acoustic speech signal to an N-best list of text strings or a richer and more compact Word Hypothesis Graph, reflecting the recogniser's best hypotheses about what the user actually said; (2) a *natural language understanding* module which analyses the output from the speech recogniser and selects the best semantic representation of the user's input; (3) a *dialogue manager* which interprets the input in the discourse context, often looks up task-relevant information in one or several external information stores, typically a database, and decides on the next spoken output; and (5) a *speech synthesiser* which converts the selected output text string to spoken language output. In-between (3) and (5) there may be (4) a *response generator* which converts the dialogue manager's semantic output to a surface language text string. Whilst (2) and (4) are still rather primitive, natural language understanding components (2), in particular, demand rapid progress. This is due to the fact that, increasingly, commercial SDSs are moving away from accepting command input-only and towards accepting spontaneous (free-form) spoken language input, liberating users from having to memorise, or tediously repeat, ever-increasing numbers of input commands. Speech recognition (1) remains a potential weakness of SDSs, and huge efforts continue to be invested in improving speaker-independent speech recognition technologies. Speech synthesis technologies (5) continue to slowly improve, keeping up with the pace of progress in the other key SDS technologies. For some years, perhaps the dominant trend in dialogue managers (3) has been towards increased modularity in order to establish clean separation between task-dependent and task-independent dialogue management, facilitating portability to new system tasks.

3 The variety of spoken dialogue systems

Whilst the task-oriented SDS application paradigm was becoming entrenched during the 1990s, researchers were discovering that the full potential of SDSs stretches far beyond the common task-oriented SDS. As a result, the multi-dimensionality of the SDS problem space has never been more apparent than today. To identify some of the key dimensions, it may be useful to take a basic version of the task-oriented SDS described in Section 2 as our model. We then conceptually "grow" the model

in order to generate the far larger space confronting SDS researchers today. More explicitly, the basic model defines a *task-oriented, single-task, single-user, all-users-are-equal, speech-only, fixed-telephone-based, information system SDS.* Made more explicit like this, the model generates, through completion, the following modifications or aspects of SDSs in conceptual space:

1. non-task-oriented SDSs;
2. multiple-task SDSs;
3. multi-speaker SDSs;
4. on-line user modelling SDSs;
5. multi-modal (non-speech-only) SDSs;
6. ubiquitous SDSs;
7. non-information system SDSs.

This is an impressive list which includes some very large-scale challenges, such as the merely negatively defined (1), (5) and (7), and only a single challenge of comparatively modest proportions, i.e. (2). Moreover, since the SDS aspects listed are mostly orthogonal, except for (1) and (2), and hence may be combined at will, the combinatorial possibilities are quite significant. To be sure, many of the SDSs in this new combinatorial space need not be more complex, in technical terms, than complex task-oriented systems, although many other systems will be. The point, rather, is that, broadly speaking, we know far less about the usability of SDSs with some or all of the properties listed above than about the usability of basic task-oriented SDSs. The aspects may be briefly described as follows.

(1) *Non-task-oriented SDSs* include all SDSs whose purpose is *not* to enable users to complete (a) specific task(s). We also call these systems *domain-oriented* systems. At the limit, these systems include systems able to pass the classical Turing test [Turing 1950]. For the developer, the primary characteristic of these systems is that it is no longer possible to use the powerful constraints provided by the task in their development. For this reason, domain-oriented systems are only beginning to be explored today. An example is the NICE system (Natural Interactive Communication for Edutainment, http://www.niceproject.com/) which will enable conversation with fairy-tale author Hans Christian Andersen [Bernsen 2003a].

(2) *Multiple-task SDSs* are SDSs which help users solve more than a single task. Such systems already exist, for instance for checking both email and calendar over the phone. However, when the tasks are mutually *interdependent*, so that, for instance, one task may be interrupted in order to do another, and then resumed, or may be interrupted by output belonging to an already accomplished task, a new state of the art challenge arises. We have addressed this problem in a car information system in which users could, e.g., interrupt and later resume a complex hotel reservation task to negotiate navigation to the nearest petrol station [Charfuelán and Bernsen 2003].

(3) *Multi-speaker SDSs* represent another recent challenge. An influential current scenario comes close to addressing the challenge. The scenario is that of the smart-room automated meeting secretary which keeps track of the meeting agenda, recognises, understands, and transcribes all individual meeting contributions, summarises these, summarises the meeting as a whole, lists action points, etc. Often, computer vision is planned to be added to the smart-room set-up to assist in solving these tasks. Whilst the secretary does not quite need to be an SDS, the example suggests how the sky is the limit in developing future collaborative systems involving SDSs. Challenges include, e.g., speaker identification and speaker separation in cases of overlapping speech.

(4) *On-line user modelling SDSs* are able to adapt their interactive behaviour to a particular user or to users belonging to a certain user group based on observation of the user's interactive behaviour. This is another emerging research area. As part of the system referred to in (2) above, we built one of the few existing examples of an SDS which creates, maintains, and uses models of its individual users in order to facilitate the complex hotel selection and reservation task [Bernsen 2003b].

(5) *Multi-modal (non-speech-only) SDSs* constitute a huge field of research and application potential. An early example of a multimodal SDSs is the Swedish Waxholm information system [Bertenstam et al. 1995]. Spoken dialogue was used to bring up Stockholm archipelago boat time-tables and other information on the screen which also included a talking face. Since Waxholm, many research projects have begun to nibble into the challenges posed by multimodal SDSs, including large projects, such as German SmartKom 1998-2003 [http://www.smartkom.org/] which investigated task-oriented spoken

dialogue in combination with animated interface agent output and camera-captured gesture input. See also Section 5.

(6) *Ubiquitous SDSs* comprise, roughly, mobile SDSs embedded in all manner of portable and otherwise mobile systems, including, e.g., mobile phones and cars, as well as (relatively) stationary ambient intelligence applications which may be embedded anywhere, including in the refrigerator and the VCR.

(7) *Non-information system SDSs* are all SDSs which do not serve purposes of information-seeking and -provision. Educational and tutorial SDSs have been investigated for some time already but these systems might be viewed as a particular pedagogical variety of information systems, see, e.g., [Cassell et al. 2000]. Clearly, however, entertainment SDSs are not information systems at all. SDSs for edutainment and entertainment are only beginning to be explored, one example being the NICE system mentioned under (1) above.

4 Usability of task-oriented SDSs

Today, arguably, we have a pretty strong state-of-the-art baseline for building usable task-oriented SDSs. This baseline is composed of (i) general HCI best practices, such as: start doing usability evaluation as early as possible during development; do your domain and task analysis properly and in depth; know the users; and know the application environment; (ii) methods for usability which are particularly helpful in SDS development, such as the Wizard of Oz simulation method in which an SDS is simulated by one or more humans to users who are made to believe that they are communicating with a real system [Bernsen et al. 1998]; and (iii) substantial work on usability evaluation criteria for task-oriented SDSs. This latter work is reviewed in this section followed by a short discussion of some remaining challenges.

Based on [Bernsen et al. 1998] and comprehensive empirical investigation of a series of task-oriented SDSs in the EU DISC project (1997-1999, www.disc2.dk), [Dybkjær and Bernsen 2000] propose a set of 14 objective (quantitative or qualitative) and subjective usability evaluation criteria. To some modest extent, the criteria are specialised to the, in general, at least, harder case of walk-up-and-use SDSs. This is in view of the well-known fact that routine users may adapt to interfaces which are sub-optimal from the point of view of walk-up-and-use users. Still, even the latter users may use the system regularly and developers should take this into account.

The 14 criteria are: modality appropriateness; input recognition accuracy; coverage of user vocabulary and grammar; output voice quality; output phrasing adequacy; feedback adequacy; adequacy of dialogue initiative relative to the task(s); naturalness of the dialogue structure relative to the task(s); sufficiency of task and domain coverage; sufficiency of reasoning capabilities; sufficiency of interaction guidance; error handling adequacy; sufficiency of adaptation to user differences; and user satisfaction. Each criterion is explained below.

1) *Modality appropriateness.* Before making the decision to use speech, dialogue designers should make sure that spoken input and output, possibly combined with other input/output modalities, is an appropriate modality choice for the planned application. As speech-based interfaces have begun to be combined with other input and/or output modalities, this issue has increased in importance. Applied theoretical results on how to do this are presented in Section 5.
2) *Input recognition accuracy.* Good recogniser quality is a key factor in making users confident that the system will successfully get what they say. Noisy environments and large vocabularies, say, +5000 words, still pose significant problems for achieving high-accuracy speaker-independent recognition. Basic recognition accuracy is relatively easy to measure and quantify. However, the information-rich speech signal continues to pose research challenges, such as speaker identification, speaker separation, or prosody recognition, solutions to which would greatly benefit the usability of SDSs. A broad overview of the state of the art can be obtained from the 2003 Eurospeech Proceedings [Bourlard 2003].
3) *Adequate coverage of user vocabulary and grammar.* Speaking to an SDS should be as easy and natural as possible. Even if the system's speech recognition is perfect in principle, users will still have problems being understood if the system's input vocabulary and grammar are not the ones

which users are likely to use for the task. Moreover, what users experience as natural input speech is highly relative to the system's output phrasing, cf. (5) below. Thus, the system's output language should be used to control-through-priming users' input language to help the latter become manageable for the system whilst still feeling natural to users. The state of the art in quantifying the quality of grammars and lexicons for SDS natural language understanding is rather advanced today. However, theory in support of building "the right" real-time robust natural language understanding technology for ever-changing SDS application types is still quite weak and characterised by experimental trial-and-error. People do not speak the same way as they write, and complex syntactic parsing based on advanced grammar formalisms often fail on the real-time requirement to SDSs.

4) *Output voice quality.* From the user's point of view, good SDS output voice quality means that the system's speech is clear and intelligible, does not demand additional listening effort, is not particularly noise-sensitive or distorted by extraneous sounds, has natural intonation and prosody, uses an appropriate speaking rate, and is pleasant to listen to [Karlsson 1999]. Good progress has been made in recent years but taken together, these requirements are still impossible to meet no matter which speech synthesis technology one chooses. The application-induced pressure for getting better speech synthesis seems to be growing due to, for instance, the need for expressing the personality and emotions of different animated characters. Voice quality evaluation continues to include an important element of subjective evaluation.
5) *Output phrasing adequacy.* As noted under (3) above, the system's output lexicon and grammar strongly co-determine how users speak to the system. When used smartly, this can help improve the success of user-system communication. Moreover, the *contents* of an SDS's spoken output is an key factor in determining the system's usability. Applied theoretical results on how to ensure adequate output phrasing are presented in Section 6.
6) *Feedback adequacy.* This point could, in fact, be viewed as part of (5) above. The user must feel confident that the system has understood the information input in the way it was intended, and the user must be told which actions the system has taken and possibly what the system is currently doing. While these requirements may seem obvious to GUI (graphical user interface) designers, they pose very different problems in SDSs because of the nature of the acoustic modality in general and speech in particular, cf. also Section 5. Evaluation of feedback adequacy is a qualitative measure based on empirical data.
7) *Adequacy of dialogue initiative relative to the task(s).* To support natural interaction, an SDS needs a reasonable choice of dialogue initiative, depending on factors such as the nature of the task, users' background knowledge, and frequency of use. Initiative adequacy must be evaluated relatively to these factors. For instance, if routine users of the SDS know exactly what the system can and cannot do and how to interact with it, the system may hardly need to take the initiative at all during dialogue. With novice users, as in walk-up-and-use systems, system-driven dialogue may be preferable and also experienced as natural by users. It is perfectly possible today to develop mixed-initiative dialogue for all or most information tasks, but this is not always an optimal solution and it does impose more complex dialogue management, including error handling, cf. (12).
8) *Naturalness of the dialogue structure relative to the task(s).* Tasks have more or less inherent structure. For instance, most users will know intuitively that it makes little sense to ask for departure time before they have told the system from where they want to travel and where they want to go. A frequently asked questions list, on the other hand, has little inherent structure. Depending on task structure and complexity, dialogue designers may have to impose some amount of additional structure onto the dialogue, determining which topics (or sub-tasks) could be addressed when. It is important that the structure imposed on the dialogue is as natural to the user as possible, reflecting the user's intuitive expectations or, at least, not contradicting them.
9) *Sufficiency of task and domain coverage.* Even if unfamiliar with SDSs, users often have rather detailed expectations to the information or service which they should be able to obtain from the system. It is essential that the system meets these expectations to the extent possible. If, for some reason, it does not fully meet them, the user must be informed somehow. This work requires task and domain expertise, task analysis, and interaction data analysis.

10) *Sufficiency of reasoning capabilities.* Contextually adequate reasoning is a standard issue in the design of natural interaction. SDSs must incorporate both facts and inferences about the task as well as general world knowledge in order to act as adequate interlocutors. A task does not have to be very complex before primitive solutions, such as command keywords or spoken menus from which the user can choose, become unwieldy. In such cases, the system must be designed to understand and process what users actually say rather than what the developers would have liked them to say. This may require substantial system reasoning. For example, reasoning about dates may be quite complex if the system is to understand, in general, the meaning of utterances such as "We will depart three days later, that is, right after the weekend".
11) *Sufficiency of interaction guidance.* Users should feel in control throughout interaction. Useful help mechanisms may be an explicit or implicit part of the spoken dialogue; be available on request by saying "help"; or be automatically enabled if the user is having problems repeatedly, for instance in being recognised. Arguably, GUI designers have never solved the contextual help problem of providing only the exact help needed when it is needed. The temporal transience of speech means that (unimodal) SDSs cannot offer a static user interface during interaction. Rather, novice users have to be told what the system can and cannot do and how to interact with it. Even for only modestly complex tasks, this cannot be done explicitly, which is why factors, such as output phrasing adequacy (5), feedback adequacy (6), natural dialogue initiative (7), natural dialogue structure (8), and intuitive task coverage (9) are so important.
12) *Error handling adequacy.* This issue may be decomposed along two dimensions. Either the system initiates error handling meta-communication or the user initiates error-handling meta-communication. When error-handling meta-communication is initiated, it is either because one party has failed to hear or understand the other or because what was heard or understood is false, or it is because what was heard or understood is somehow in need of clarification. As a general rule, error correction meta-communication is easier to deal with in SDSs than clarification meta-communication [Bernsen et al. 1998]. However, the field of research into on-line error handling in SDSs is large and growing, and it is not possible to do it justice in the present paper.
13) *Sufficiency of adaptation to user differences.* It seems useful to distinguish between system expert/domain expert, system expert/domain novice, system novice/domain expert and system novice/domain novice users. A particular SDS needs not support all four groups. Until very recently, in the context of SDSs, the all-dominant interpretation of "adaptation" was design-time adaptation. For instance, dialogue structure, feedback mechanisms, or dialogue initiative was determined at design-time for a particular target user group and only very modest on-line adaptation mechanisms were included, such as the option to skip the system's introduction to itself, or barge-in which enables the user to interrupt the system before it has finished its spoken output. On-line user modelling for SDSs offer entirely new opportunities for adaptation to user differences, cf. Section 3.
14) *User satisfaction.* This evaluation criterion is standardly applied through the use of questionnaires and interviews, yielding subjective evaluation measures. Questionnaire design still lacks established guidelines, and questionnaire results remain hard to interpret. The PARADISE framework [Walker et al. 2000] is an interesting, and somewhat controversial, attempt to correlate selected objective performance measures, such as transaction success and others, with user satisfaction in order to enable prediction of the latter from the former.

When applied correctly and in a timely fashion during development, the usability criteria listed above support comprehensive-if-not-exhaustive usability evaluation of a particular SDS. As expert evaluation is often difficult to obtain, quantitative evaluation remains an important issue. An important quantitative evaluation measure not mentioned so far is *transaction success.* The idea is to measure the extent to which the task was successfully completed, for instance, whether the system actually booked the flight ticket specified by the user [Bernsen et al. 1998]. However, high transaction success, in this sense, is compatible with cumbersome and not very user friendly dialogue design, lots of unnecessary error correction, etc. All it takes is a very persistent user! Furthermore, many task-oriented systems do not solve "monolithic" tasks such as the booking of a flight ticket. A frequently asked questions system, for instance, must solve as many tasks as the user asks questions. Domain-oriented systems do not solve tasks at all. For reasons such as these, attempts are being made to clearly define more fine-

grained dialogue success measures, looking at the success of processing each individual user input, counting the number of meta-communication turns, etc. As for quantifying the smoothness of spoken dialogue, measuring the *number of interaction problems* is an interesting approach which will be discussed in Section 6.

5 When (not) to use speech

The idea of speech-including multimodal interaction goes back, at least, to Bolt's idea of speaking into a graphical (screen) output environment [Bolt 1980]. In this context, the term "modality" may be traced back to Hovy and Arens' observation that e.g., tables, beeps, written and spoken natural language may all be termed 'modalities' in some sense [Hovy and Arens 1990]. This intriguing remark led one of the present authors to develop a theory, called modality theory, of *unimodal* and *multimodal* (re-)presentation of information in the physical *media* of graphics/light, acoustics/sound and haptics/touch, i.e. the media for information representation and exchange which would be available at the human-computer interface in the foreseeable future [Bernsen 1994]. The theory is based on generation from basic principles of an exhaustive set of unimodal output modalities in those media at a particular level of abstraction. The generated unimodal modalities can then be abstracted at higher levels of description as well as detailed-by-analysis at as many lower levels of description as required. Having been uniquely identified in this way, each unimodal modality can be analysed in detail at each level of abstraction, and, per level of abstraction, the unimodal modalities present at this level can be used to generate, or decompose, all possible multimodal combinations. Let us explain how this works by reference to Figure 5.1 from [Bernsen 2002] which shows the taxonomy of *output* modalities in modality theory.

The first-generated level is the *generic level* (second column from the left). At the *super level* (first column from the left), the generated modalities have been classified into linguistic, analogue, arbitrary and explicit modalities, respectively. This higher-level classification can be done in other ways, yielding differently organised modality trees with exactly the same unimodal modalities, such as a tree organised according to acoustic, graphic, and haptic modalities, or a tree organised according to static and dynamic modalities. So far, the generic-level *arbitrary* (13-16) and *explicit structure* (17-20) modalities do not seem to need further expansion. Arbitrary modalities include, e.g., the use of arbitrary output sounds for alarms. Explicit structure modalities include, e.g., the ubiquitous boxes used for grouping information in GUIs. An *analogue* modality is an information representation which, by contrast with most parts of most *linguistic* modalities, has perceivable similarity with what it represents.

The generic-level *linguistic* (1-8) and *analogue* (9-12) modalities are expanded in more detail at the *atomic level*. It is at the atomic level that one finds many familiar unimodal output modalities, such as graphically presented gesture (4a-c), graphical and haptic (e.g., Braille) static written text (5a, 7a), spoken discourse (6a), and static and dynamic graphical images (9a, 11a). To illustrate the unlimited "downwards" extensibility of the unimodal output hierarchy, static graphical written text (5a), for instance, such as the text which the reader is reading right now, is expanded at the *sub-atomic level* into typed (5a1) and hand-written (5a2) modalities. If one wants to generate-through-expansion, for instance in order to analyse in more detail, animated output faces, one would have to expand from the atomic to the sub-atomic level the generic-level modality dynamic graphic images (12a).

Once the unimodal modalities have been uniquely identified as shown in Figure 5.1, it becomes possible to analyse their individual *modality properties* in depth. Much of this is still unpublished work. Importantly, the hierarchical organisation of the modality tree means that properties are inherited downwards in the hierarchy. Thus, once the properties of, e.g., acoustics have been analysed, these properties get inherited by spoken discourse (6a). For instance, speech is omnidirectional *because* sound is omnidirectional. As we know, the omni-directionality of speech has important implications for the use of speech in human-computer interfaces. It is this property of speech which implies that speech is undesirable for providing bank account numbers to bank teller machines on the street. The analysis of spoken discourse only has to add to the analysis of acoustic information the - incidentally, quite rich - peculiarities of spoken discourse information over and above what already characterises acoustic information in general.

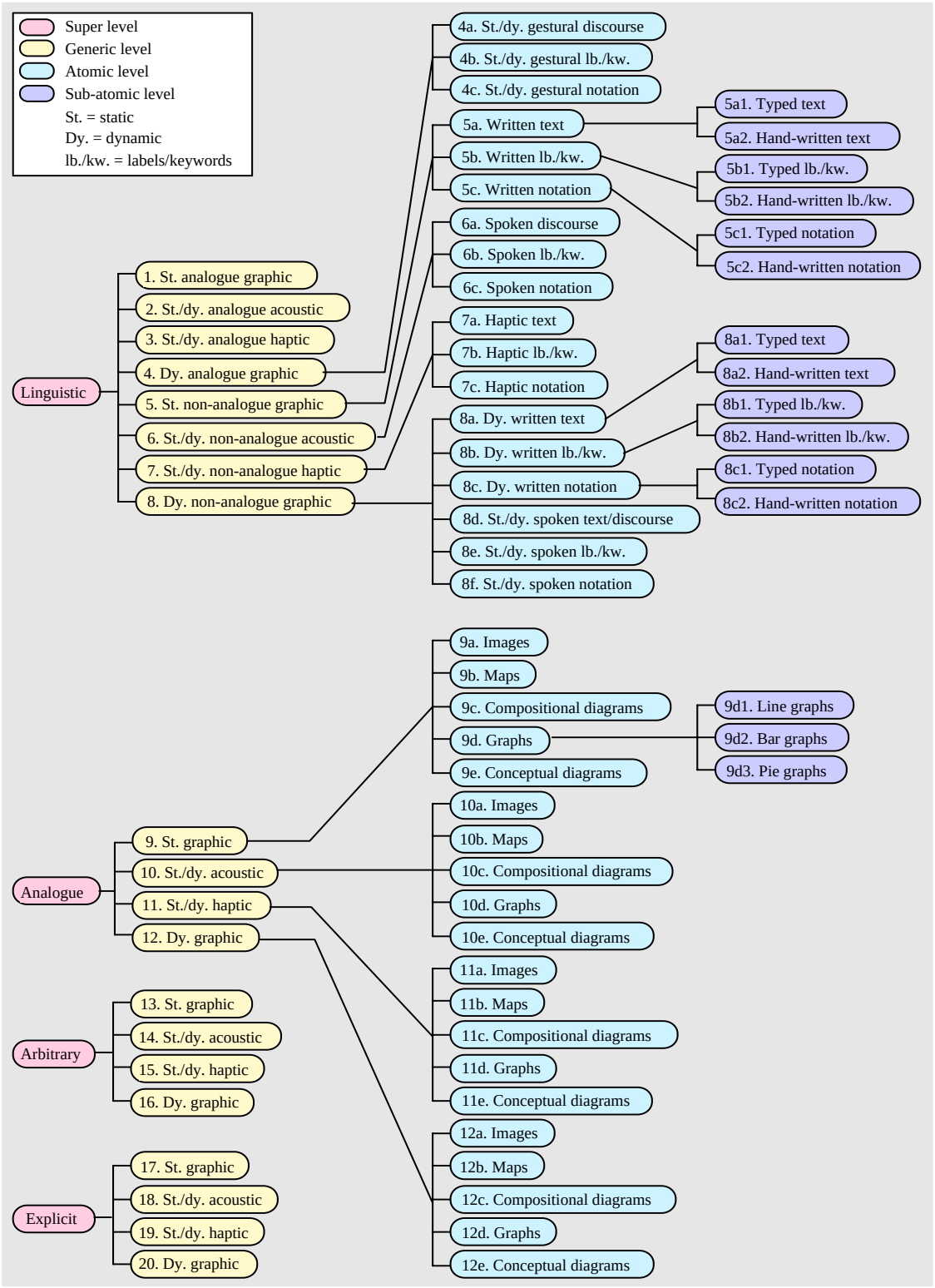

Figure 5.1. The taxonomy of unimodal output modalities. The four levels are, from left to right: super level, generic level, atomic level and sub-atomic level.

Based on analysis of what turned out to be the relevant modality properties, we have applied modality theory to the issue of speech functionality, i.e. the question of when (not) to use speech for interfacing with computer systems. It is easy to see from Figure 5.1 that the combinatorics of potential unimodal modality combinations which include speech are quite significant if one, for instance, wishes to investigate all <n modality combinations where n = 11. Therefore, rather than analysing all possible combinations, we did two studies of the literature. In the first study [Bernsen 1997], all of the 120

speech functionality claims made in the +20 papers in [Baber and Noyes 1993] were evaluated by reference to modality properties. Reflecting the state of the art at the time, [Baber and Noyes 1993] included rather few claims about the use of speech in a multimodal context. So, the second study [Bernsen and Dybkjær 1999a, b], using the same methodology, evaluated all of the 153 speech functionality claims made in a selection of +20 papers published between 1993 and 1998. Each of this total of 273 claims were evaluated as to whether a claim was justified, supported, or corrected by modality properties. An example of a claim evaluation is:

> ***48.*** *Interfaces involving spoken ... input could be particularly effective for interacting with dynamic map systems, largely because these technologies support the mobility [walking, driving etc.] that is required by users during navigational tasks. [14, 95]*
>
> *Data point 48.* ***Generic task*** *[mobile interaction with dynamic maps, e.g. whilst walking or driving]: a speech input interface component could be* ***performance parameter*** *[particularly effective].*
>
> *Justified by MP5: "Acoustic input/output modalities do not require limb (including haptic) or visual activity." Claims type:* ***Rsc.*** (*recommends speech in a multimodal combination).*
>
> ***NOTE:*** *The careful wording of the claim "Interfaces involving spoken ... input". It is not being claimed that speech could suffice for the task, only that speech might be a useful interface ingredient. Otherwise, the claim would be susceptible to criticism from, e.g., MP1. Note also that the so-called "dynamic maps" are static graphic maps, which are interactively dynamic.*
>
> *True.*

What we found was that the use of modality properties justified, supported, or corrected 97% of the claims in the first study of 120 claims and 94% of the claims in the second study of 153 claims. Assuming the representativity of the analysed claims with respect to all possible claims about speech functionality, modality properties are demonstrably quite relevant to judging speech functionality in early SDS design and development. Equally interesting was the following finding. The 120 evaluations made in the first study required reference to 18 modality properties. However, despite the fact that the 153 claims evaluated in the second study were far more concerned with the use of speech in many different multimodal contexts, their evaluation only required addition of a mere seven modality properties. For what it is worth, this lends plausibility to the conclusion that the issue of when (not) to use speech becomes tractable when addressed on the basis of modality theory. Table 5.1 shows a fragment of the modality properties used in the evaluation.

No.	Modality	MODALITY PROPERTY
MP1	Linguistic input/output	Linguistic input/output modalities have interpretational scope, which makes them eminently suited for conveying abstract information. They are therefore unsuited for conveying high-specificity information including detailed information on spatial manipulation and location.
MP2	Linguistic input/output	Linguistic input/output modalities, being unsuited for specifying detailed information on spatial manipulation, lack an adequate vocabulary for describing the manipulations.
MP3	Arbitrary input/output	Arbitrary input/output modalities impose a learning overhead which increases with the number of arbitrary items to be learned.
MP4	Acoustic input/output	Acoustic input/output modalities are omnidirectional.
MP5	Acoustic input/output	Acoustic input/output modalities do not require limb (including haptic) or visual activity.
MP6	Acoustic output	Acoustic output modalities can be used to achieve saliency in low-acoustic environments. They degrade in proportion to competing noise levels.
MP7	Static graphics/haptics input/output	Static graphic/haptic input/output modalities allow the simultaneous representation of large amounts of information for free visual/tactile inspection and subsequent interaction.

Table 5.1. Examples of modality properties.

6 Guidelines for cooperative spoken dialogue

The call for guidelines in support of interaction design is not new. In HCI, the self-defeatingly large guidelines sets of the 1970s became replaced by smaller and far less specific ones, such as Schneidermann's '8 golden rules' for general interaction design [Schneidermann 1987]. [Baber 1993] reviewed the need for SDS design guidelines, considering Grice's well-established conversational maxims of human-human spoken conversation [Grice 1975] and Schneidermann's rules. He concluded that it was far from obvious how to use such principles for SDS design. [Bernsen et al. 1998] argued that a key to successful dialogue design is to ensure adequate system co-operativity during interaction. To this end, they present a set of empirically based guidelines for task-oriented, shared-goal spoken interaction, which, at a late stage in their development, had come to incorporate the Gricean maxims. System co-operativity is the best possible means for preventing the need for on-line error handling during spoken dialogue. On-line error handling hampers task completion, has a negative effect on user satisfaction, and is, in some respects, at least, difficult to design for.

The guidelines cover seven different aspects of interaction. An aspect serves to highlight the property of interaction addressed by a particular guideline, thus identifying dimensions of co-operativity over and above the level of the cooperative guidelines themselves. At guideline level, we distinguish between generic and specific guidelines (GGs and SGs). A generic guideline is general and typically states: "Do (make, be, avoid, provide, etc.) X". A generic guideline may subsume one or several specific guidelines related to the generic guideline in a kind-of relationship. Specific guidelines specialise the generic guideline to certain classes of phenomena, thus elaborating what the interaction model developer should be looking for when designing cooperative system behaviour.

The guidelines were originally developed during the design, implementation and test of the interaction model for the Danish Dialogue System, 1991-1996 [Bernsen et al. 1996, Bernsen et al. 1998]. A first set of guidelines was developed on the basis of analysis of 120 examples of user-system interaction problems identified in a corpus of dialogues from Wizard of Oz (WOZ) simulations of the system. The guidelines were then refined and consolidated through comparison with Grice's maxims which turned out to form a proper subset of the guidelines. The consolidated guidelines were tested as a tool for the diagnostic evaluation of a corpus of 57 dialogues collected during a scenario-based, controlled user test of the implemented Danish Dialogue System. Nearly all dialogue design errors in this corpus could be classified as violations of the guidelines. Only two specific guidelines on meta-communication, SG10 and SG11 (see below), had to be added. This was no surprise as meta-communication had not been simulated and hence was mostly absent in the WOZ corpus. Completeness of the guidelines set has later been tested on other corpora from shared-goal, task-oriented spoken human-computer dialogue, showing no need for adding new guidelines [Dybkjær et al. 1997].

In the following we take a brief walkthrough of the guidelines. A * means that a guideline corresponds to a Gricean maxim. 'Meta-communication', or communication about the communication itself, means that communication error handling takes place, resulting in "lost" dialogue turns, user dissatisfaction, and potential transaction failure. In most applications, user-initiated clarification meta-communication, in particular, commands respect among SDS developers because it is difficult or impossible to design for. More details on the guidelines, including examples of violations, can be found at http://www.disc2.dk/tools/codial/.

Aspect 1. Informativeness

GG1: Say enough*. If the system's contribution is not sufficiently informative, this will typically lead to misunderstanding which may only be detected much later during interaction, if at all, or, at best, lead to an immediate request for clarification by the user.

SG1: State commitments explicitly. Commitments made during the dialogue should be summarised to make sure that the key information exchanged was correctly understood, e.g., on what a user committed himself to buy. This is sometimes called *summarising feedback.*

SG2: Provide immediate feedback. It is good practice to provide immediate, implicit or explicit feedback on each piece of information provided by the user which is intended to contribute to the

achievement of the goal of the dialogue, such as making a flight ticket reservation. The sooner misunderstandings can be corrected, the better.

GG2: Don't say too much*. The user may become inattentive or try to interrupt if too much information is being provided in a single system turn. In the worst case, the user may start clarification meta-communication as a result.

Aspect 2. Truth and evidence

GG3: Don't lie*. The user must be able to trust what the system says. Users have good reason to become annoyed if the system provides false information on, e.g., departure times, prices or meeting venues. Still, this may happen, for instance because of bugs in the database.

GG4: Check what you say*. The system must make sure that information is correct before giving it to the user. Otherwise, the implication may be very much the same as for GG3.

Aspect 3. Relevance

GG5: Be relevant*. Lack of relevance in the system's utterances will typically lead to confusion and clarification dialogue. System output irrelevance may be caused by misrecognition or misunderstanding. The system's reply may be perfectly relevant given its interpretation of the user's utterance but totally irrelevant given what the user actually said.

Aspect 4. Manner

GG6: Avoid obscurity*. Obscurity naturally leads to doubt and need for clarification in the user. Therefore it should be carefully checked that the system's output is not obscure.

GG7: Avoid ambiguity*. Ambiguity creates a need for clarification *if* detected by the user. If undetected, as often happens, the user may select a non-intended meaning of system output, and anything can go wrong leading to repair meta-communication or even transaction failure.

SG3: Ensure uniformity. Uniform formulations of a question may ensure that it is interpreted in the same way in different contexts. Moreover, the use of uniform formulations helps reduce users' vocabulary because users tend to model the phrases used by the system. The drawback is the risk that the dialogue appears monotonous.

GG8: Be brief*. The user may become bored and inattentive, and surprisingly quickly so, or may try to interrupt if the system talks too much.

GG9: Be orderly*. To avoid user-initiated clarification, the system should address the task-relevant topics of interaction in an order which is as close as possible to the order expected by the user. Studying the structure of human-human conversation in the domain for which the system is being designed may support orderly interaction design.

Aspect 5. Partner asymmetry

GG10: Highlight asymmetries. A non-normal interaction partner should inform its partners of the particular non-normal characteristics which they should take into account in order to act cooperatively, e.g. limited understanding capabilities. However, such requests must be feasible. If they are not, difficult or impossible cases of miscommunication may proliferate.

SG4: State your capabilities. Users must be told what the system knows about and what are its limitations. This can be difficult but is of particular importance in walk-up-and-use systems where users do not have access to, e.g., written information about the system.

SG5: State how to interact. Like SG4, SG5 addresses both the system's task capabilities and its communication capabilities. If the system is unable to handle some task in a standard way or is only able to handle the task in one among several standard ways, this should be communicated to users to prevent interaction failure.

Aspect 6. Background knowledge

GG11: Be aware of users' background knowledge. The system needs to adjust to users' background knowledge and inferences based thereupon. Otherwise, the users may fail to understand the system and initiate clarification meta-communication.

SG6: Be aware of user inferences. If the system does not take into account possible user inferences by analogy, this may invite users to ask clarification questions or leave them with unanswered questions.

SG7: Adapt to the target group. There are major differences between the needs of novice and expert users. If the system favours expert users, it is likely to fail as a walk-up-and-use system. If it favours novice users, it is likely to be perceived as cumbersome and redundant by expert users.

GG12: Be aware of user expectations. To be an expert within its declared domain of expertise, the system must possess the amount and types of background knowledge which a user legitimately may expect it to have. Otherwise users may become confused or annoyed with what they rightly regard as a deficient system.

SG8: Cover the domain. The system must be able to provide appropriate domain information when and as required by its users. The system must also be able to make appropriate inferences to avoid lengthy and inefficient turn-taking which only serves to clarify something which the system could have inferred on its own.

Aspect 7. Repair and clarification

GG13: Enable meta-communication. Users as well as systems need to initiate clarification or repair meta-communication from time to time due to, e.g., system violation of a cooperativity guideline, user inattention, or system misunderstanding.

GG9: Enable system repair. If user input cannot be interpreted as meaningful in the context, the system must be able to ask for repetition or otherwise indicate that it did not understand what was said.

GG10 Enable inconsistency clarification. If the user's input is inconsistent, clarification becomes necessary. The system should not try to guess the user's priorities because if the guess is wrong, the user will have to initiate meta-communication instead, possibly in the form of clarification.

GG11: Enable ambiguity clarification. If the user's input is ambiguous, clarification becomes necessary. As in GG10, the system should not try to guess what the user means.

The guidelines are used by manually evaluating if each system utterance in isolation as well as in context violates any of the generic or specific guidelines. If it does, this is a potential source for interaction problems which should be removed. Using the guidelines as design guidelines thus means to apply them to analytical 'walk-throughs' through the emerging interaction model for the SDS that is being designed.

It should be noted that guidelines may support one another as well as conflict when applied during interaction design. When guidelines conflict, the designers have to trade off different design options against one another, with each option having a different weighting of the guidelines. When designing a system introduction, for instance, developers may find that GG2 (don't say too much) conflicts with GG1 (say enough), SG4 (tell what the system can and cannot do) and SG5 (instruct on how to interact with the system). If the introduction is long and complex, and even if all the points made are valid and important, users tend to get bored and inattentive (GG8). On the other hand, if the introduction is brief or even non-existent, important information may have been left out, increasing the likelihood of interaction problems during task performance.

7 Conclusion

In this paper, we have presented the current baseline SDS, i.e. the task-oriented spoken dialogue system, and used it as a model for generating a far wider space of SDS aspects, all of which are being investigated today. We then briefly described three different approaches to SDS usability. The first approach could perhaps be said to specialise HCI to the particular field of SDS usability, whereas the two other approaches reach into the foundations of multimodal interaction and cooperativity in shared goal, task-oriented dialogue, respectively. As Section 3 on the variety of SDSs attempts to demonstrate, research into SDSs usability is currently facing an exponential complexity of new, emerging types of application. The sheer amount of new usability issues arising will, no doubt, require substantial effort before they can be satisfactorily resolved.

8 References

8.1 Literature

[Baber and Noyes 1993] Baber, C. and Noyes J. (Eds.). *Interactive Speech Technology*. London: Taylor & Francis, 1993.

[Baber 1993] C. Baber: Developing interactive speech technology. In Baber and Noyes 1993, 1-18.

[Baekgaard et al. 1995] A. Baekgaard, N. O. Bernsen, T. Brøndsted, P. Dalsgaard, H. Dybkjær, L. Dybkjær, J. Kristiansen, L. B. Larsen, B. Lindberg, B. Maegaard, B. Music, L. Offersgaard, and C, Povlsen: The Danish Spoken Dialogue Project - A General Overview. *Proceedings of the ESCA workshop on Spoken Dialogue Systems*, Vigsø, Denmark, 1995, 89-92.

[Bernsen 1994] Bernsen, N. O. Foundations of multimodal representations. A taxonomy of representational modalities. *Interacting with Computers* 6, 4, 347-71, 1994.

[Bernsen 1997] Towards a tool for predicting speech functionality. *Speech Communication* 23, 181-210, 1997.

[Bernsen 2002] Bernsen, N. O.: Multimodality in language and speech systems - from theory to design support tool. In Granström, B., House, D., and Karlsson, I. (Eds.): *Multimodality in Language and Speech Systems*. Dordrecht: Kluwer Academic Publishers 2002, 93-148.

[Bernsen 2003a] Bernsen, N. O.: When H. C. Andersen is not talking back In Rist, T., Aylet, R., Ballin, D. and Rickel, J. (Eds.): *Proceedings of the Fourth International Working Conference on Intelligent Virtual Agents (IVA'2003)*, Kloster Irsee, Germany, 2003. Berlin: Springer Verlag 2003, 27-30.

[Bernsen 2003b] Bernsen, N. O.: On-line user modelling in a mobile spoken dialogue system. In Bourlard, H. (Ed.): *Proceedings of Eurospeech'2003, 8th European Conference on Speech Communication and Technology*, Geneva, Switzerland. Bonn: International Speech Communication Association (ISCA), 2003, Vol. I, 737-740.

[Bernsen et al. 1996] Bernsen, N. O., Dybkjær, H. and Dybkjær, L.: Cooperativity in human-machine and human-human spoken dialogue. *Discourse Processes*, Vol. 21, No. 2, 1996, 213-236.

[Bernsen et al. 1998] Bernsen, N. O., Dybkjær, H. and Dybkjær, L.: *Designing Interactive Speech Systems. From First Ideas to User Testing*. Springer Verlag 1998.

[Bernsen and Dybkjær 1999a] Working Paper on Speech Functionality. *Esprit Long-Term Research Project DISC Report* D2.10. University of Southern Denmark. See www.disc2.dk, 1999.

[Bernsen and Dybkjær 1999b] A theory of speech in multimodal systems. In: Dalsgaard, P., C.-H. Lee, P. Heisterkamp & R. Cole (Eds.). *Proceedings of the ESCA Workshop on Interactive Dialogue in Multi-Modal Systems*, Irsee, Germany. Bonn: European Speech Communication Association: 105-108, 1999.

[Bertenstam et al. 1995] Bertenstam, J., Beskow, J., Blomberg, M., Carlson, R., de Serpa-Leitao, A., Elenius, K., Granström, B., Gustafson, J., Hunnicutt, S., Högberg, J., Lindell, R., Neovius, L., Nord, L., and Ström, N.: The Waxholm system - a progress report. In *Proceedings of the ESCA Workshop on Spoken Dialogue Systems*, Vigsø, Denmark, 1995, 81-84.

[Bolt 1980] Bolt, R. A.: "Put-That-There": Voice and gesture at the graphics interface, *Computer Graphics*, 14, 3, 262-270, 1980.

[Bossemeyer and Schwab 1991] R. W. Bossemeyer and E. C. Schwab: Automated alternate billing services at Ameritech: Speech recognition and the human interface. *Speech Technology Magazine* 5, 3, 1991, 24-30.

[Bourlard 2003] Bourlard, H. (Ed.): *Proceedings of Eurospeech'2003, 8th European Conference on Speech Communication and Technology*, Geneva, Switzerland, 2003. Bonn: International Speech Communication Association (ISCA), 2003.

[Cassell et al. 2000] Cassell, J., Sullivan, J., Prevost, S., Churchill, E. (Eds.): *Embodied Conversational Agents*. MIT Press, Cambridge, MA (2000).

[Charfuelán and Bernsen 2003] Charfuelán, C. and Bernsen, N. O.: A task and dialogue model independent dialogue manager. In Angelova, G., Bontcheva, K., Mitkov, R., Nicolov, N., and

Nikolov, N. (Eds.): *Proceedings of the International Conference on Recent Advances in Natural Language Processing* (RANLP-2003), Borovets, Bulgaria, 2003. INCOMA, Shoumen, Bulgaria 2003, 91-97.

[DARPA 1992] DARPA. *Proceedings of the Speech and Natural Language Workshop.* San Mateo, CA, Morgan Kaufmann, 1992.

[Dybkjær and Bernsen 2000] Dybkjær, L. and Bernsen, N. O.: Usability issues in spoken language dialogue systems. In *Natural Language Engineering*, Special Issue on Best Practice in Spoken Language Dialogue System Engineering, Volume 6 Parts 3 & 4, 2000, 243-272.

[Dybkjær et al. 1997] Dybkjær, L., Bernsen, N. O. and Dybkjær, H.: Generality and objectivity. Central issues in putting a dialogue evaluation tool into practical use. *Proceedings of the ACL/EACL Workshop on Spoken Dialog Systems*, Madrid, 1997.

[Grice 1975] Paul Grice: Logic and conversation. In P. Cole and J. L. Morgan (Eds.), *Syntax and Semantics* Vol. 3: *Speech Acts*. New York: Academic Press 1975, 41-58.

[Haton 1988] J. Haton: Knowledge-based approaches in acoustic-phonetic decoding of speech. In Heinrich Niemann, M. Lang, and G. Sagerer (Eds.): *Recent Advances in Speech Understanding and Dialog Systems.* NATO ASI Series, Vol. F46, Berlin, Springer Verlag, 1988, 51-70.

[Hovy and Arens 1990] When is a picture worth a thousand words? Allocation of modalities in multimedia communication. Paper presented at the *AAAI Symposium on Human-Computer Interfaces,* Stanford, 1990.

[Karlsson 1999] A survey of existing methods and tools for development and evaluation of speech synthesis and speech synthesis quality in SLDSs. *DISC Report* D2.3, 1999.

[Niemann et al 1988] H. Niemann, A. Brietzmann, U. Ehrlich, S. Posch, P. Regel, G. Sagerer, R. Salzbrunn, and G. Schukat-Talamazzini: A knowledge based speech understanding system. *International Journal of Pattern Recognition and Artificial Intelligence* 2, 2, 1988, 321-350.

[Peckham 1993] J. Peckham: A new generation of spoken dialogue systems: Results and lessons from the SUNDIAL project. In *Proceedings of Eurospeech'93*, Berlin, Germany, 1993, 33-40.

[Schneidermann 1987] Ben Schneidermann: *Designing the User Interface*. Reading, MA, Addison-Wesley, 1987.

[Turing 1950] Turing, A.: Computing machinery and intelligence. *Mind* 59, 1950, 433-60.

[Walker et al. 2000] Walker, M.A., Kamm, C.A. and Litman, D.J.: Towards developing general models of usability with PARADISE. *Natural Language Engineering, Special Issues on Spoken Dialogue Systems*, Vol. 6, No. 3, 2000.

[Walker et al. 2002] Walker, M, Rudnicky, A., Prasad, R., Aberdeen, J., Bratt, E., Garofolo, J., Hastie, H., Le, A., Pellom, B., Potamianos, A., Passonneau, R., Roukos, S., Sanders, G., Seneff, S. and Stallard, D.: DARPA Communicator: Cross-system results for the 2001 evaluation. *Proceedings of 7th International Conference on Spoken Language Processing* (ICSLP), 2002, 269-272.

8.2 Websites

DISC: www.disc2.dk

NICE: http://www.niceproject.com/

SmartKom: http://www.smartkom.org/