Building a framework for developing interaction models: Overview of current research on dialogue and interactive systems

B.W. van Schooten

Faculty of Computer Science, University of Twente

Postbus 217, 7500 AE, Enschede, The Netherlands

schooten@cs.utwente.nl

February 15, 1999

Abstract

This text is an overview of literature relevant to the development and modelling of interactive systems. It is the most comprehensive part of the documentation I made during the literature research stage of my PhD project, which should result in a modelling framework for interactive systems. More precisely, the scope of this project is natural language, multimodal, and virtual reality systems, and ways of modelling the dynamics of interaction of such systems for the purpose of aiding system development. The topics addressed in this text naturally include existing systems and modelling techniques, but theoretical frameworks and methodological issues are also addressed. Short introductions are given to cognitive psychology, semiotics, communication models, and HCI. Systems and strategies are reviewed, covering the topics multimodality, redundancy and confirmation, and ways of modelling dialogue and task structure. Methodological issues are reviewed, in particular some methodologies, and strategies for design/implementation and analysis/evaluation. Finally, some conclusions are given, as well as an indication of the directions of my future research.

Contents

1 Foreword

This text is a result of my work as a PhD student at the Parlevink project, which is part of the TKI (Taal, Kennis, en Interactie) group, formerly called SETI (Software Engineering en Theoretische Informatica). Thanks go to my advisors E.M.A.G. van Dijk, A. Nijholt, and R. op den Akker for reviewing this text.

2 What this text is about

This text is meant as an overview of the field of natural language interaction, multimodal interaction, and general HCI. This includes theories, general HCI issues, methodological issues, modelling and analysis techniques, design enviroments, existing systems and existing interaction strategies. It is an overview of most of the literature I have accumulated until now, classified and summarised in order to allow me, and other people, to retrieve information and literature on the subject more easily. This text is a neatened snapshot of a draft text that will be maintained further as new literature and ideas come in. The newest version of the draft text can be found on my homepage:

`http://wwwhome.cs.utwente.nl/~schooten/`

In the beginning of most sections, some introductory literature on the subject is listed. Note that some of the topics are covered in more detail than others. This reflects:

1. the areas of emphasis of my research up till now,
2. attempts to make the text more self-contained,
3. the areas I knew particularly little about.

So, some citations are little more than pointers, meant to show where to find more information about the subject. Other citations include lists of models or systems that were reviewed in them, showing where to find further information about those particular models and systems, without having to cite literature on them directly. Other citations include a full summary.

The preliminary title of my PhD thesis is 'Interaction models in (spoken) dialogue systems'. The initial PhD proposal consists of building a modelling framework to obtain more control over the development of interaction models in (spoken and natural language) interactive human/computer systems. Meanwhile, the scope has extended to include multimodal systems as well.

2.1 Interaction models

An 'interaction model' or 'dialogue model' dictates the interaction patterns, styles or strategies that are and can be used in the interaction between human and computer system. Knowledge about existing or possible interaction models should facilitate a more structured Human Computer Interaction (HCI) design. In particular, more needs to be known about designing interaction for information retrieval and exploration tasks, done by different kinds of users, both computer-naive and experienced. Also, more needs to be known about some of the more interesting interaction types by means of which such interaction may be achieved, in particular natural language (NL) and multimodal interaction.

2.2 Objectives of the intended framework

Note that the following objectives are very much abstract and general objectives, but it does show the benefits of each objective, and the different options available. The purpose of this text is to review the literature in order to enable us to identify more specific objectives, and to obtain an idea of the specific form that an actual framework may take.

2.2.1 Specification

A framework may provide a means to specify an interaction model, i.e. to 'cast' any (envisioned) dialogue system or user interface within a well-defined class of such systems into a uniform specification, with as little loss of information as possible. This may include the relevant task, goal, and environment. Possible benefits are: easier comparison of systems and implementations, processing of corpora, or more mutual applicability and transferability of software tools or theories based on the framework.

2.2.2 Abstraction

Furthermore, if the framework would allow abstraction (i.e. simplification with as little loss of relevant information as possible) of specific aspects of the interaction, then interaction models may be understood more easily in some respects. For example, it may lay a stronger link between NL and other types of HCI, allow abstraction from content or modality in multimodal interaction, or enable a more balanced choice of evaluation metrics.

2.2.3 Modelling human users

If the framework would explicitly incorporate description of features or variables of human cognition, it may provide a means to understand, model, and generally talk about human cognition. It might provide a basis of understanding and possibly predicting aspects of individuals and individual differences. It might also provide a basis for forming criteria for when, or reasons why, some types of interaction may or may not work in some situations.

2.3 Overview of the current text

Part I describes various theoretic views of and approaches to human cognition, human interaction with the environment, and communication. This should give a notion of the available intuitive ideas and concepts being used in the relevant areas. The areas that have been considered relevant up to now are: theories related to cognition, HCI, formal communication models, agent models, and semiotics.

Part II gives an overview of real situations, existing interaction strategies and dialogue systems, and existing HCI methodologies, methods and tools. The emphasis of this part is practical, explaining real human/computer systems, their architecture, and how they are and can be built.

Part I
Theories

In this part, we will look at cognition, interaction, and communication. In our terminology, cognition is anything that goes on inside the human mind. A cognitive framework is any theoretical framework that helps in explaining aspects of cognition. Communication is similar to interaction, but implies meaning and cooperation: the parties involved must understand and meaningfully react to each other in a certain sense. Interaction is a more general term, and covers any kind of interaction, whether possibly meaningful or not.

3 Cognitive frameworks

3.1 Cognitive psychology

The best-known cognitive framework is cognitive psychology. It is the study of human intelligence, viewing thought and intelligence as computation (Pasquini et al., 1998). This is in clear contrast with its predecessor, behaviourism, which effectively denies that there is anything substantial going on inside the mind. The traditional viewpoint is that 'computation' is analogous to symbol manipulation as found in classical AI.

In an introductory text on cognitive psychology, one typically finds the following aspects of human cognition (Veer and Lenting, 1995):

- Perception and primary memory: The human sensory apparatus has relatively well-measured properties. Sensory data is apparently stored in great detail for very short amounts of time, like a few hundred msec. This is called primary memory. The recognition of elements in the environments, sometimes called 'gestalts', like visual objects, letters, and sounds, depends on the raw sensory data and the situation or context in which the data is received.
- Short term memory (STM): Some perceptions get stored ('coded') into the STM. It has only a limited and fixed capacity: any individual has between 5 and 9 memory segments, called 'chunks'. When chunks are somehow too similar, they may be confused and the capacity will be less. Coding implies a great deal of interpretation, and depends greatly on knowledge already present. It is generally assumed that experts in a domain need fewer chunks to code the same information, because they have a larger repertoire of possible chunks available. When the memory is full, chunks are discarded. In the most basic STM model, these are the chunks that haven't been used for the longest time.
- Long term memory (LTM): Transfer from STM to LTM occurs slowly and after repetition. The LTM contains well-structured episodic and semantic information. The high degree of structure is related to people's use of analogies in understanding. A related concept is 'consistency': information to be learned is consistent if analogies can be used effectively. Episodic aspects of the LTM are sometimes modelled using event schemas, which are like event sequence templates, or production rules, which are like IF...THEN clauses. Semantic aspects are often modelled using semantic nets, which are labelled graphs. The nodes in semantic nets are concepts, and the arcs are relations, labelled with the type of the relation, like 'subclass', 'instance', 'is-part-of', etc. The nodes are thought to be closely related to nouns in language use, and to objects in the physical world. Episodic aspects of language are thought to be related to episodic memory. This

implies that the surface form of language does not have an independent meaning in the thinking process, but is tightly coupled to real-world imagery.

- Reasoning: Reasoning is seen as activity of a central symbolic processing unit, much like many classical AI reasoning models. The processing unit has a limited capacity, and the extent to which it is used is called 'cognitive load'. High-level reasoning models, such as agent models, often assume the existence of clearly-defined goals and intentions. This is somewhat similar to the idea of Searle's 'intentional states' (Searle, 1983). Many also assume the construction and execution of plans, which are series of actions, in order to accomplish goals. These are called plan-based models.
- Individual differences: Next to the knowledge structure and STM capacity, other differences between individuals are identified. Some of them may change easily by means of learning or training, while others may not. Some cognitive variables, from easily changeable to unchangeable:
  - knowledge: the presence of certain production rules, event schemas, or semantic nodes.
  - strategies: holism–serialism (general versus specific knowledge seeking)
  - styles: field dependency (the extent to which people are influenced by the immediate context), operation learning (stepwise learning of procedures), comprehension learning (discovery of general rules), reflection–impulsivity (accuracy versus speed), and visual–verbal (preferring visual versus text representation).
  - personality: capacity of the STM, intelligence, creativity, introversion–extraversion.
- Error: People typically make errors on a regular basis. Error rate increases with higher cognitive loads. Usually, two kinds of errors are distinguished, according to the cognitive level at which they happen: there are slips, which are errors occurring in routine tasks, and mistakes, which are planning and reasoning errors. In Rizzo's lecture in (Pasquini et al., 1998), there is a classification into three kinds of errors: slips, lapses and mistakes. This more precise distinction is based on the existence of intentions and plans. A slip is a mismatch between intended action and actual action. A lapse is a memory failure, in which either the goal, the action to be executed, or the information needed to execute the action is not in memory when needed. A mistake is a mismatch of plan or subgoal with the main goal, which may occur either because of lack of knowledge of the situation, erroneous knowledge, or faulty reasoning.

3.1.1 Extensions

However, people have argued that traditional cognitive psychology does not account well enough for context. Other frameworks try to remedy the situation (Nardi, 1992): there is activity theory (Kaptelinin et al., 1995), distributed cognition theory (Zhang and Norman, 1994), and situated action modelling. Basically, they assert that human cognition cannot be seen as separate from the environment, in particular, the tasks and tools in the environment. It is said that, as a human may shape a tool according to a task at hand, the tool will also shape the human, as the human is restricted to working with the limitations of the tool; humans and their tools co-evolve. Representation tools are a relevant case: an external representation is read (internalised) or formed (externalised). Thinking is seen as an internal activity with internal representations. Use of a specific external representation may make cognitive tasks easier or harder, and may eventually shape thought. Consider the following example: the two-player 'game of 15'. In this game, there are nine numbers, $1 \ldots 9$. Each

player may in turn encircle a number. The player whose sum of numbers equals 15 first, wins the game. As it turns out, this game is simply tic-tac-toe with a different representation. However, one may expect that the original spatial version is quicker to understand and play by most people than the numerical one.

One concrete example of the importance of context can be found in Rizzo's lecture in (Pasquini et al., 1998): he considers distinguishing coins. A traditional cognitive psychologist might say that the distinguishability between two coins is adequately represented by difference in measurable features, such as colour and size, assuming that the human mind simply computes these to determine coin type. However, in reality, people's ability to distinguish two types of coins also depends on the existence of other coins: for example, if only one copper-coloured coin type exists, people may confuse any new copper coin type with the existing one, even though it is as different in size from the existing one as are any two of the existing silver-coloured coin types.

3.2 Semiotics

Semiotics is the study of signs. The best-known people who have laid the foundation for semiotics are Peirce (Peirce, 1958), Saussure (de Saussure, 1916), and Eco (Eco, 1976). For a full beginners' introduction to semiotics and some of its applications, see (Zoest, 1978). In semiotics, a sign identifies something that stands for something else, when interpreted by some individual interpreter in some individual situation. In its simplest form, a sign is a 2-tuple, consisting of: the form of the sign (the significatum), and what it stands for (the denotatum). The precise form of the significatum and denotatum, and how interpretation takes place, is not prescribed by general semiotics. In some semiotic theories though, a sign simply stands for another sign. For any individual, anything that involves interpretation may be called a sign: for example, smoke may be a sign of fire, a closed door may be a sign of a certain person's absence. This also includes signs of culture and convention, such as language, road signs, etc., at least when they indeed are interpreted as such. Note that the notion of the sign's significatum in these examples already implies some (relatively low) level of interpretation. Another sign may have participated at this level. In the first example, it may have been a sign consisting of a significatum in the form of a retinal pattern and a denotatum in the form of the concept 'smoke'.

Around this concept of sign, there are general classifications, some of which are used often. We will discuss the particularly popular classification into syntax, semantics, and pragmatics, and into icon, index, and symbol below.

So, firstly, semiotics identifies the existence of signs, in the form explained above, as units in human interpretation. Also note that the idea of interpretation as replacing one sign by another looks much like the symbolic processing paradigm of cognitive science. In these ways, semiotics is a cognitive framework. We may also view signs not only as units of interpretation, but as units of transmission by someone with the purpose of causing interpretation by another, where transmission is the inverse process of interpretation. In this way, semiotics is also a framework of communication. In both cases, semiotics by itself is a very loose framework, as little is actually prescribed. For example, it is not prescribed when to identify something as a sign, or as part of a sign, or as multiple signs, etc., so this is left to intuition or to other disciplines. However, there is a body of sign classifications available in the literature: specific signs are identified, described, and classified using semiotic concepts. Such semiotic theories exist for psychology, music, cinematography, HCI, etc.

3.2.1 Icon, index, and symbol

Peirce classifies signs according to the relation between the significatum and denotatum: icon (likeness), index (indirect manifestation), and symbol (relation by convention). Any sign may be more or less iconic, indexical, or symbolic in nature. According to Van Zoest, this classification is well-established because it has proven itself in practice. Each class may be attributed certain extra properties (Zoest, 1978):

- **icon.** If the significatum looks or sounds like the denotatum, or has another such likeness to the denotatum, then the sign is an icon. Example: a photograph of a person may be a sign for that person. Iconic signs are easy to interpret and 'seductive'.
- **index.** If the significatum is an indirect manifestation of, or is thought to be caused by, the denotatum, then the sign is an index. Example: smoke may be a sign of fire. Indexical signs are convincing, because they show us that something 'really happens'. According to Van Zoest, index is in a way similar to icon, because in both cases, significatum refers to manifestations of the denotatum, though it is not entirely made clear in what way they are similar or different. In my own opinion, the distinction is practical, according to the following criterium: with an icon, it is apparent that the icon is not a manifestation of the thing itself, while with an index, the sign does appear to be a manifestation of the thing itself. Example: seeing a person walk down the hall is an index of that person being at that particular location. A photograph of a person is an icon referring to that person. Seeing something that looks like a photograph of a person is an index referring to a photograph of that person.
- **symbol.** If the sign is part of rules and conventions made by people, it is called a symbol. Example: most English words are symbols. Since symbols are usually specially made for communication, symbols may be transmitted more easily and in more sophisticated ways than other kinds of signs. For example, to signify a person, it is easier to call that person's name than to draw a picture of that person. The name can also be used in sentences, stating all kinds of things about that person. Though even Plato was of the opinion that choosing iconic signs for representing something is superior to choosing arbitrarily-chosen symbols (in our time, Plato might have been an advocate of graphical user interfaces (GUIs)), Van Zoest also notes that people can get used to symbols very well. For example, note that even in highly iconic languages, like Chinese, many of the icons have changed so much in the history of the language that they have become hardly recognisable as icons. Probably, the signs of the language have adapted to increase efficiency of transmission and ease of reading, which are the more important factors after a language user has gotten used to the signs.

3.2.2 Syntax, semantics, pragmatics

Syntax, semantics, and pragmatics are about sign systems (called 'grammars' in Van Zoest's terminology), rather than individual signs. Van Zoest calls them semiotic syntaxis, semiotic semantics, etc. to distinguish them from the linguistic versions of the concepts. In semiotics, these concepts have a more abstract and general meaning. First, of course, one has to identify what is part of a 'sign system', and what is not. Examples of sign systems given in (Zoest, 1978) are: traffic signs (red for danger, circle for 'forbidden', etc.), fairy tales (they often start with 'Once upon a time...', there's usually a story with certain elements, etc.). Viewing one sign system in isolation may not be enough, as sometimes, multiple sign systems work together as a bigger sign system, for example, facial expression and tone may denote sarcasm, possibly inverting the meaning of a spoken sentence.

- **Syntax.** Syntax refers to the rules governing the structure of the significata of a sign system. Significata may seem relatively easy to analyse objectively, but there is no meaning in identifying significata as part of a sign system without at least looking at the semantics as well.
- **Semantics.** Semantics is the relation between significata and denotata of the signs in a sign system, in relation to the other signs. For example, the reason for success or failure of a sign transmission is a semantic issue.
- **Pragmatics.** Pragmatics refers to the relation between sign and sign user, in other words, why does the user use a sign, and what happens when a user uses that sign? Note that 'user' may both refer to a person who interprets a sign, and one who transmits a sign. According to Van Zoest, pragmatics is a relatively ill-covered terrain in semiotics. One well-known pragmatical classification of sign transmissions (and language utterances in particular) is the classification into locution, illocution and perlocution, originally proposed by (Austin, 1962). Locution is the information contained in an utterance. Illocution is the purpose that an utterance has, like informing, convincing, requesting, or demanding. Another illocutionary classification is expressive (informing) versus evocative (wanting to make the other party do something). Perlocution is the actual effect that a statement has.

3.2.3 Syntax, semantics, and pragmatics in dialogue

In dialogue, utterances which influence or are explicitly meant to control the course of the dialogue may be seen as pragmatical in nature, as they are a result of the goals of the participants in relation to the dialogue. On the other hand, one can imagine that there are rules governing dialogue control, which form a separate sign system.

Indeed, some people like to speak of 'conversational grammar' or 'dialogue grammar' to describe patterns found in sequences of utterances. This dialogue grammar is on a different level than the language grammar, and amounts to a classification of utterance types (for example statement, question, or affirmation) and possible sequences of these. Some criticism can be given to the 'dialogue grammar' view:(Good, 1989). See fig. 1 for a schematic view of this idea.

4 Communication

4.1 Abstract models

Communication complexity (Kushilevitz, 1997) addresses how two parties could compute a function, while each party only has a part of the needed information, with a minimum amount of communication.

Information theory (Shannon and Weaver, 1964) is a mathematical theory of computer-computer communication, but it has had some impact on people's thinking about human communication as well. It deals with efficiency and reliability of the communication of essentially raw data. One of the main ideas is that a well-expected event contains less information than a little-expected event because it can be shown that it can be coded more efficiently. The theory also addresses how redundancy can be added to communicate reliably over a noisy channel.

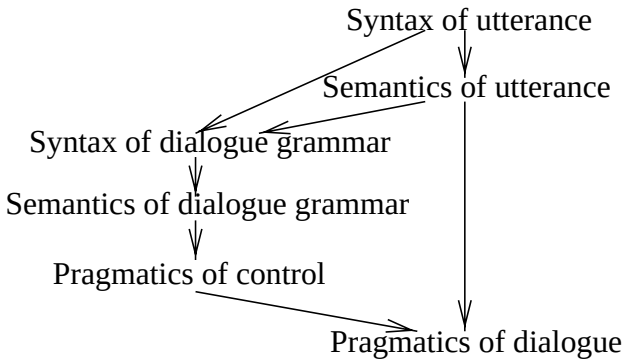

Figure 1: Levels of semiotics in dialogue
Possible placement of the linguistic and dialogue sign systems in the 'dialogue grammar' view. The arrows stand for 'is derived from' or 'leads to'. The previous dialogue utterances are not represented in this graph, but are assumed to be known. Note that both syntax and semantics of the single utterance are needed as input for the interpretation of the dialogue sign system. For example, a surface feature such as a question mark may indicate an intention to provoke an answer, though, depending on the meaning of the sentence, it may also happen to be meant as a rhetorical question.

4.2 Communication and cooperation

Some analyses of cooperating human behaviour exist, see for example (Darses et al., 1993) and (Karsenty, 1993). In these particular cases, the purpose is to gain insight in HCI issues. The article (Darses et al., 1993) analyses cooperating human partners in design dialogues and advisory dialogues, in a tutor-tutee setting. The data was segmented by hand into Basic Cooperative Behaviours (much like illocutionary acts, see section 3.2) and classified into one of 8 types, and the segments grouped into Basic Cooperative Interactions (much like subdialogues) and classified into one of 11 types. Various qualitative observations are made about tutor-tutee dialogues.

Well known are the Gricean cooperative principles (Grice, 1975): quantity (make response exactly as informative as required), quality (do not lie, do not state something you are uncertain of), relation (be relevant), manner (avoid obscurity of expression, ambiguity, be brief and orderly, etc.). Grice also discusses relation to other principles, and clashes and violations of principles. The principles can be said to have the common basis of maximising effective communication. The Gricean principles imply an understanding of the other participant's intentions. However, they do not explicitly say when the cooperating partner should take initiative.

Castelfranchi (Castelfranchi, 1991) decouples linguistic cooperation from goal-level cooperation. For example, compliance of language utterances (taking desires 'literally') may mean uncooperativeness at the goal level. He argues cooperation may or may not exist at the monitoring level (checking whether information is received), linguistic level (following of orders), and at the goal level independently. Note that he distinguishes between these three levels implicitly. Explicitly, he distinguishes only between level one and level two/three. He argues that his view is in conflict with most views of cooperative AI, which assume cooperation at the first two levels, but ignore the third. However, what is usually called 'cooperative behaviour' in NL systems also encompasses tracking the user's implicit goals and plans.

Some theories address ways of tracking and aiding a person's intentions. One is missing-axiom-theory (Smith, 1996), which is a theory of human reasoning. In this theory, goals are modelled as the intention to prove certain propositions. Wanting to obtain missing information (missing axioms) needed to prove the propositions is what results in dialogue. A related idea can be found in (Dessalles, 1998): here, it is proposed that cognitive conflict drives dialogue. An attempt to form a complete executable model of cooperative communication is found in the DenK project (Bunt et al., 1995). DenK includes NL parsing, the result of which is represented in underspecified language form (ULF), and belief modelling and visual knowledge modelling, using type theory to represent knowledge.

When communicating, people adapt to a certain extent to the other participant. This is a problem when envisioning new systems using previous experience. An example of lexical accommodation can be found in (Fais and ho Loken-Kim, 1995): people adapt their word usage to their dialogue partner. The cause that is suggested in this article is social approval.

4.2.1 Cooperative agent models

The origin of the term 'agents' lies in AI, where people found out that large knowledge bases became unmaintainable by a single inference engine, because they would become intractable or inconsistent. As a way around the problem, multiple inference engines (agents) with private databases and negotiation protocols were introduced. In computer science, use of the word 'agent' has recently become more fashionable, and is probably being used for many more things that it originally was. It is sometimes even used as a new name for the more established concept of module or subprogram.

There are several reasons for adopting the concept 'agents'. They are software engineering (as an alternative and extension to the object-oriented view), social and emergent system modelling, design of negotiation systems and protocols, and user interface modelling.

In all cases, 'agent-oriented' it is a metaphor to aid in thinking, like 'object-oriented'. Therefore, it is no surprise that the term 'agent' is based on the intuitive, possibly naive conception that people have when they think of the word 'agent'. Here, we assume any agent has at least one of the following qualities:

- 'autonomy' or 'pro-activeness': it is able to initiate relatively complex actions, or initiate actions without needing a specific trigger or command, for example after waiting a certain amount of time.
- 'intentionality': it is able to reason, and have explicit or complex intentions.
- 'communicativeness': it is able to communicate complex semantic and pragmatic information or is capable of negotiation with other agents.

Possible benefits of agent theory for dialogue system development are:

1. *Modelling of cognition.* When human and computer are viewed as two agents, the communication between them can be modelled using some of the agent theories.

   Ex. It is important that intelligent agents should communicate their own view in a clear way. The two parties usually found in NL systems are like agents in this respect: the system should understand how the user thinks, and especially when the system's understanding starts failing, it is important that the user understands how the system 'thinks'.

Ex. The protocols used are often a kind of formal versions of the ideas found in dialogue systems. Analysis of the workability (like self-consistency or completeness) of such protocols may be extendable to those of NL systems.

Ex. Existing agent implementation languages may be used to easily build prototypes of interactive systems.

2. *Modular design.* Which parts of the total interface know or need to know of each other? Should the different parts of a program that perform different functions be explicitly presented to the user as such?

   Ex. Some models assign roles to humans and computers after a first stage of design. (Palanque and Bastide, 1996) even uses formal models to assess this kind of design decisions.

   Ex. Anthropomorphic interfaces, for example in a virtual-reality (VR) environment, may be directly modelled using agents.

3. *Multiple parties.* It might be beneficial to be able to incorporate the possibility of more than one human party into the framework.

   Ex. In a VR environment with multiple computer and human agents, all parties may be viewed as agents.

We continue with an overview of existing agent systems, models, and theories.

(Norman, 1994) discusses social and ergonomic aspects of humans using computer agents. Some guidelines given are: keep a feeling of control, make sure people don't overexpect regarding agent's intelligence. Safety and privacy issues are also discussed.

A model of psychological agents is introduced in (Watt, 1997). The article addresses means to integrate human and computer parties effectively. It is argued that computer agents should be able to communicate with humans using means that are native to humans, because this would allow integration in human environments. Agents are classified by the number of understanding levels they incorporate. The levels are, from low to high, behavioural (any program), intentional (with reasoning engine) and psychological (able to understand/be understood by humans). A scheduler agent Luigi is explained in this article, though not in much detail.

A formal model of social agents is described in (Dignum and Linder, 1997). This model has four levels: informational (belief and knowledge), action (temporal aspects, including possibilities, agent actions and cause-effect relationships), motivational (wishes and goals), social (illocutionary acts: commitments, directions, declarations, assertions). Some related approaches are summarised also.

The PAC-Amodeus model (Nigay and Coutaz, 1997) is a compositional model for software and user interface design. PAC stands for Presentation-Abstraction-Control, referring to the composition of a basic unit (the agent) in a PAC specification. Note that this looks much like some of the Seeheim models discussed in section 5.2.1.

The Cyc project (Guha and Lenat, 1994) aims at providing an engine that allows common sense reasoning using complex inference over a huge database of logical statements. It is argued that such a large amount of common sense is needed to allow any two agents to communicate. Internally, Cyc is in a way also agent-oriented, as multiple independent sub-databases are maintained.

An article focussing on agent communication languages is (Nwana and Wooldridge, 1997). It is argued that cooperating multi-agent systems imply complex communication, because agents can not know everything that is relevant to the global task. Agent communication languages, ontologies, and software tools are reviewed. This includes Knowledge Query and

Manipulation Language (KQML), which is like speech act theory. Each KQML message has illocutionary meaning, which is explicitly stated in the message. Other information about the message are: the body, the language in which the body is written, the ontology which is to be used, and information about the dialogue that the message is part of. Also reviewed are Knowledge Interchange Format (KIF), which is an interlingua for ontologies, and Cyc.

The article (Genesereth and Ketchpel, 1994) is also about agent communication, but specially focusses on trying to set a standard for communication between heterogeneous applications. This is done using a universal language, Agent Communication Language (ACL). ACL is made up out of a lexicon, which may be selected from one out of multiple ontologies, the language KIF, and a communication protocol using KIF which is similar to KQML.

The article (Barbuceanu and Fox, 1996) discusses COOL, an agent implementation language which enables KQML-like communication, and cooperation and conflict management.

DESIRE (DEsign and Specification of Interactive REasoning components) is described in (Brazier et al., 1995) and (Brazier et al., 1994). DESIRE is a design framework, complete with diagram notation and executable specification language. The framework is based on the 'shared task model', i.e. a model of the (design) task that can be understood by all parties.

The task knowledge includes: information about the task structure, which should be hierarchical, with complex tasks (tasks which can be subdivided) and primitive tasks (which cannot be subdivided further), the knowledge structures, the necessary information exchange between the tasks, subtask sequencing constraints, task delegation constraints (specifying which agent may do what).

This task knowledge can be specified in the specification language. Each task is mapped 1-to-1 to a component. As is the case with tasks, there are composed and primitive components. Composed components contain: input/output interface (the interface to components outside itself), task control structure (control flow constraints of the (sub)tasks known by the component), the subtasks and their information links, and domain knowledge. Primitive tasks may be implemented using conventional software.

Each component is specified in predicate logic, using sorts, predicates (relations), functions, and constants. In order to allow flexible task allocation, information about task-knowledge allocation and task allocation (which agent can and will do what) can also be specified. Between the components there are information links. Each link links the truth value of one component's atom to one of another component's atom. The mapping is changeable using a translation table. Component-environment and component-human interaction can also be described by viewing resp. the environment or human as a separate component.

The specification can be executed. The reasoning engine of each component entails object reasoning (reasoning about states of variables), meta-reasoning, meta-meta-reasoning, etc. using the specifications given.

There's no complex negotiation between components built into DESIRE; communication means sending or waiting for raw data of a specific type over each specific link. Agents with complex negotiation will have to be implemented by hand, for example by implementing a special kind of composed component (Brazier et al., 1995).

4.3 Natural language

Language allows communication to others, but language also seems to be closely related to human thought processes, as advocated by cognitive science, see section 3. This seems to be supported by experimental evidence from HCI, for example, there appears to be interference of language use with thought (Ericsson and Simon, 1980). Thus, dialogue can be seen as

collective thinking (Allwood, 1995).

4.3.1 Underspecification, redundancy and error in language

According to (Levinson, 1987), people understate and oversuppose. If anything turns out to be unclear, this will generally be detected later, and can be explained afterwards. Note that it may be a significant problem finding out just what was and was not understood. For NL systems, this means that good dialogue managing may compensate for bad speech or language understanding (Fraser, 1995).

On the other hand, language contains a great degree of redundancy. It is often assumed that redundancy is meant to provide more robust communication across everyday's noisy environment. The noise may sneak in at any level of communication: examples are speech understanding trouble and misunderstood shades of meaning. This would also mean that more efficient communication can and will probably occur in cases when there is less noise. For example, when explaining something complex, it is probably best to define each word you use precisely, and use very strict and unambiguous syntax, and while talking about something routine or within a clear context, it is possible to skip a lot of words and generally be unsyntactic.

5 Human Computer Interaction

An overview of HCI is given in (Shneiderman, 1998) and in (Veer and Lenting, 1995). Overviews typically include models of human-computer systems, classifications of types of user interfaces, and methods and guidelines to improve usability of a computer system. For an example of user interface (UI) design recommendations, see (Roe and Arnold, 1988).

The article (Haan et al., 1991) summarises possible strategies to improve usability, which are: user cooperation in system design, prototyping, using analogies or metaphors, using guidelines or standards, or formal task-goal modelling coupled with metrics.

Generally, what is usually called 'HCI' focusses on more 'classical' user interfaces rather than multimodal and NL interfaces. This is a problem, because much of it is based on experimental findings, which cannot always be generalised easily. Still, it would be useful to understand the relationship between these different kinds of interaction. For an overview of speech and NL research topics in relation to HCI, see for example (Baecker et al., 1995). For an introduction to dialogue system design, see (Bernsen et al., 1998). For articles on NL in HCI, see for example (Aust and Oerder, 1995). For some examples of models and theories of semantics and pragmatics in human-computer (and human-human) dialogue, see the workshop proceedings (Hulstijn and Nijholt, 1998).

5.1 General criteria that determine usability

Well-known in the field of usability of computer systems is Jakob Nielsen (see for example (Nielsen, 1993)). For Nielsen, usability is part of 'usefulness' which is in turn part of a bigger system acceptability scheme, distinguishing 'social acceptability' and 'practical acceptability'. Usability is classified into: easy to learn, efficient to use, easy to remember, few errors, and subjectively pleasing. Note the distinction between learning and remembering: learning includes learning the general concept of the system, while remembering is only remembering the fussy details, while the general concept is already known. The criterium 'easy to remember' is therefore especially important for casual users.

The criteria most often found in other literature are effectiveness, efficiency, satisfaction and learnability (Jones and Galliers, 1996). Variations on this theme exist. For example, there is Relevance, Efficiency, Attitude, Learnability (REAL) (Ek, 1997) and functional, easy to use, learnable, pleasant to use (Haan et al., 1991). Note that the aspect effective/relevant/functional is not included in Nielsen's view of usability. For Nielsen, this is part of a more general aspect, usefulness. Note also that error is no longer a separate category.

The article (Carlson and Hall, 1993) tries to determine a finer-grained covering set of criteria by means of a review of existing literature. These are user performance (degree of success or number of mistakes or time spent in error recovery), reliability (system does what user intends), stability (system guards against damaging actions or allows easy recovery), power (amount of functionality and number of different ways to achieve something), speed (speed in executing the user's commands), efficiency (speed with which the user can control the system), effectiveness (time taken by the user to accomplish a task). The tradeoff relations between the criteria are explained. The criteria consistency, flexibility, ease of use, ease of learning are argued to be contained in the others. However, the argument for ease of learning is not quite convincing: it is more than the combination of reliable, stable and effective, because these do not imply a good learning curve. The satisfaction/attitude criterium is not accounted for. Note that consistency is usually not seen as a general usability criterium but as one of the means to achieve usability, see section 5.2.

5.2 Separation of interface and the computer's internals: the UVM

Users, and often the interface designers themselves, typically don't want to concern themselves with the details of the underlying system. Rather, some HCI designers like to talk about a user's conception of the system as fully separate from the system, and have a separate model for it. This is called the conceptual model or User's Virtual Machine (UVM). This is the view that a person has or should have of the system he/she is using. The UVM can be designed independently from the actual system according to ergonomic principles.

One of the most important principles for UVM design is consistency. Consistency means that the different actions and representations in the UVM are somehow analogous to other actions and representations. This should make the system easier for people to learn, understand, and remember. Note that 'analogous' is in principle a highly intuitive concept, but its meaning is often quite clear in practice. Example: In a command-line system, it would be consistent to have the source operand always in front of the destination operand, and the command line switches always in front of the operands. One can imagine that a command-line system in which the order of these operands is different for each command, is much harder to use. Analogy can refer to both the relation between different features within the UVM, or to the UVM in relation to everyday life. The last form of analogy is called metaphor.

However, care should be taken that the actual system will be able to comply to the UVM. Often, it turns out that it is not fully possible to hide the system internals from the user. This may happen in the following situations (note that this classification is my own):

- When the UVM is not watertight, and some of the internals show through in unexpected places, presenting a particularly inconsistent system to the user.
- When it turns out that functionality is required that is quite possible using available tools, but is not within the part of the interface that is designed according to the UVM. Even while it may be consistent, a beginner-friendly UVM usually covers only a subset of functionality. The user has to learn how the system really works before being able to understand the special tools.

- When things go wrong. Either internal messages suddenly come pouring out, or the nature of the system errors, or even the errors themselves, are hidden. The user is baffled, but does not get the information required to take actions to repair the error. This problem is addressed in more detail in (Lewis and Norman, 1986).

One may expect that these problems get worse when the UVM is farther away from the real system, or when the possible user tasks are more complex. One way to prevent them would be to design the UVM carefully, taking into account the system internals as well.

5.2.1 Seeheim-like models

This is a class of HCI models which model human-computer interaction as two-party communication, described as flows of specific kinds and levels of information between software components and the human. Components typically include presentation, domain, and interaction handlers. One of the first and best-known is the Seeheim model–hence the name of this section. This model, along with some others, is described in (Encarnacão, 1997). The Seeheim model basically consists of four components, connected serially as follows: the human, the presentation component (lexical level), the dialogue component (syntactic level), the application interface (semantic level), and the computer program. The other models are rather similar to the Seeheim model, but are more elaborate. They include: two versions of the arch model (this model, specifically meant for modelling adaptive user interfaces, has similar components, represented as an arch bridging between the user and the system), and the triple agent model (which consists of three components: the user, the user discourse machine, and the task machine). The purpose of such models is usually to model proper distinction between the UVM and the underlying system. Some of these also model the user in terms of the same kind of components as the computer. This more symmetrical view is closer to agent models, discussed in section 4.2.1. The difference is that they always model a single human and a computer party, and that they do not explicitly refer to the concept of agent as used in typical agent models.

The article (Waugh and Taylor, 1995) introduces a layered feedback model of communication. In this model, three theories are examined and brought together: Norman's human-computer hierarchical feedback theory (goal-intention-action-feedback-evaluation), Layered Protocol (LP) (a two-party cooperation model like Norman's feedback theory but with parallellism and information integration within each party and 'virtual communication' between similar levels of the two parties ('real' communication only occurs at the lowest, physical, level)), Perceptual Control Theory (PCT) (a human is modelled using a hierarchy of Elementary Control Systems, each of which stands for an activity related to other activities by a task-subtask hierarchy).

In (Diel et al., 1993), an object-oriented UI specification language is described, with the main goal of separating program from user interface. The UI is specified by specifying objects out of four main types: Abstract (modality-independent), Data (specifies data structures), View (which of the items of the Abstract object should be viewed), Presentation (implementation of view). The bulk of the interaction should be implemented in the Presentation objects. Some related models are mentioned also. These are Seeheim, Model View Controller, PAC, Tube UIMS, ScreenView, MacApp, InterView.

5.3 Cognitive psychology in HCI

Cognitive psychology and HCI are much related, but often, the contribution of cognitive psychology to HCI design is implicit or indirect. Sometimes however, cognitive psychology

concepts are translated directly to HCI guidelines. Some examples of the latter case are given here.

In (Nielsen, 1993) for example, it is argued that gestalt theory should be applied to design layouts, memory load should be minimised, and the frequency and consequences of slips should be minimised by good interaction design. In 'chunking and phrasing in HCI' (Buxton, 1986), the concept of chunking is translated to: one should design possible actions in the interface in such a way, that they correspond one-to-one to chunks in an expert's memory. This enables quicker learning. In 'designing for error' (Lewis and Norman, 1986), the distinction between slips and mistakes goes parallel with a distinction of feedback needed to enable the user to recover from an error.

5.4 Cinematography in HCI

Some people argue that ideas from cinematography and theatre can effectively be used for HCI. These ideas can be divided into two kinds: adapting the design paradigm, or adapting practical principles, such as filming techniques.

The book (Laurel, 1993) argues for adopting the design paradigm of structuralist theory of theatre. Basically, the human and computer (or rather, representations of 'agents' by the computer) are seen as analogous to agents participating in a play. One of the main points made is that the traditional use of metaphor is insufficient for designing easy-to-use systems, because human-computer activity is necessarily different from real-life activity. The alternative would be to design the metaphor as a deliberately and carefully simplified and modified caricature of reality. Another main point can be seen in the fact, that the word 'human-computer interaction' is deliberately replaced by 'human-computer activity' throughout the book. The main purpose of this is apparently to advocate an integrated view, in which the human is supposed to feel part of the action, rather than feel that the computer is something separate which has to be communicated with (as is stated in the book, 'see the computer as a medium, not as a tool'). More detailed adoptations of theatre theory include the notions of formal cause (form), material cause (the materials used), efficient cause (the way in which they are implemented by the actors) and end cause (the purpose); probability and possibility; complication and resolution; discovery, surprise, and reversal; the effect of constraints, etc. However, the examples given are not quite concrete enough to show how this theory is an actual improvement of current practice in HCI development, nor are the illustrations of the 'failures' of traditional HCI always quite accurate.

An example of the use of practical principles of cinematography in HCI can be found in (May and Barnard, 1995). The main idea pointed out in this article is: make sure of cognitive congruence. Two examples are given: a new object that should be in focus should start at the same screen position as the previous object that was in focus, and continuous sound should remain continuous during change of viewpoint.

5.5 Semiotics in HCI

Some material exists that tries to couple semiotic theory with HCI, for example (Andersen, 1990), (Uzilevsky, 1994), and (Ehrich and Williges, 1986) chapter 5. Possibly, the concept of 'signs' may aid in forming a more general understanding of, and relations between human cognition, HCI, NL, and multimodal interaction.

In (Callahan, 1994), the relevance of the sign type classifications icon, index, and symbol for HCI is addressed. It is argued that the choice of sign type in a user interface (for example, direct manipulation is mostly iconic, command line is mostly symbolic) influences the kinds

of interaction that are possible or feasible. For example, a point is made in favour of using symbols (words) rather than icons (pictures), because if you give something a name, you may give the user an opportunity to talk about it, using the superior techniques that symbolic communication enables.

Part II
Systems

6 Systems

6.1 Purpose

Many HCI analyses and models are based on the assumption that there is some concrete task to be achieved or problem to be solved (task-oriented or cooperative problem solving models), while some focus on vaguer tasks, like advisory, exploratory, and learning tasks. (Walker and Whittaker, 1990) contrasts properties of advisory dialogues (AD) and task-oriented dialogues (TOD).

Some task-oriented NL dialogue systems can be found in (Fischer and Reeves, 1991), (Sikorski and Allen, 1996), and (Stent and Allen, 1997).

Relatively often however, NL, multimodal, and 'intelligent' interfaces are used for the classes of less well-defined tasks.

One is evaluation-oriented information provision (Jameson et al., 1995), which means information provision specifically meant to assess a certain situation, such as personal assistants or sales assistants. Another well-known example is expert systems (Fass et al., 1996).

Often found is information retrieval (IR) (Bouzeghoub and Metais, 1995). (Androutsopoulos et al., 1995) gives an overview of existing IR systems. A more specific domain is NL database query (NLDB) system for network management (Chau, 1993). This article also lists general advantages and disadvantages of NLDB: Advantages are adaptation to user domain knowledge, no learning time and remembering commands, and contextual ability. Disadvantages are: the imperfection of NLP technology, because there are hard-to-avoid ambiguities at all levels: unambiguous generated language looks awkward, and users always make (grammatical) mistakes. Another is users' overestimation of system capabilities.

In educational research, much material concerns instructional dialogues. (Beun et al., 1995) contrasts instructional with informational dialogues. In instructional dialogues, the tutor typically provides information and then asks questions about the subject matter, after which a dialogue may occur. Unlike informational dialogues, the first information is given spontaneously, i.e. without a query, and questions are asked by the one who gave the information (the tutor), because the tutee usually doesn't react spontaneously. So, the tutor usually takes initiative even while the information to be conveyed is usually complex. The student's current knowledge and learning abilities will have to be understood well by the tutor. A particularly subtle technique in instruction is pedagogically motivated misrepresentation (PMM). (Gutwin and McCalla, 1992) explains how and when PMMs are used, and a system that uses them is presented. More specific examples of instructional dialogue are: design instruction (Lee, 1995), optics instruction (Reiner, 1995).

6.1.1 Cooperativeness

In some systems, and especially in NL systems, cooperativity is explicitly modelled. For a taxonomy of cooperative response types, see (Yamada et al., 1993).

Much of the basic 'cooperative answers' as referred to in NL (and sometimes command line) systems can be seen as equivalent to browsing in GUIs. In GUIs, data that is related to the data requested is displayed on screen while the user is selecting the request. For example,

users select a specific entry out of a list of files, emails, or icons. If the user is not quite sure what to select he/she is enabled to browse. The linguistic equivalent of browsing is the computer coming up with data that may be useful in response to a not-quite-valid or imprecise request. In other words, cooperativeness is the same as ensuring visibility. See for example (Wolf et al., 1995), where a voice-driven email system was tested and was compared to its graphical equivalent. 'Visibility' is shown to be an issue in the voice-driven variant: one has to: 1. make sure the user knows what the computer has understood, 2. give cues as to where the user is in the interface, 3. for larger mailboxes, query commands plus cooperative answers rather than browse commands would be more appropriate.

There is very little research on uncooperative dialogue systems: the most notable example is PRACMA (Jameson et al., 1994), which tries to maintain discourse obligations while having conflicting goals with the user (Jameson and Weis, 1995). The example application domain is second-hand car sales.

6.2 Overview of existing systems

In this section, an overview of some major existing NL and multimodal projects and systems is given. Other such overviews exist. In (Androutsopoulos et al., 1995), an overview of dialogue systems for IR is given. In (Smith and Hipp, 1994), an overview of NL dialogue systems is given.

The systems are summarised in the table below. The classification of the systems is divided into application domain, technical abilities, and the strategies and styles used. Most systems use a NL strategy, all systems use at least a NL style (see section 7 for definition of strategy and style).

Abilities:

NLP Natural language parsing

DIS Discourse and context tracking, and negotiation

GES Gesture recognition

SPR Speech recognition

IMM Integration of multiple modalities (or sensory channels, for more detailed information see section 7), either in the form of recognition or generation.

Strategies and styles:

PNT Pointing, including menus, tables, buttons, maps, drag-n-drop

VIR 3D, virtual reality

MUL Support for multiple participants

System, name and reference	Domain	Abilities					Styles		
		NLP	DIS	GES	SPR	IMM	PNT	VIR	MUL
Schisma: (Nijholt et al., 1998) (Lie et al., 1998)	theatre booking	x	x				x	x	
Dialogos: (Albesano et al., 1997)	railway enquiry	x	x		x				
TRAINS93: (Sikorski and Allen, 1996), (Stent and Allen, 1997)	route planning	x	x		x				
ARTIMIS: (Sadek et al., 1997) (Bretier and Sadek, 1997)	telephone enquiry	x	x		x				
PADIS: (Bouwman, 1998)	telephone enquiry	x	x		x				
COSMA: (Busemann et al., 1997)	appointment scheduling	x	x						x
CASSY: (Gerlach and Onken, 1993)	pilot assistance		x		x		x		
WOMBAT: (Blandford, 1995)	purchase planning		x				x		
CARTOON: (Martin, 1997)	map enquiry				x	x	x		
InterLACE: (Trafton et al., 1997)	map enquiry	x				x	x		
ORIMUHS$_1$ (Encarnacão, 1997)	intelligent help		x		x	x	x		
QuickSet(Johnston et al., 1997)	war tactics		x	x	x	x	x		
Circuit fix-it shop (Smith and Hipp, 1994)	circuit repair	x	x		x				

Schisma ((Nijholt et al., 1998), more specific information about the parser can be found in (Lie et al., 1998)) is a theatre inquiry and booking system. Its language parser is based on 'rewrite-and-understand', which means any language utterance is rewritten, using production rules, into a 'canonical' form. The dialogue manager is based on resolving unfilled entries after a transaction is initiated. Schisma is currently being integrated with the Virtual Music Centre (VMC), which is a virtual-reality version of the music centre building in Enschede.

Dialogos (Albesano et al., 1997) is a task-oriented railway-enquiry telephone speech system which has been tested with a large user base.

TRAINS93 (Sikorski and Allen, 1996), (Stent and Allen, 1997) is a system for route planning of cargo trains. TRAINS93's dialogue manager (Traum, 1997) is described as a reactive-deliberative dialogue agent. Modelled are social attitudes: mutual belief (grounding), obligations, and multi-agent plan execution. Formal descriptions of intention, commitment, planning and plan execution, goal satisfaction, failure, and repair are given.

ARTIMIS (Sadek et al., 1997) (Bretier and Sadek, 1997) is a telephone inquiry system. It is plan-based, and is described as a rational communicating agent. It is reputed to be a good system.

PADIS (Philips Automatic Directory Information System) (Bouwman, 1998) is a voice dialling system, used as a test-bed for HDDL (High Level Dialogue Definition Language), developed at Philips.

COSMA (Busemann et al., 1997) is a NL front-end for appointment scheduling applications. Uses the InterRAP agent architecture as a basis. The program has to communicate with applications and multiple users.

CASSY (Cockpit ASsistant SYstem) (Gerlach and Onken, 1993) tries to assist pilots in flight planning and decision tasks. Its ultimate goal is to reduce error in the human decision-making process.

WOMBAT (Blandford, 1995) is a decision making tutor/assistant ('agent') system, which actually 'abstracts' from using full natural language, but instead, focusses on problem solving and argumentation. Each party chooses from a set of canned sentences with parameters that can be filled in at specific places. The system models user goals and beliefs and knowledge

about the situation. It decides what to say next by weighing the different options against a Gricean-like priority-system (though the rules are somewhat more tailored to the 'tutor' purpose). The system generates reasonable replies but does not support complex argumentation.

CARTOON (CARTography and cOOperatioN between modalities) (Martin, 1997) is a system to assist in obtaining information from a map. The system attempts modal integration of user input, which comes in the form of NL and pointing.

InterLACE: (Trafton et al., 1997) is a multimodal system, combining pointing on a map with typed text. It includes a full natural language engine.

ORIMUHS$_1$ (Encarnacão, 1997) is a generic intelligent help and support system for users of complex software. It keeps a user, task, and discourse model to generate multimedia help. It can also be given commands using speech, keyboard, and mouse. It is implemented as a server, serving multiple individual users at the same time.

QuickSet (Johnston et al., 1997) is an assistant for planning war tactics. The user can supply speech combined with gestures on a map. The gestures recognised are out of a limited set of symbols, such as fortified line, area, tank platoon and mortar. The input from the different modalities is unified to form a feature structure. This is called unification-based multimodal integration.

The circuit fix-it shop is described in detail in the book (Smith and Hipp, 1994), along with the underlying theories and ideas, and evaluations of the system. The application domain is highly task-oriented, and consists of diagnosing and repairing an electronic circuit. The user is capable of doing the physical actions to repair the circuit, while the computer knows what to do. It is plan-based, assuming a single shared goal, and dynamically generating the best next step, based on the current situation. An inference engine, combined with missing-axiom theory, is the central mechanism for driving the dialogue. It is used to: determine the next step in the task, keep track of user knowledge, and model dialogue focus. The system allows mixed-initiative by distinguishing four modes, which augment the working of the central engine. The modes are: directive (computer has control), suggestive (computer suggests but user may change the course of the task or dialogue), declarative (the user has control but the computer mentions relevant facts), and passive (user has control).

7 Dialogue styles and strategies

Classifications of dialogue strategies in HCI can be found in many HCI overview books. One classification that is recommended often is the one by Shneiderman (Shneiderman, 1998). A similar one can be found in (Veer and Lenting, 1995).

7.1 Mode, style, and strategy

Mode and style are sometimes used to indicate classification of surface features of interaction, though the precise meaning varies. Indexical, iconic, and symbolic are sometimes called modes (Callahan, 1994). Others call 'interactive' and 'noninteractive' modes. (Oviatt and Cohen, 1989), for example discusses the effect of interactiveness on verbal explanations.

A classification of user interface types often found is: conversational, direct-manipulation, command-line, question answering, menu choosing, and form-filling. In (Shneiderman, 1998), they are called styles, while in (Veer and Lenting, 1995), they are called 'paradigms'.

In (Androutsopoulos et al., 1995) (page 7), the advantages of NL are contrasted with those of other interface styles. Some interesting advantages are given: NL is better for some questions,

which would require lenghty notation in other languages, NL discourse has context, which allows briefer queries. An apparently similar contrast, namely conversational versus direct-manipulation, is given in (Stein and Maier, 1995). However, here it is argued that the conversational style can also be used in conventional GUIs. The GUI example given in this article even uses a dialogue grammar.

In an attempt to get rid of the confusion, a terminology will be defined here. We will not use the word 'mode'. We will indicate the use of a particular (static) representation that is used to communicate a state of the system to the user and vice-versa with the word 'style', while we indicate the possible system dynamics, when viewed as abstract system states and possible transitions between them, with the word 'strategy'. In this view, the two discussions summarised in the previous paragraph are really talking about different things: while both are talking about dialogue strategy and style as found in dialogue-grammar models of human-human conversation, (Androutsopoulos et al., 1995) is talking about natural-language strategy, while (Stein and Maier, 1995) is talking about natural-language style combined with GUI strategies.

7.2 Syntactic

7.2.1 Modality

What is generally meant by 'a modality' is a specific channel through which communication can be conveyed. Typically, multiple channels may be identified, each with its own distinct properties, and each, in some way, separate from the others. The first question that comes to mind is: what channels are there? The answer probably depends on your viewpoint: in what way are the channels meaningfully distinct and separate? We will not take one viewpoint here, but instead, show some of the options available.

If we look at human perception, the most 'objective' answer is probably that a modality corresponds to a human sensory and motoric subsystem, like the eyes, ears, hands, and voice. When we consider a 'deeper' cognitive level, we might also say that, for example, people think differently about written language than they do about graphs, even though both are visual, suggesting another classification of modalities.

On the other side, if we are considering technological issues, we might view modalities as corresponding to the computer's input and output subsystems, like screen, speaker, keyboard, mouse, and microphone. A classification according to a 'deeper' level of technological issues could also be made, for example, continuous speech versus isolated-word speech, or screen versus paper.

After having recognised the existence of modalities, some issues arise:

- **Choosing between modalities.** According to the situation, one modality may be more efficient or comprehensive than another.
- **Integration of modalities.** In what way is or should the data arriving from the different modalities be merged into coherent information?

A classification of the ways in which combinations of modalities can be used to convey information is found in (Martin, 1997). He identifies equivalence (modalities can convey the same information), specialisation (a modality is used for a specific subset of the information), redundancy (information conveyed in modalities overlaps), complementarity (information from different modalities has to be integrated to arrive at coherent information), transfer (information from one modality is transferred to another), concurrency (information from different modalities is not related, but merely speeds up interaction).

(Bernsen, 1996) presents *modality theory*, which classifies modalities according to both human and computer issues, for example, written text, typed text, typed keywords, continuous speech, and isolated-word speech are different modalities. Properties of each modality are given, so that the effects of a certain choice of modality or modalities can be predicted. The emphasis of the article lies on the speech modalities, and on when to choose for or against using speech input or output. Another article contrasting advantages and disadvantages of voice to other modalities is (Cohen and Oviatt, 1995).

Using multiple modalities coherently in some way is called multimodality. (Nagao and Takeuchi, 1994) gives a classification of the multimodal interfaces typically encountered:

1. NL and direct manipulation,
2. NL and contextual feedback,
3. NL combined with facial expression.

In the general case, multimodality helps to decrease ambiguity by supplying additional context. The article itself addresses an example of the third kind. Examples of the first and second kinds are given below.

An example of integrating gestures and natural language can be found in (Fahnrich and Hanne, 1993). Another can be found in (Johnston et al., 1997), which is called unification-based multimodal integration. This approach uses feature structures in which drawings are incorporated as concepts. The system QuickSet, which is also described, illustrates the approach. (Lee, 1995) concerns the feasibility of combining graphics and NL in design instruction. Design activities are argued to be found as part of all kinds of problem-solving activities. Design is argued to be impossible to explain and has to be learnt while doing. So, a 'design studio', an environment for designing and learning to design, is argued for. Such a design studio may be implemented on a computer. An analysis is given of human-human interior design dialogues aided by drawings. The messages conveyed in the drawings are sometimes quite subtle, and go beyond the usage of graphical symbols out of a limited symbol set, as is assumed by a lot of gesture- and drawing-interpreting computer systems.

An example of the second class may be found in(Nagao and Rekimoto, 1995). Here, the context is some nearby object. The interface is meant to act as if the user talks to the object directly.

(Dybkjaer et al., 1995) addresses the effect that choice of representation modality has on the language use of the user.

An example of modalities at a 'deeper' cognitive level is given in (Reiner, 1995), which discusses the use of different kinds of symbols in an optics learning environment. Disciplines like physics and optics use a non-everyday set of symbols and expressions. It is argued that symbols ('labels') can be learned best by using them in communication. It is studied how such means of expression can be acquired in a collaborative-learning setting, using a computer tool to construct optics models. Four kinds of symbols are studied during the learning session: graphical (using the program), mathematical, verbal, and physical (using real objects). The introduction of new symbols in the course of the dialogue are plotted against time. Some requirement relations between symbols are shown. The students appear to prefer graphical representations.

Using multiple output modalities is called multimedia. See for example (Andre and Rist, 1995) and (Sutcliffe and Faraday, 1993). Generating multimedia presentations from task plans using a generalised text-discourse generation method is discussed in (Rist and Andre, 1993). A cognitive-walkthrough method for designing multimedia is discussed in (Faraday

and Sutcliffe, 1993). Their underlying theory is that visual and text objects are interpreted to form respectively objects and propositions, which are then integrated using rules from LTM into 'macro-propositions', which are mode-independent. From there, a sequence of 'macro structures' is formed, representing the discourse. The method is as follows: fist, make a task analysis, then evaluate user attention, topic focus, argument repetition (is a concept repeated at the right times and consistently?), and macro-proposition and macro structure forming (do the propositions make sense?).

7.3 Semantic

(Rauterberg, 1993) proposes a semantic description of user interfaces, which can be used to derive measures of user interface properties, such as interactive directness, feedback, flexibility of dialog interface, flexibility of application interface. The main idea behind the description is the identification of 'object space', 'function space' and 'functional interaction points'. Note: history is not accounted for in the model. For example, command-line interfaces should have scored higher on feedback because they explicitly show history.

7.3.1 Redundancy and confirmation

The relation of redundancy with mutual belief is discussed in (Walker, 1992). It is argued that informationally redundant utterances (IRUs) often occur in human-human dialogue. IRUs are meant to make inferences explicit. For example, paraphrases occur often and are meant to represent the conclusion that the hearer has drawn. (Walker, 1994) and (Walker, 1996a) go on where the previous article left off, and discuss the relation between reasoning and redundancy. It is argued that the occurrence of IRUs is also related to the trade-off between cost of retrieval for the hearer and cost of communication for the speaker. This contrasts with the assumption of omniscient parties.

(Smith, 1997) evaluates strategies for selection of utterance verification in case of speech understanding uncertainty in the Circuit Fix-It Shop. It is not efficient or user-friendly to verify everything, so, selections are made according to 'parse cost' (the unlikeliness that an insertion, substitution, or deletion of a recognised word could have happened) and 'expectation cost' (the unlikeliness that a specific semantic frame could occur given the previous utterances). Experimentally, a combination of both proves best. Introduction of a variable, topic-dependent, verification threshold is shown to make little difference. The remaining bottleneck appears to be misrecognition of content words (like one digit for another, or one name for another).

(Trabelsi et al., 1993) discusses heuristics for generating informative responses to failing queries in NLDB. The kinds of failures are classified into: value (there were no matches for a specific value of an attribute), condition (no matches for value in current context), attribute (attribute does not exist), concept (concept or entity type is not known). Basically, the solution to each is to state clearly the reason why the query failed. Repair is done using lists of options to be selected until a complete query is specified or the user decides that the option wanted is not present.

7.4 Pragmatic

Several varieties of plan-based models exist: one can model one joint plan, one plan per agent, or multiple (tentative) plans for a single agent. (Ramshaw, 1991) discusses a plan exploration model. This allows multiple and hypothetical plans. It has three levels: domain

level, exploration level, and discourse level. (Jameson and Weis, 1995) discusses discourse obligations in noncooperative dialogues, which implies non-shared plans.

7.4.1 Initiative

Initiative, or control, refers to who is taking control of the interaction. Typically distinguished are mixed-, user-, and system-initiative. (Kitano and Ess-Dykema, 1991) discusses a plan-based understanding model for mixed-initiative dialogues. The model proposed allows non-shared domain plans.

Mixed versus system initiative in NL dialogues is examined experimentally in (Walker et al., 1997a). One particularly striking remark made here was that people found the pace of the system-initiative alternative faster, even though the mixed-initiative was actually quicker. A reference was made to similar findings in the case of GUIs (Smith, 1996).

7.4.2 Modelling of dialogue structure

Dialogue structure refers to the discourse structure of one participant, and to initiative and initiative shifts between participants. (Passoneau, 1989) discusses discourse structure in relation to the discourse referents 'it' and 'that'. Theory about the function of the two pronouns is given. The article concludes with a set of rules relating pronoun to discourse center, center retention, and antecedent. (Walker and Whittaker, 1990) discusses topic centering in relation to mixed initiative.

(Karsenty, 1993) models interactive explanations using rhetorical schemas. Human-human design dialogues and draft pictures were recorded and analysed using rhetorical schemas. It was found that rhetorical schemas only explain dialogue goals, and not task goals.

The relation between communication goals and feedback is discussed in (Nivre, 1995). Communication goals are classified into a three-level hierarchy: evocative (pragmatic), signalling (semantic), utterance (syntactic). Feedback may concern each of these goals, and may be positive (OK), neutral (inconclusive), negative (failed). Some combinations are not possible: negative feedback on a lower goal implies negative feedback on a higher goal. The converse goes for positive feedback. Veridical feedback is any feedback received; intentional feedback is explicit feedback by the other participant.

(Bunt, 1995) argues for transparency and naturalness as the key concepts for successful interaction. To achieve this, it is argued that precise modelling of what is going on in a dialogue is needed. Presented is Dynamic Interpretation Theory: the contextual aspects covered are linguistic, semantic, physical, social, and cognitive context. Dialogue acts (one utterance may consist of multiple acts) are classified into dialogue control and task-oriented dialogue acts. A classification of acts is given.

In (Tillmann and Tischer, 1995), self-repair disfluencies (hesitations and repetitions) are measured in different control circumstances, which are using a button to shift control, using normal conversation, and using normal conversation without a given problem to solve.

(Heeman and Allen, 1997) analyses speech to determine intonational boundaries (segment the speech into phrases), speech repairs (self-repairs), discourse markers (these influence the structure of the discourse) . They argue for the need of an integrated analysis, integrating these three kinds of information.

In plan-based models, plans may include discourse plans next to task plans. For example, (Moore and Paris, 1989) discusses text planning in advisory dialogue. A distinction is made between intentional and attentional structure. A scheme is explained with which the system

is able to take task goals, text goals and text structure into account for generating text. (Lambert and Carberry, 1991) discusses another plan-based model of dialogue, which has three plan levels: domain level, problem solving level, and discourse level. (Lambert and Carberry, 1992) continues on the previous model, which is extended for modelling conflicting beliefs and negotiating about such conflicts, thus enabling negotiation subdialogues. Another approach is found in(Kitano and Ess-Dykema, 1991), a plan-based understanding model for mixed-initiative dialogues. This particular model also allows non-shared domain plans.

Another often-found approach to model the effect of dialogue utterances to the dialogue structure is the dialogue grammar approach. Sometimes, the dialogue grammar approach is meant merely to give some extra cues to predict control shifts. The types of utterances identified have at least the distinction question–assertion. In (Jönsson, 1993), this approach is compared to the plan-based approach. It is argued that, in practice, the dialogue grammar approach works as well as the plan-based approach. Shifts in initiative are usually modelled either by means of a finite state automaton or a hierarchy of dialogues and subdialogues. (Walker, 1996b) contrasts a linear with a hierarchical model of control shifts. Since old utterances are forgotten, it is argued that control shifts are never completely hierarchical.

(Walker and Whittaker, 1990) discusses the relation between centering and conversation control in advisory dialogues (ADs) and task-oriented dialogues (TODs). The utterance types that are identified are assertions, commands, questions, and prompts. Control shift types that are identified are abdication, summary, and interruption. Control shifts are either persistent or temporary. In the case of temporary control shifts, the control is relinquished as soon as possible. Possibly, these interruptions have a hierarchical structure. Apparently, interruptions happen when there are problems with either the information quality or the plan quality. It is shown that control-shifts have a specific influence on references. Also, control shifts happen more often in ADs, especially summaries and abdications occur more often in ADs.

(Chu-Carroll and Brown, 1997) discusses prediction of initiative in collaborative (planning) dialogues. A summary of previous work on dialogue initiative is given. It is argued that task initiative and dialogue initiative have to be separated. A predictive-cue approach, which is similar to the dialogue grammar approach, is proposed, obtained after annotating and analysing TRAINS91 dialogues. Nine cue types, classified into three classes, are used to predict initiative.

(Iwadera et al., 1995) divides a dialogue in components (acts or utterances) which each have a type, and can be combined into higher-level structures, called moves and exchanges. The theory classifies topics into short-term and long-term. The scope of a topic is determined by the act-move-exchange-structure. Note: it is not clear how the theory deals with sudden interruptions or changes of topic.

8 The development process

8.1 Description

There are soms studies that try to analyse the processes that developers (analysts, designers, programmers) go through (Schooten, 1997). The following general observations are particularly relevant:

- In typical development activities, developers should be able to choose what they want to see and what they want to hide. This can range from something as simple as being able to temporarily replace a part of the design by a single box or piece of text, to being able to view a design from different complementary viewpoints.

- Experts on one field have problems understanding other fields. In HCI, the classical example is HCI analysts versus software developers.

8.2 Prescription

We will define a *methodology* as a stepwise prescription of development activities. Methodologies may have various levels of detail and rigidity. For example, specific activities and/or specific kinds of descriptions may be prescribed for each step. Various low-detail and general examples can be found in Kaaniche and Mazet's lecture in (Pasquini et al., 1998), some of them prescribing a rather complex sequence of steps. Most often found, however, is the 'design cycle' that goes something like: analysis - design - implementation - evaluation - analysis - ..... The purpose of going through the stages systematically is to be able to gain maximum insight needed to develop a successful system. Usually, one starts with the analysis stage, analysing the system that is already present and has to be rebuilt, but in some methods, one may jump in at any stage. Some variants are not cyclic, but finish at the evaluation stage, assuming the problems found in evaluation are small enough to be fixed on the fly, without requiring a reiteration. Some methods lump together the design and implementation stage into a single stage, while others lump together the analysis and evaluation stage.

> *Analysis* is finding out things about the current situation, which includes existing system(s), task(s), and environment.
>
> *Design* is describing a system. The description may be in the form of a specification of constraints the system must conform to (this is usually called *specification*) or a first description of the system to be built. Design is sometimes split into the separate stages specification and descriptive design.
>
> *Implementation* is design that leads to an executable specification.
>
> *Evaluation* is analysis with the purpose of determining whether a (new or old) system fits its purpose. This may mean evaluation of the implementation and the description against the specification (testing or verification), or of the working system in its real environment.

Why use a development methodology? (Tullis, 1993) shows that a naive choice of interface strategy may not be the best one. In their experiment, naive designers turn out to be fond of 'drag and drop', while evaluation points out that it is one of the slowest to use, nor is it the strategy that the users prefer. Also shown in this study is a positive correlation between user's preference and efficiency.

The EAGLES handbook (Gibbon et al., 1997) proposes guidelines for specification, design, and assessment (which means evaluation) of NL and spoken dialogue systems, as well as practical tips. The book identifies and addresses three different kinds of dialogue systems: menu, spoken, and multimodal. These are also contrasted with command systems. The book gives some general recommendations for dialogue systems, as well as methodological recommendations for building such systems. Possible purposes of development methodology are classified into:

- Comparing systems to select the best one. Note that comparison is hard, as different systems and the situations in which they work are generally too far apart.
- Diagnosing a system in order to improve it, which means making it either more effective and efficient, or more general.

- Knowing which steps to take in designing a system from scratch.

Design strategies are classified into: design by intuition, by observation (analysing existing corpora), or by simulation (also called Wizard of Oz (WOz) experiment; this means using a human to simulate a computer, see section 10). Iteration in design strategy is also described. Some issues concerning WOz experiments are addressed: a set of subject, wizard, dialogue model, and communication channel variables for WOz are explained. An assessment is described as consisting of two components: characterisation and assessment framework. Characterisation consists of a list of system, user, task, environment, the resulting corpus, and overall characteristics. Assessment framework consists of assessment situation, choice of glass or black-box view, quantitative and qualitative measures, and methodological recommendations. The quantitative metrics listed in the book are: average dialogue duration, average turn duration, contextual appropriateness, correction rate, transaction success.

Group Task Analysis (GTA) (Veer et al., 1996b), (Veer et al., 1996a) prescribes a combination of ethnography with more traditional task analysis methods and notations, like Hierarchical Task Analysis (HTA), work flow analysis, and object modelling. Unlike the name suggests, it covers both analysis and design stages, but does not prescribe precise procedures for specific stages. Furthermore, one may jump in at either stage. The method does prescribe the definition of two task models for each loop in the design cycle, and suggests various methods for obtaining the knowledge needed to form these models. Task model 1 describes the current situation, and is part of analysis. Task model 2 describes the functioning of the envisioned changed system, and is part of design.

Task Oriented Modelling (Warren, 1993) (TOM) splits the problem to be tackled into three models: domain, user, and device model. The development process is split into four steps. The process may be repeated until the resulting system is satisfactory.

1. Preliminary models can be constructed using expertise or experience in order to aid construction of a prototype system.
2. Real user behaviour with the system is observed. From these, the domain, user and device models are constructed.
3. These three models are used to determine the value of the Domain Quality, User Costs, and Device Costs metrics.
4. Redesign decisions can now be made, supported by the models thus formed, so no further experiments are required.

In this article, the three models are not explained further.

Method for Usability Engineering (Lim and Long, 1994) (MUSE) is an attempt to integrate HCI into structured software engineering methods. It is argued that in most methodologies used in practice, the consideration of HCI issues is limited to the evaluation phase, which is usually at the end of the design process. This means there is not enough human factors input. In their own words, human factors are addressed 'too little, too late'. In order to overcome this problem, concepts, notations, and design procedures have to be integrated with existing software engineering methodologies. MUSE's emphasis is on defining procedures of notational transformation, rather than defining HCI knowledge and methods needed to do this successfully; it is assumed that the method is used by experienced HCI developers. The general idea of MUSE is to incorporate both human and computer factors in the task and domain models used in the development process. The basic conception of human and computer factors is by a separation of: online tasks (with computer) and offline tasks (completely without computer) tasks. Online tasks are split into interactive tasks and automated tasks.

A number of notations are suggested, in particular, semantic nets for domain modelling, and structured diagrams for task modelling. The stages of the methodology are:

1. Information elicitation and analysis consists of the extant systems analysis, in which the current version of the system and related systems are analysed. The main results are a task description and an 'extant generalised task model' of each system. An extant generalised task model is an attempt to bring the task description closer to the envisioned system by removing device-dependent details.
2. The generalised task model stage consists of building two task models: the target generalised task model, derived from the client's wishes, and the extant composite task model, which is an attempt to merge the extant generalised task models.
3. The statement of user needs stage results in plain text stating the user needs and a domain of design discourse model.
4. The composite task model stage mainly results in a model of the complete system.
5. The system and user task model stage results in two models, the target system task model (specifying online tasks) and target user task model (specifying offline tasks).
6. The interaction task model stage results in a detailed device-dependent task model, with additional screen layouts at specific points in-between tasks.
7. The interface model and display design stages, which are iterated. They mainly result in interface models (detailed models of dialogue in the form of task models), a dictionary of screen objects, and pictorial screen layouts.

Integration with popular software engineering methodologies is discussed, in particular with the Jackson Structured Development (the result is called MUSE*/JSD). The book ends with an assessment of MUSE and MUSE*/JSD, though it should be noted that limited evidence was available at the time.

8.3 Products of the development process

Specification, or modelling, is description of a system and context in any stage of the development process, typically describing objects, processes, or constraints directly related to the system.

Data collection is gathering raw data about the system which may be interpreted by the developers afterwards. Sometimes, the data is gathered by humans, by hanging around, observing, and keeping a log, as for example happens in ethnographical methods. In other cases, data may be collected automatically, by means of computer input logging or video cameras. In any case, the experimental setup (or absence of any explicit setup, as in ethnography), has to be specified, which describes how the data is obtained. After data is collected, it may be interpreted, resulting for example in performance information (such as performance metrics) or qualitative information (such as a specification). Data collection only happens in the analysis and evaluation stage.

We define abstraction as simplification or reduction of information to filter out only those parts that are the most relevant, as seen from a specific viewpoint. Specification necessarily means abstraction, though some abstractions may be more explicitly or better chosen than others. Note that data collection also implies abstraction, because, necessarily, not all data is collected. It is important to realise that abstraction always implies throwing away data that *may* actually turn out to be relevant. Abstractions are sometimes chosen according to the same theory that underlies the design of the system that is being abstracted. If the

abstraction is wrong, the developers may remain blind to the real issues. So, it is useful to be aware when abstraction is occurring, and to know the underlying assumptions.

8.3.1 Formal specifications

Formal specification is unambiguous and explicit, hence it allows exact reasoning and description. Sometimes such a specification is executable (which means it can be run on a computer) as well. It is also possible to under-specify, explicitly leaving things unspecified, thus specifying a range of systems, rather than just one. Underspecified specifications are generally not simulable.

The article (Wright et al., 1997) shows an example of how formal specification may play a role in the HCI development process. Formal specification (Z) is considered complementary to non-formal empirical evaluation (cognitive walkthrough and usability inspection). Formal modelling enforces precision, and going from an abstract to a concrete model shows possible options clearly. Empirical evaluation shows what relevant aspects of the model are still missing, and need further elaboration.

The article (Palanque and Bastide, 1996) argues for formal modelling of both the user's task (task model) and the system's functionality using Petri nets. It is shown how these two models can be combined to obtain a human-computer-interaction model with explicit and even executable dynamics. This allows a clear view of what roles the human and computer are playing, and may make remodelling by shifting tasks between human and computer easier.

9 Design and implementation

9.1 Design methods and notations

(Lauesen and Harning, 1993) argues that current UI design is mostly done either using formal methods (which are not good for user-centered design) or using prototyping (which is not structured enough for large systems). An attempt is made to supply an alternative by using a variation on traditional design methods for user-centered design, participating the users in the design process. Special care has to be taken to make sure the users understand the specifications, i.e. the diagrams used in the design method have to be novice-friendly. The notations in the method consist of:

1. user data model (the user's relevant data and its relationships) preferably shown in the way it may be shown in the real system
2. task list, together with a 'formal' check whether all data can be seen and changed to make sure the task list is exhaustive,
3. information demand diagrams (which information must be visible during each task),
4. function diagram (like information demand diagrams but with the possible operations that can be done),
5. screen outlines,
6. state diagrams,
7. syntactical design (binding the operations to modalities).

The article (Lim and Long, 1993) argues for using structured notations from software engineering for UI specification. In their view, structured notations may solve the lack of specificity,

descriptive scope, communicability, and maintainability of current methods. The notations are intended to be read by the users as well. Examples given are: semantic nets, network diagrams (DFDs), and structured diagrams (flowcharts).

In (Ehrich and Williges, 1986) chapter 3, a notation for control and data flow for specifying dialogues, called SUPERMAN, is described. Chapter 5 describes language specification in Backus-Naur Form (BNF) and by stating examples.

Some notations are simply programming languages, tailored for interaction design. Sometimes, the programs written in these languages are not quite fit for producing professional products, because the languages are very limited, to keep them simple, but they are still useful to create an approximation of an envisioned system which allows early analysis. Examples are Speechmania (Philips, 1998), which is meant for NL dialogue design, and 3dt (Lewin, 1997), which is meant for dialogue design for more traditional user interfaces.

The article (Baekgaard, 1995) describes Dialogue Description Language (DDL). DDL consists of three layers: the graphical layer, the frame layer, and the textual layer. In this article, only the graphical layer is described, which is similar to a finite-state automaton. It is not possible to describe mixed-initiative systems with this language.

The article (Palanque and Bastide, 1996) describes Petri nets to model both human and computer activity in a global task. It is also shown how the Petri net can be used to provide automatically-generated help on specific actions.

9.2 Design environments

The article (Kinoe et al., 1993) introduces a tool for both the analysis stage and the redesign stage of (iterative) user-centered UI design. In the first stage, empirical data is segmented (cut up in units according to the user's basic subtasks) which are tagged (given attributes according to the analysis model). This is done by hand with support by the tool. The results can be ordered in several ways, and can then be reordered and commented on by hand. Reordering is supported by a genetic algorithm that reshuffles orderings. The designers can select the produced orderings that look best.

DIGIS (Bruin and Bouwman, 1993) (Direct Interactive Generation of Interacting Systems) is an object oriented design environment based on the PAC (Presentation, Abstraction, Control) model. UI design is argued to have 3 aspects: presentation, control flow, and interfacing with application. Objects consist of attributes and access protocols. A protocol is described using a regular expression that defines the allowable sequences of actions. On top of that, higher-level tasks can be described as regular expressions of lower-level tasks.

AME (Martin and Winterhalder, 1993) (Application Modeling Environment) is based on using traditional CASE-tools for object oriented (OO) analysis and design, combined with a knowledge-based User Interface Management System. AME's representations are split into three levels: analysis/design, construction, and generation. In the analysis/design level, an OO representation of the interface is constructed using object oriented analysis and design tools. This can then be refined in the construction level. The generation level only generates code. Note that there is no extra help for UI design except the ability of constructing an interface rapidly, assuming that using OO as a paradigm is indeed a good choice.

10 Analysis and evaluation

The book (Jones and Galliers, 1996) addresses some methodological aspects of NL evaluation in great detail. The book is mostly about speech recognition, language parsing, language

generation, and NL information retrieval (this aspect is the closest to NL dialogue).

First, terminology and experimental design issues are addressed. An evaluation has a setup (environment of the system), system (including goal and identifications of subsystems), task, and domain language. Evaluation is separated into three levels: the criteria, measures, and methods levels. The *criteria* are the general requirements. They are classified into intrinsic (related to system objective), and extrinsic (related to system function inside setup). Multiple *measures* (or metrics) may be used to measure each criterium. Measures may be general (applicable to multiple systems), in the form of baselines or benchmarks, and can be compared to exemplars or norms. A *method* is the way an experiment is designed. Evaluations are classified into investigation (reviewing a system at work), and experiment (trying to figure out how something will work). The factors that determine performance are classified into system variables and environment variables. It is argued, though, that it is hard to identify environment variables meaningfully. Note that a system may be viewed from different angles and perspectives: the goals or interests of different users/people may be very different, implying there is no one set of criteria that's unequivocally important. For evaluation, generic systems (systems which can be reprogrammed entirely) are a particularly hard case: possible setups and environments may vary greatly, and cost of customisation is also important.

Evaluation tools are described next. For the purpose of comparative evaluation, these tools are shared criteria, measures, and methodologies. General problems with the use of such tools are incomparability of systems, or goals and setups that are too different. For basic evaluations, there are: test data, evaluation data (this is test data with answers), benchmarks, test beds, (support) toolkits. Tools from social science are: the general usability criteria effectiveness, efficiency and acceptability, and the general validation measures reliability (are the results consistent?), and validity (how good is the relation to criterium?). There is supposed to be a norm value for each measure. Other things discussed are antecedent and intervening variables, and quantitative and qualitative measures.

The book continues with a review of evaluations and evaluation methodologies in the literature. The only actual dialogue systems reviewed are database query systems. Typical measures used are ratio of successful answers, percentages of utterances understood, duration measures, problem complexity measures, and utterance rating (such as type of utterance and correctness). An interesting measure is dialogue tree breadth, obtained by asking many people what utterance should be next at a point in the dialogue. A particular problem is the validity of using general measures, which may not be quite applicable to systems. For example, a parser may yield output that works very well for the next subsystem in the chain, but may look bad when viewed in comparison with a parser norm based on parsers with a slightly different purpose or underlying theory. Summarising performance in only very few numbers is dangerous: the numbers do not answer why a system has a particular performance, and the numbers may not even be relevant, especially in view of variations in the larger setting of a system.

Corpus issues are discussed next. Addressed first is design and purpose of Wizard of Oz (WOz) experiments. The idea behind WOz is to assist evaluation by supplying corpora which can be tested. The performance results can be gathered easily and may cover many aspects of language processing. However, it is limited to syntax parsing, and the usual problems of incompatible metrics are not solved. Also, domain properties are not accounted for. Further corpus design issues addressed are using corpora and test collections, test suites and tools, architectures (which means ways to build systems from components), and standards (such as annotation standards).

Mega-evaluation, which is evaluation across many systems, is addressed next. Two models are discussed: the hub-and-spokes model and the braided-chain model. In the hub-and-spokes

model, evaluations are considered to be linked (meaning that they are comparable) when they have common data, tasks, systems, or metrics. The links can be mapped to a hub-and-spokes map, with the hub standing for inter-system comparison, and the spokes for testing data mismatch problems. In the braided chain model, comparison among different tasks is modelled as well. Different kinds of systems can be 'braided' together by supplying the output of one as input to another. This way, different kinds of systems can be evaluated stand-alone as well as in combination with other systems.

Some final comments and issues on evaluation are given. In particular, it is claimed that evaluation is generally task-oriented, and it is generally not clear how to decompose or identify system and environment factors.

10.1 Some analysis methods and notations

The article (Frascina and Steele, 1993) discusses using task analysis for UI development for a hospital information system. The old method (using paper instead of computers) was analysed first. The task analysis methods used is called TAKD (Task Analysis for Knowledge Descriptions). It consists of: data collection and creation of a list of activities, then construction of a task description hierarchy from the activity list, and then an analysis of sentences of Knowledge Representation Grammar, which are generated from the task hierarchy.

The article (Chase et al., 1993) aims at describing and assessing different user activity notations from the literature according to: scope (what design activities/phases they support), content (what design aspects/objects they can represent), and requirements (what kinds of communication/documentation are required). A 3D chart is made with a (hopefully) covering set of criteria describing each of the three dimensions. Each intersection point in the 3D cube can be given a value. The method was tested by measuring analysts' agreement after analysing User Action Notation (UAN) using the method.

A method for finding out and anticipating common errors by means of early experiments in multimodal systems is described in (Trafton et al., 1997).

The article (Brouwer-Janse, 1995) describes a method to analyse problem-solving procedures. First, data is collected by means of the think-aloud method, followed by an interview and a retrospective report of the task. Analysis is done in two stages: first, the procedures used are identified by analysing the data, then, each step in each procedure is classified into one of 24 'subroutines'.

In (John and Marks, 1997), the effectiveness of some usability evaluation methods is compared. The methods compared are: claims analysis, cognitive walkthrough, GOMS, heuristic evaluation, user action notation, and simply reading the specification. Comparison is done by determining whether problems were identified, whether the problems led to a design change, and whether the design change was good or bad. The methods were tested on a partially-implemented multimedia authoring package. The results are not clear-cut, but it was concluded that they were generally unsatisfactory, especially since the 'simply reading the specification' method seems to come out relatively well.

10.2 Data collection

Figure 2 shows the possible experimental setups and sources of experimental data. This only concerns the evaluation of a one-on-one system, and the broader context of global goals or organisational settings is not included. The different combinations of agents and data flows shown in the figure will be explained below.

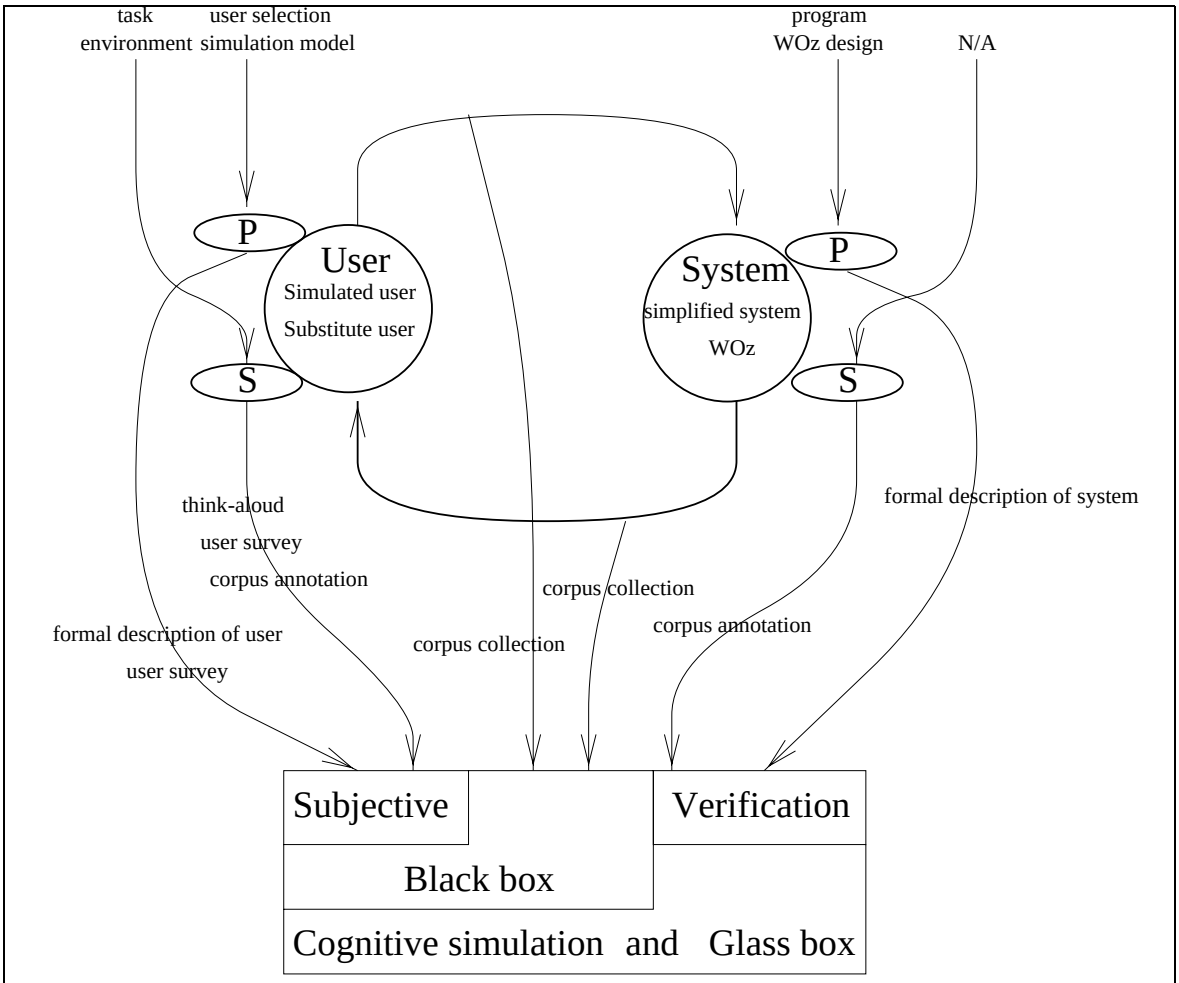

Figure 2: Evaluation setups and data collection
This figure shows the different evaluation setups encountered. Circle = role. The bold letters in each circle denote the agent that normally fills that role. Box = evaluation method. P = Static properties of agent. S = State of agent. Arrow = data flow. The text denotes the different options available for each data flow, agent, or evaluation method. N/A = not relevant.

The following combinations of systems may be found:

- **Wizard of Oz (WOz) simulation**. One places a human where the computer normally is. This is often done to obtain initial data for a system, when no working computer system is available yet. WOz is only applicable to situations where the interaction can be achieved relatively easily by a human, and relatively hard by a computer. It is usually used for NL systems, and sometimes used for multimodal systems. A special case is the 'bionic wizard', which is a human wizard, heavily aided by computer tools to simulate the interface.
- **Computer simulation**. One places a computer where the human normally is. This is attractive in the sense that human effort is greatly reduced. Usually, the system's part of the interaction is simplified.
- **Verification or cognitive walkthrough**. One uses a (possibly much simplified) specification of the user and system, instead of an actual experimental setup of user and system. Analysis of certain properties may be done formally (verification) or informally (cognitive walkthrough), completely by hand or by using computer tools.
- **Prototype or mock-up**. One uses a simplified system where the system should normally be.

An unusual example of protocol verification in NL systems is found in (Guinn, 1995). Here, the self-consistency of a natural-language dialogue system is tested by simulating it against itself. Since NL dialogue tries to offer 'human-like' communication, the relation between human and computer is more symmetrical, and it becomes feasible to make a computer system talk to itself to test it.

In an evaluation with a real system and real users, the role of the user may be filled by actual end users, or by other people representative of the end users. The task and environment may be the real-life situation (which is called field study) or may be designed carefully to reduce random environment factors (which is called laboratory- or controlled study).

The following means of getting data for analysis are found:

- **corpus collection**. This includes all data that can be collected fully automatically. It is relatively easy to obtain a corpus for a given system: for example, offer the system as a free service to the people it was designed for. However, the intentions behind the actions done by the users is not known.
- **think-aloud method**. This is an attempt to gather immediate and detailed information about what the user is thinking when he/she is using the system, rather than information that has to be reconstructed afterwards. Note that the requirement to talk while using a system may influence the user's behaviour. Speech systems are the worst case, as the user is already using speech and language in his/her interaction with the system. See (Ericsson and Simon, 1980) for detailed information about issues concerning the gathering and use of think-aloud data.
- **user surveys**. Instead of using think-aloud, the user may also be asked about the system afterwards. This may be both qualitative information (why did you do that?) or quantitative information (did you find this particular aspect of the system good or bad?).
- **corpus annotation**. Usually, corpora are annotated by hand, which is very time-consuming. Note that corpus annotation is generally not done by the users themselves,

so it is an attempt to interpret what the user was thinking when he/she was making contributions to the corpus. In case annotation consists of parsing the users' NL utterances, for which it is most often used, interpretation is relatively straightforward.

There are several attempts at standardisation of corpus annotation. Such standardisation should allow general tools to be used for annotation, and better comparison of different systems. One such attempt is described in (Dahlback et al., 1997). It addresses:

- Forward-looking communicative function. For each utterance, there can be zero or more labels out of the types: statement of belief, addressee's future action, commitment of action, and miscellaneous (including explicit performative, exclamation, convention openings/closings). It was chosen that the label should be assigned according to the utterance's effect rather than the speaker's intention. The labeler may not take into account too many future utterances when interpreting an utterance. Collaborative completions are not resolved. It was noted that, in the current scheme, joint future action cannot be distinguished well from multiple individual future action.
- Backward looking communicative function. Issues here are: what is being responded to? Which utterances belong to the response? What is the type of the response? The attributes to be tagged are: understanding, agreement (which is classified into closing, nonclosing, and hold), informational relation with source utterance, answer, segmentation of dialogue acts, information level (task, about task, communication, about communication, nonrelevant) and status (old, new).
- Coreferences. Words that refer to other words are labelled with where they refer to.

In the proceedings (Andernach et al., 1995), using corpora for systems development is addressed. Using corpora for utterance type prediction is addressed in (Andernach, 1996). Using corpora for obtaining a simulated-user model for evaluation is described in (Eckert et al., 1998).

The following means of analysing data are found:

- **Qualitative analysis.** Here, the result of analysis is some kind of specification of what is going on in the interaction, hopefully resulting in new insight, for example in the form of an explicit task, user, or system model.
- **Metrical analysis.** Here, data from either corpora or quantitative user surveys is summarised by measuring and counting surface properties. For corpora, this can for example be duration and success percentage measures.
- **Formal analysis.** Here, formal properties of a system are derived, for example, reachability of certain states and deadlock-freeness.

Some examples of formal verification exist. In (Lewin, 1997), deadlock and reachability of states is verified, using a finite-state model of the system. In (McInnes et al., 1995), similar verification is done, though it also includes some text style checks. The article also discusses the possibilities of statistical state-transition checking, based on statistics of frequency of use. This idea is much like the user simulation scheme described in (Eckert et al., 1998).

10.2.1 Simulating users

There are a particularly large number of user simulation systems, which are described in this separate section. Usually, the simulation model is directly based on cognitive psychology, and often, the results of the simulation are used in combination with metrical methods.

The article (Haan et al., 1991) summarises a number of simulated-user analysis models: ETIT, TAG, GOMS, ETAG, CCT, and CLG. All these models are based on a compositional description of the user's task. Rules are defined which break down a high-level description of the task into low-level descriptions (usually, this simply means the steps to be taken to achieve the task). Levels that are typically identified in these models are some selection from the levels task, semantic, syntactic, and physical. Usually missing in these models is a way of accounting for error. Missing in the examples and considerations in the article however, is the possibility of modelling the effect of computer feedback (which is essential in NL systems; note that this may also account for error). Computer feedback may actually cause the user's subtasks and subgoals to change, especially for tasks where the outcome is not known yet (for example in exploration, when getting to know a computer system, or when searching complex databases).

Goals, Operators, Methods, and Selection rules (GOMS, see (Shneiderman, 1998) page 55 and (Olson and Olson, 1990) for an overview) is a procedural model of user activities. GOMS is a well-established method, and many variations exist. GOMS tries to model users' tasks all the way from the 'goal level' to the 'operator level' (which is the lowest level of subtasks). This is done using a set of rules (methods) describing the sequence of steps (operators) that have to be taken to complete a goal, and rules (selection rules) of how to choose between alternative methods according to more specific goal information.

The main idea is that one arrives at a sequence of operators, which are low-level enough to allow prediction of performance (like speed) easily, using easily-obtainable experimental data (for example, duration of pressing a key, typing a word, clicking a mouse, etc.). At this lowest level, a simple cognitive performance model is assumed. Basically, it amounts to summing up all operator durations to arrive at the total duration. Some variations on GOMS have more complex low-level cognitive models, which take into account operators which can be done simultaneously, by using critical path analysis (Gray et al., 1990). Users can be simulated by feeding all rules into an inference engine or an AI (for example, SOAR), and then feeding the resulting sequence operators into the low-level cognitive model.

GOMS is typically used for traditional HCI in which the users' goals are clearly defined and the user drives the system. However, it was also used with good effect for a real-time machine-paced interface with machine-driven subtasks (a video game) (John and Vera, 1992), which shows that GOMS is more universally usable.

(Byrne et al., 1994) describes a system (USAGE) that automatically generates a NGOMSL specification from a user interface specification created in an UIDE (User Interface Development Environment) and runs it with user task specifications to obtain an efficiency prediction automatically. The biggest limitation mentioned is that no multi-level task hierarchies are supported.

Execute Process-Interactive Control (EPIC) is described in (Kieras et al., 1997). EPIC is based on CPM-GOMS, which makes use of information about temporal dependencies of subtasks, and uses the Model Human Processor model to simulate human behaviour. Like CPM-GOMS, EPIC is advocated as being well-suited for multimodal and complex tasks. It is argued that CPM-GOMS is labour-intensive, as the sequence human behaviour corresponding to each task to be tested must be specified explicitly. EPIC tries to solve this problem by generating simulated human behaviour directly from the task specification. The article refers to other cognitive simulation models MHP, HOS, SAINT, CCT, ACT-R, and SOAR.

EPIC consists of a relatively complete and detailed cognitive model, with auditory and visual processor, vocal and manual processor, and a cognitive processor with short-term memory which is based on production rules. This means that the computer party, with input and output devices, can also be effectively simulated with a good deal of detail. This way, reactive

tasks (where information that influences the task structure only arrives later on in the task) can also be modelled. Results show that performance times can be predicted reasonably accurately.

The Procedural Knowledge Structure Model (PKSM) is described in (Benysh and Koubek, 1993). PKSM models procedural knowledge as a flowchart of which some nodes (called the task goals) can be subdivided further until one obtains the full-detail flowchart with only task actions and decisions. The flowchart allows multiple ways to do the same task.

There are also automatic methods which are specifically used on NL dialogue. In (Baber and Hone, 1993) and (Hone and Baber, 1995), task flow modelling is used to determine dialogue duration. The emphasis lies on comparing different repair strategies in relatively simple NL dialogues. In their task flow model, statistical word duration, word misrecognition probability and utterance correction probability are used as parameters for a flowchart model of the dialogue, which can then be used to predict task duration. One of their findings was that the best choice of strategy depends on error rate.

In (Eckert et al., 1998), a dialogue system is evaluated using a statistical user model, obtained from statistical corpus analysis. The model assumes there are only a limited number of different kinds of utterances, and that the user's utterance depends on the last few utterances only.

10.3 Metrical evaluation

Evaluation by using metrics is used often, and warrants a separate section. Such evaluation tries to tell whether a system component or aspect is good or bad. Metrics are easy to use, but there is the risk that important information is not reflected in them. Metrics are often used to compare one system to another system or another version of the system (comparative analysis), or to check whether an existing system is acceptable. Such comparative analysis is useful for verifying a redesign or for deciding whether to commit to the usage of a new system. It may even be possible to compare multiple systems across multiple domains (benchmarking). See (Minker, 1998) and (Hirschman and Thompson, 1996) for an overview of metrical evaluation. The evaluation schemes described below are mostly for evaluating NL systems.

Note that there is often no way to verify the evaluation method itself. For some concrete problems that may occur, see (Walker, 1989). Also, metrical evaluation does not explicitly say *why* one system is better or worse than another, though there are some ways in which it may help to obtain such information:

- Subject each component of a system to performance analysis, so bottlenecks can be identified. The interaction between components and the effect on overall performance must be accounted for.
- Use performance analysis as a filter to obtain the most interesting examples from a large corpus.
- Do a performance analysis after a redesign step to determine the effect of the redesign.

Some metrical systems have been proposed, with metrics that try to measure various things. The most often used metric is length, which comes in the form of duration and number of turns. Another is accuracy, for instance word recognition accuracy, meaning accuracy, and number of turns spent on repair dialogues.

In (Minker, 1998), some benchmarking metrics are proposed. These are word recognition error (substitutions, insertions, deletions), semantic all-label (compare all generated semantic

word labels with manual transcriptions), semantic concept-value (compare only the labels that belong to values that fill the slots in the relevant query), system response (first, see how answerable the user query was, then compare the system's information with a minimal and maximal reference answer). The article also addresses evaluation of translation systems.

Another metrical evaluation framework is PARADISE (see (Walker et al., 1997b) and (Walker et al., 1997a)), which measures success rate and dialogue duration. PARADISE's aim is to specify metrics that calibrate for differences in tasks, so it can be used for comparison across different systems. A metrical system similar to PARADISE can be found in (Eckert et al., 1998).

In (Danieli and Gerbino, 1995), more metrics are described. These are: Implicit Recovery (IR), Contextual Appropriateness (CA), Turn Correction Ratio (TCR), and Transaction Success (TS).

TRAINS95 (Sikorski and Allen, 1996) and TRAINS96 (Stent and Allen, 1997) have a task (namely, find as short a route as possible) which allows multiple possible solutions but which is clear enough to allow a special metric to be used effectively, namely solution quality.

In (Albesano et al., 1997), evaluation of Dialogos is described. A special metric is used for transaction success. This amounts to classifying transactions into: Success (S), Success with Constraint Relaxation (SC), System Failure (SF), User Failure (UF).

In (Polifroni et al., 1998), metrical analysis of whole systems and system components is described, with the main criterium measured being accuracy. Accuracy was obtained by comparing the computer parse with a human transcription containing syntactic and semantic information. A concrete system with several components is analysed. The discourse-tracking module is tested by looking at the accuracy difference between the semantic frames with the predictor turned on or off. The response generator is tested by comparing the query hypothesis with the query hypothesis resulting from the transcription.

(Dillon et al., 1993) studies effects of vocabulary size and experience on efficiency and acceptability of speech systems. Efficiency was measured using completion time, word non-recognitions, phrase misrecognitions, and items skipped by the users. Acceptability was measured by a survey with 15 bipolar questions. Basically, both improve efficiency, and experience improves acceptability.

Part III
Conclusions

11 Review of this text and future directions of the framework

After getting an idea of what literature is relevant for understanding human-computer systems, we may attempt to specify the framework in further detail.

11.1 What are the problems anyway?

When looking at today's everyday computer applications and transaction systems, it is apparent that there are many usability problems. In the literature discussed here, the existence, and the cause and nature of these problems are often implicit in the solutions offered. Sometimes it is questionable whether the solutions are actually solutions to real, existing problems. Hence, the next stage of this research necessarily includes working with a real system.

In real life, it is often the case that the practice of developing systems is purely by intuition, rather than by making use of any of the available theories or methods. Even if such theories or methods are used, results are not always clear-cut. It appears that they are not easy to use, because (Nielsen, 1995):

1. considerable experience and insight is needed to apply them successfully (Blandford, 1998),
2. communication between parties (in particular the software engineering, HCI, and end-user parties) does not proceed easily (for example, in the application of MUSE (Lim and Long, 1994) it was mentioned that the communication prescribed was starting to take too long and was shortened),
3. they imply considerable investment of resources.

In a way, one of the usability problems is that the usability frameworks themselves need to be made more usable also.

NL and multimodal systems have some special problems. In these systems, an attempt is made to make the interaction strategies, and not just some of the styles, analogous to real-life, and in particular, to human-human interaction strategies. However, this implies that the computer has to do things it isn't very good at. To make the systems work, natural interaction has to be 'forged' by only accounting for the interaction patterns that occur most frequently, and by careful engineering around the available techniques, like the various existing speech processing, integration of modalities, NL parsing, and dialogue tracking techniques. Most of these techniques have their own specific problems, and, when built into a system, often turn out to have unpredictable weaknesses. Part of the weaknesses are problems that are caused by inconsistency between the conceptual model and the actual system. A basic example is that people tend to overestimate the abilities of NL systems. Hence, a relatively large part of NL and multimodal systems development is about corpus collection and evaluation.

A first look at our own Schisma system makes it apparent that its domain coverage could be improved, even, for example, by the ability to supply answers to questions such as 'Where am I?' or 'Who are you?', since users could have arrived at Schisma from any location on the Web. It could also be improved at the dialogue level, for example by providing a better invalidation mechanism for repair utterances. Perhaps, the story is not as easy as it sounds,

and these particular problems are a result of underlying design decisions that imply that the other feasible alternatives are worse. Perhaps more problems, or solutions, could be found after specifying the system in a more abstract way, or making a first analysis of the system's possible usage and real-life setting.

The current development surrounding Schisma includes adding Schisma as an agent (the 'Karin' agent) into the Virtual Music Centre (VMC) virtual-reality environment, giving Karin a facial expression, adding other agents to make the system easier and/or quicker to use, such as the talking notice board and the navigation-assisting agent, and adding new possibilities to the lower-level interface, like multiple dynamically-generated and tiled windows. This implies a lot of extra complexity, and may provide a test bed for examining many of the aspects of human-computer system development we have encountered in this review of literature.

11.2 What solutions could a framework offer?

In the literature we have reviewed, we have found several major classes of strategies for attacking the usability problem:

1. Providing classifications of interaction strategies and styles, and supplying recommendations or guidelines about them.
2. Clarifying design decisions by means of various kinds of diagrams, formal specification, or by making a prototype or a simple screen layout for examination by the relevant people.
3. Obtaining information about the (old or new) system's real functioning, setting, and users, by using various analysis strategies (such as ethnography and interviews) and notations (such as hierarchical task diagrams).
4. Verifying the design by means of quantitative user surveys, metrical and simulated-user evaluation, or formal verification.
5. Trying to give a proper place for each of these strategies in any practical development process by defining methodologies.

11.3 The framework, take two

In this second attempt at the framework, we try to make room for all of these strategies. Perhaps we will fill in only one or two of them, but further research may result in a more complete development framework. The framework may be part of a methodology, even if just prescribing documentation of the system using certain specifications at first. Some care should be taken to make the framework developer-friendly.

It seems feasible to use the current VMC system as a test-bed to examine if such a framework can be a help in its development. The current research is centred around a formal specification notation, based on 'agent' models. It seems feasible to use such a compositional 'agent' model for this system. Such a model could be used to specify interaction between the different elements of the interface, both at the 'deeper' level (the agents walking around inside the 3D environment) and the connection between the 'deeper' and 'surface' level (how are the windows that are opened and can be used related to the agent–for example, which window belongs to which agent?). Windows may be modelled as a special kind of agent.

Such a notation, if properly designed, may aid in all five strategies listed:

1. provide recommendations based on the framework of possibilities within such a specification, for example, how to keep the model 'consistent' when viewed as a conceptual model,
2. (a) clarify design decisions by allowing multiple views, for example, conceptual model and implementation model, or internal model (agents only) and full model (agents, windows), or a dynamical communication model of one agent, and the system of agents as a whole,
   (b) demonstrate the working of the actual system more easily by being executable,
3. be suited as part of the documentation for systems analysis,
4. (a) be suited to formal verification because of its formal properties,
   (b) be suited for simulated users by modelling users as agents within or as part of the system,
5. provide a basis for documentation in a methodology.

It may also provide help in the issues specific to NL and multimodal interaction. The context and all available communication channels for each agent should be made explicit. Existing agent communication frameworks could be adopted, giving a basis for reasoning, communication, and dialogue. Existing agent or parallel programming languages could be adopted, allowing parts of the specification to be immediately executable.

Our current research includes combining process algebra (CSP) and predicate logic (Z) to obtain a system model at the 'deeper' level. The process of going from an 'intuitive' model to a formal model is being examined. For example, our CSP specification starts with a data flow diagram. After CSP specification, we will try to add more details by means of Z. Our first experiences with modelling the VMC's agent platform were that several things about the documentation need further clarification. HCI aspects and implementation aspects may be unified by viewing the model as a conceptual model so it can be seen whether it is consistent. For example, we found that discussions emerged about the naturalness of having the blackboard communicate with Karin directly, about whether agents that were in different parts of the world should know of each other, or how a dialogue initiation should take place, based on context cues such as proximity and direction of vision. Possibly, formal verification of consistency may be possible by designing a generic abstract model, and fitting a concrete model of the current system into it.

References

Albesano, D., Baggia, P., Danieli, M., Gemello, R., Gerbino, E., and Rullent, C. (1997). Dialogos: A robust system for human-machine spoken dialogue on the telephone. In *Proc. ICASSP '97*, pages 1147–1150, Munich, Germany.

Allwood, J. (1995). Dialog as collective thinking. In Pylkkänen, P. and Pylkkö, P., editors, *New Directions in Cognitive Science. Publications of the Finnish Artificial Intelligence Society. International conferences no. 2.*

Andernach, J. A. (1996). A machine learning approach to the classification and prediction of dialogue utterances. In *Proceedings of the Second International Conference on New Methods in Language Processing*, pages 98–109.

Andernach, J. A., Burgt, S. P. v., and Hoeven, G. F. v., editors (1995). *TWLT9: Corpus-based approaches to dialogue modelling.*

Andersen, P. B. (1990). *A Theory of Computer Semiotics.* Cambridge Series on Human-Computer Interaction. Cambridge University Press, Cambridge, UK.

Andre, E. and Rist, T. (1995). Generating coherent presentations employing textual and visual material. In *Integration of Natural Language and Vision Processing (Volume II): Intelligent Multimedia*, pages 75–93. Kluwer.

Androutsopoulos, I., Ritchie, G. D., and Thanisch, P. (1995). Natural language interfaces to databases - an introduction. *Natural Language Engineering 1:1*, pages 29–81.

Aust, H. and Oerder, M. (1995). Dialogue control in automatic inquiry systems. In *ESCA workshop on spoken dialog systems: theories and applications*, pages 121–124.

Austin, J. L. (1962). *How to do things with words.* Harvard university press.

Baber, C. and Hone, K. S. (1993). Modelling error recovery and repair in automatic speech recognition. *International Journal of Man-Machine Studies*, 39(3):495–515.

Baecker, R., Grudin, J., Buxton, W., and Greenberg, S., editors (1995). *Readings in Human-Computer Interaction: Towards the Year 2000*, chapter 8. Morgan Kaufmann.

Baekgaard, A. (1995). A platform for spoken dialogue systems. In *ESCA workshop on spoken dialog systems: theories and applications*, pages 105–108.

Barbuceanu, M. and Fox, M. S. (1996). Capturing and modeling coordination knowledge for multi-agent systems. *Internation Journal of Cooperative Information Systems Vol.5 Nos.2-3*, pages 273–314.

Benysh, D. V. and Koubek, R. J. (1993). The implementation of knowledge structures in cognitive simulation environments. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 309–314.

Bernsen, N. O. (1996). towards a tool for predicting speech functionality. *Free Speech Journal*, 1.

Bernsen, N. O., Dybkjaer, H., and Dybkjaer, L. (1998). *Designing interactive speech systems.* Springer.

Beun, R., Baker, M., and Reiner, M. (1995). editor's introduction. In Beun, R., Baker, M., and Reiner, M., editors, *Dialogue and instruction. Modeling interaction in intelligent tutoring systems. Proceedings of the NATO Advanced Research Workshop on natural dialogue and interactive student modeling*, pages 1–12.

Blandford, A. E. (1995). Deciding what to say: an agent-theoretic approach to tutorial dialogue. In Beun, R., Baker, M., and Reiner, M., editors, *Dialogue and instruction. Modeling interaction in intelligent tutoring systems. Proceedings of the NATO Advanced Research Workshop on natural dialogue and interactive student modeling*, pages 157–166.

Blandford, A. E. (1998). Training software engineers in a novel usability evaluation technique. *International journal of human-computer studies*, 49:245–279.

Bouwman, A. G. G. (1998). Spoken dialog system evaluation and user-centered redesign with reliability measurements. Master's thesis, University of Twente, Deparment of Computer Science. February draft.

Bouzeghoub, M. and Metais, E., editors (1995). *Proceedings of the First International Workshop on Applications of Natural Language to Databases (NLDB'95)*, Versailles, France.

Brazier, F., Dunin-Keplicz, B., Jennings, N., and Treur, J. (1995). Formal specification of multi-agent systems: a real-world case. In *First International Conference on Multiagent Systems*.

Brazier, F. M. T., Langen, P. H. G. v., J., T., Wijngaards, N. J. E., and M., W. (1994). Modelling a design task in DESIRE : the VT example. Technical Report IR-377, Faculteit der Wiskunde en Informatica, Vrije Universiteit, Netherlands.

Bretier, P. and Sadek, D. (1997). A rational agent as the Kernel of a cooperative spoken dialogue system: Implementing a logical theory of interaction. In Müller, J. P., Wooldridge, M. J., and Jennings, N. R., editors, *Proceedings of the ECAI'96 Workshop on Agent Theories, Architectures, and Languages: Intelligent Agents III*, volume 1193 of *LNAI*, pages 189–204, Berlin. Springer.

Brouwer-Janse, M. D. (1995). Method for dialogue protocol analysis. In Beun, R., Baker, M., and Reiner, M., editors, *Dialogue and instruction. Modeling interaction in intelligent tutoring systems. Proceedings of the NATO Advanced Research Workshop on natural dialogue and interactive student modeling*, pages 275–285.

Bruin, H. d. and Bouwman, P. (1993). The software architecture of DIGIS. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 244–249.

Bunt, H., Ahn, R., Beun, R.-J., Borghuis, T., and Overveld, K. v. (1995). Cooperative multimodal communication in the denk project. In *Proceedings of the International Conference on Cooperative Multimodal Communication CMC/95*, pages 79–102.

Bunt, H. C. (1995). Dialogue control functions and interaction design. In Beun, R., Baker, M., and Reiner, M., editors, *Dialogue and instruction. Modeling interaction in intelligent tutoring systems. Proceedings of the NATO Advanced Research Workshop on natural dialogue and interactive student modeling*, pages 197–214.

Busemann, S., Declerck, T., Diagne, A. K., Dini, L., Klein, J., and Schmeier, S. (1997). Natural language dialogue service for appointment scheduling agents. In *Fifth Conference on Applied Natural Language Processing*, pages 25–32.

Buxton, W. (1986). Chunking and phrasing and the design of human-computer dialogues. In Kugler, H. J., editor, *Information Processing '86, Proc. of the IFIP 10th World Computer Congress*. North Holland Publishers, Amsterdam.

Byrne, M. D., D.Wood, S., Noisukaviriya, P., Foley, J. D., and Kieras, D. (1994). Automating interface evaluation. In *Proc. of CHI-94*, pages 232–237, Boston, MA.

Callahan, G. (1994). Excessive realism in GUI design: Helpful or harmful? *Software Development*.

Carlson, J. R. and Hall, L. L. (1993). The impact of the design of the software control interface on user performance. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 116–121.

Castelfranchi, C. (1991). No more cooperation, please! in search of the social structure of verbal interaction. In Orthony, A., Slack, J., and Stock, O., editors, *AI and Cognitive Science Perspectives on Communication*. Springer.

Chase, J. D., Hartson, H. R., Hix, D., Schulman, R. S., and Brandenburg, J. L. (1993). A model of behavioral techniques for representing user interface designs. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 861–866.

Chau, R. (1993). Natural language interfaces for integrated network management. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 368–372.

Chu-Carroll, J. and Brown, M. K. (1997). Tracking initiative in collaborative dialogue interactions. In *ACL '97/EACL '97: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and of the 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 262–270.

Cohen, P. P. and Oviatt, S. L. (1995). The role of voice input for human-machine communication. *Proc. Natl. Acad. Sci. USA*, 92:9921–9927.

Dahlback, N., Reithinger, N., and Walker, M. A. (1997). *Standards for Dialogue Coding in Natural Language Processing: Report on the Dagstuhl-Seminar*. Dagstuhl.

Danieli, M. and Gerbino, E. (1995). Metrics for evaluating dialogue strategies in a spoken language system. In *Proceedings of the 1995 AAAI Spring Symposium on Empirical Methods in Discourse Interpretation and Generation*, pages 34–39.

Darses, F., Falzon, P., and Robert, J. M. (1993). Cooperating partners: Investigating natural assistance. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 997–1002.

de Saussure, F. (1916). *Cours de linguistique générale*. Payot. Missing from my collection.

Dessalles, J.-L. (1998). The interplay of desire and necessity in dialogue. In *TWLT13: Formal semantics and pragmatics of dialogue*, pages 89–97.

Diel, H., Uhl, J., and Welsch, M. (1993). An information-based user interface architecture. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 110–115.

Dignum, F. and Linder, B. v. (1997). Modelling social agents: Communication as action. In Müller, J. P., Wooldridge, M. J., and Jennings, N. R., editors, *Proceedings of the ECAI'96 Workshop on Agent Theories, Architectures, and Languages: Intelligent Agents III*, volume 1193 of *LNAI*, pages 205–218, Berlin. Springer.

Dillon, T. W., Norcio, A. F., and DeHaemer, M. J. (1993). Spoken language interaction: Effects of vocabulary size and experience on user efficiency and acceptability. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 140–145.

Dybkjaer, L., Bernsen, N. O., and Dybkjaer, H. (1995). Scenario design for spoken language dialogue systems development. In *ESCA workshop on spoken dialog systems: theories and applications*, pages 93–96.

Eckert, W., Levin, E., and Pieraccini, R. (1998). Automatic evaluation of spoken dialogue systems. In *TWLT13: Formal semantics and pragmatics of dialogue*, pages 99–110.

Eco, U. (1976). *A theory of semiotics*. Indiana University Press.

Ehrich, R. W. and Williges, R. C., editors (1986). *Human-Computer Dialogue Design*, volume 2 of *Advances in Human Factors/Ergonomics*. Elsevier Science Publishers B. V., Amsterdam.

Ek, Å. (1997). Margrathea - a planning tool for environment adaptation. usability test and evaluation. Master's thesis, Lund university, Sweden.

Encarnacão, M. (1997). *Concept and realization of intelligent user support in interactive graphics applications*. PhD thesis, Faculty of computer science, Eberhard-Karls-University, Tübingen.

Ericsson, K. A. and Simon, H. A. (1980). Verbal reports as data. *Psychological review*, 87(3).

Fahnrich, K.-P. and Hanne, K.-H. (1993). Aspects of multimodal and multimedia human-computer interaction. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 440–445.

Fais, L. and ho Loken-Kim, K. (1995). Lexical accomodation in human-interpreted and machine-interpreted dual language interactions. In *ESCA workshop on spoken dialog systems: theories and applications*, pages 69–72.

Faraday, P. and Sutcliffe, A. (1993). Toward a walkthrough method for multimedia design. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 452–457.

Fass, D., Hall, G., Laurens, O., Popowich, F., and von Rüden, M. (1996). A distributed intelligent information system with natural language input for ad hoc knowledge discovery in databases. In *Energy Information Management VI: Conference Papers*, volume I: Computers in Engineering.

Fischer, G. and Reeves, B. (1991). Beyond intelligent interfaces: Exploring, analyzing, and creating success models of cooperative problem solving. *Applied Intelligence*, 1:311–332.

Frascina, T. and Steele, R. A. (1993). Task analysis in design of a human-computer interface for a ward based system. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 226–230.

Fraser, N. M. (1995). Quality standards for spoken language dialogue systems: a report on progress in EAGLES. In *ESCA workshop on spoken dialog systems: theories and applications*, pages 157–160.

Genesereth, M. R. and Ketchpel, S. P. (1994). Software agents. *Communications of the ACM*, 37(7):48–53.

Gerlach, M. and Onken, R. (1993). A dialogue manager as interface between aircraft pilots and a pilot assistant system. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 98–103.

Gibbon, D., Moore, R., and Winski, R. (1997). *Handbook of Standards and Resources for Spoken Language Systems*. Mouton de Gruyter, Berlin.

Good, D. A. (1989). The viability of conversational grammars. In Taylor, M. M., Neel, F., and Bouwhuis, D. G., editors, *The Structure of Multimodal Dialogue*, pages 135–144. North-Holland, Amsterdam. Missing from my collection.

Gray, W. D., John, B. E., Stuart, R., Lawrence, D., and Atwood, M. E. (1990). GOMS meets the phone company: Analytic modeling applied to real-world problems. In *Proceedings of IFIP INTERACT'90: Human-Computer Interaction*, Foundations: Cognitive Ergonomics, pages 29–34.

Grice, H. P. (1975). Logic and conversation. In Cole, P. and Morgan, J. L., editors, *Syntax and Semantics: Vol. 3: Speech Acts*, pages 41–58. Academic Press, San Diego, CA.

Guha, R. V. and Lenat, D. B. (1994). Enabling agents to work together. *Communications of the ACM*, 37(7):127–142.

Guinn, C. (1995). The role of computer-computer dialogues in human-computer dialogue system development. In *Empirical Methods in Discourse Interpretation and Generation. Papers from the 1995 AAAI Symposium*, pages 47–52. AAAI Press USA.

Gutwin, C. and McCalla, G. (1992). Would I lie to you? Modelling misrepresentation and context in dialogue. In *30th Annual Meeting of the Association for Computational Linguistics*, pages 152–158.

Haan, G. d., Veer, G. C. v., and Vliet, J. C. v. (1991). Formal modelling techniques in human-computer interaction. *Acta Psychologica*, 78(1-3):27–67.

Heeman, P. A. and Allen, J. F. (1997). Intonational boundaries, speech repairs and discourse markers: modeling spoken dialog. In *ACL '97/EACL '97: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and of the 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 254–261.

Hirschman, L. and Thompson, H. (1996). *Overview of Evaluation in Speech and Natural Language Processing*, pages 475–518. Cambridge University Press.

Hone, K. S. and Baber, C. (1995). Using a simulation method to predict the transaction time effects of applying alternative levels of constraint to user utterances within speech interactive dialogues. In *ESCA workshop on spoken dialog systems: theories and applications*, pages 209–212.

Hulstijn, J. and Nijholt, A., editors (1998). *TWLT13: Formal semantics and pragmatics of dialogue*.

Iwadera, T., Ishizaki, M., and Morimoto, T. (1995). Recognizing an interactional structure and topics of task-oriented dialogues. In *ESCA workshop on spoken dialog systems: theories and applications*, pages 41–44.

Jameson, A., Kipper, B., Ndiaye, A., Schafer, R., Simons, J., Weis, T., and Zimmermann, D. (1994). Cooperating to be noncooperative: the dialog system pracma. In Nebel, B. and Dreschler-Fischer, L., editors, *KI-94: Advances in Artificial Intelligence. 18th German Annual Conference on Artificial Intelligence.*, pages 106–117. Springer-Verlag.

Jameson, A., Schäfer, R., Simons, J., and Weis, T. (1995). Adaptive provision of evaluation-oriented information: Tasks and techniques. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 1886–1893.

Jameson, A. and Weis, T. (1995). How to juggle discourse obligations. In *Proceedings of the Symposium on Conceptual and Semantic Knowledge in Language Generation*, pages 171–185.

John, B. E. and Marks, S. J. (1997). Tracking the effectiveness of usability evaluation methods. *Behaviour and Information Technology*, 16(Issue 4–5):188–202.

John, B. E. and Vera, A. H. (1992). A goms analysis of a graphic, machine-paced, highly interactive task. In *Proc. of CHI-92*, pages 251–258, Monterey, CA.

Johnston, M., Cohen, P. R., McGee, D., Oviatt, S. L., Pittman, J. A., and Smith, I. (1997). Unification-based multimodal integration. In *ACL '97/EACL '97: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and of the 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 281–288.

Jones, K. S. and Galliers, J. R. (1996). *Evaluation natural language processing systems, an analysis and review.* springer.

Jönsson, A. (1993). *Dialogue management for natural language interfaces.* PhD thesis, Linköping University, department of computer and information science.

Kaptelinin, V., Kuutti, K., and Bannon, L. (1995). Activity theory: Basic concepts and applications. *Lecture Notes in Computer Science*, 1015:189–??

Karsenty, L. (1993). Task-dependent descriptions: A preliminary study. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 897–902.

Kieras, D. E., Wood, S. D., and Meyer, D. E. (1997). Predictive engineering models based on the EPIC architecture for a multimodal high-performance human-computer interaction task. *ACM Transactions on Computer-Human Interaction*, 4(3):230–275.

Kinoe, Y., Mori, H., and Hayashi, Y. (1993). Integrating analytical and creative processes for user interface re-design. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 163–168.

Kitano, H. and Ess-Dykema, C. v. (1991). Toward a plan-based understanding model for mixed-initiative dialogues. In *29th Annual Meeting of the Association for Computational Linguistics*, pages 25–32.

Kushilevitz (1997). Communication complexity. In *Advances in Computers, ed. by Marshall C. Yovits*, volume 44. Academic Press.

Lambert, L. and Carberry, S. (1991). A tripartite plan-based model of dialogue. In *29th Annual Meeting of the Association for Computational Linguistics*, pages 47–54.

Lambert, L. and Carberry, S. (1992). Modeling negotiation subdialogues. In *30th Annual Meeting of the Association for Computational Linguistics*, pages 193–200.

Lauesen, S. and Harning, M. B. (1993). Dialogue design through modified dataflow and data modelling. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 220–225.

Laurel, B. (1993). *Computers as theatre*. Addison-Wesley.

Lee, J. R. (1995). Graphics and natural language in design and instruction. In Beun, R., Baker, M., and Reiner, M., editors, *Dialogue and instruction. Modeling interaction in intelligent tutoring systems. Proceedings of the NATO Advanced Research Workshop on natural dialogue and interactive student modeling*, pages 72–84.

Levinson, S. C. (1987). Minimization and conversational inference. In Verschueren, J. and Bertucelli-Papi, M., editors, *The pragmatic perspective: selected papers from the 1985 International Pragmatics Conference*, pages 60–129. Missing from my collection.

Lewin, I. (1997). *3dt: User Manual (draft April 13 1997?)*.

Lewis, C. and Norman, D. A. (1986). Designing for error. In Norman, D. A. and Draper, S. W., editors, *User Centered System Design: New Perspectives on Human-Computer Interaction*, pages 411–432. Erlbaum, Hillsdale, NJ.

Lie, D., Hulstijn, J., Akker, R. o. d., and Nijholt, A. (1998). A transformational approach to natural language understanding in dialogue systems. In *Natural language processing and industrial applications (NLP+IA 98) - Special accent on language learning*, pages 163–168.

Lim, K. Y. and Long, J. (1994). *The MUSE method for usability engineering*. Cambridge University Press.

Lim, K. Y. and Long, J. B. (1993). Structured notations for human factors specification of interactive systems. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 325–331.

Martin, C. and Winterhalder, C. (1993). Integrating CASE and UIMS for automatic software construction. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 291–296.

Martin, J.-C. (1997). Towards 'intelligent' cooperation between modalities: the example of multimodal interaction with a map. In *Proceedings of the IJCAI'97 workshop on Intelligent Multimodal Systems*.

May, J. and Barnard, P. (1995). Cinematography and interface design. In Nordby, K., Helmersen, P. H., Gilmore, D. J., and Arnesen, S. A., editors, *Human Computer Interaction: Interact '95*, pages 26–31. Chapman and Hall, London.

McInnes, F. R., White, L. S., Foster, J. C., and Jack, M. A. (1995). An automated style checker for human-computer dialogue engineering. In *ESCA workshop on spoken dialog systems: theories and applications*, pages 149–152.

Minker, W. (1998). Evaluation methodologies for interactive speech systems. In *First International Conference on Language Resources and Evaluation (LREC)*, pages 198–206.

Moore, J. D. and Paris, C. L. (1989). Planning text for advisory dialogues. In *27th Annual Meeting of the Association for Computational Linguistics*, pages 203–211.

Nagao, K. and Rekimoto, J. (1995). Ubiquitous talker: Spoken language interaction with real world objects. In *IJCAI-95 Proceedings*.

Nagao, K. and Takeuchi, A. (1994). Speech dialogue with facial displays: Multimodal human-computer conversation. In *Proceedings of ACL-94*.

Nardi, B. A. (1992). Studying context: A comparison of activity theory, situated action models, and distributed cognition. In *East-West International Conference on Human-Computer Interaction: Proceedings of the EWHCI'92*, pages 352–359. Missing from my collection.

Nielsen, J. (1993). *Usability engineering*. Academic Press.

Nielsen, J. (1995). Getting usability used. In *Human-computer interaction, Interact '95*, pages 3–12. Chapman and Hall.

Nigay, L. and Coutaz, J. (1997). Software architecture modelling: bridging two worlds using ergonomics and software properties. In Palanque, P. and Paterno, F., editors, *Formal Methods in Human Computer Interaction*, chapter 3, pages 49–73. Springer-Verlag.

Nijholt, A., Hessen, A. v., and Hulstijn, J. (1998). Speech and language interaction in a (virtual) cultural theatre. In *Natural language processing and industrial applications (NLP+IA 98) - Special accent on language learning*, pages 176–182.

Nivre, J. (1995). Communicative action and feedback. In Beun, R., Baker, M., and Reiner, M., editors, *Dialogue and instruction. Modeling interaction in intelligent tutoring systems. Proceedings of the NATO Advanced Research Workshop on natural dialogue and interactive student modeling*, pages 231–240.

Norman, D. A. (1994). How might people interact with agents. *Communications of the ACM*, 37(7):68–71.

Nwana, H. S. and Wooldridge, M. (1997). Software agent technologies. In *Software agents and soft computing*, pages 59–78.

Olson, J. R. and Olson, G. M. (1990). The growth of cognitive modeling in human-computer interaction since GOMS. *Human-Computer Interaction*, 5(2-3):221–265.

Oviatt, S. L. and Cohen, P. R. (1989). The effects of interaction on spoken discourse. In *Proc. of the 27th ACL*, pages 126–134.

Palanque, P. and Bastide, R. (1996). A design life-cycle for the formal design of interactive systems. In Roast, C. R. and Siddiqi, J. I., editors, *Proceedings of the BCS-FACS Workshop on Formal Aspects of the Human Computer Interface*.

Pasquini, A., Monfardini, G., Kaaniche, M., Mazet, C., Anderson, S., Embry, D., Rizzo, A., Veer, G. v. d., Sonneck, G., Ciciani, B., Laprie, J. C., Kanoun, K., Littlewood, B., Mortensen, U., Goerke, W., and Strigini, L. (1998). Lecture sheets and notes from the OLOS reliability and safety of human-computer systems summer school. Handouts.

Passoneau, R. J. (1989). Getting at discourse referents. In *27th Annual Meeting of the Association for Computational Linguistics*, pages 51–59.

Peirce, C. S. (1931–1958). *Collected papers*. Harvard University Press. Missing from my collection.

Philips (1998). *SpeechMania 2.0: Dialogue Description Language, developer's guide, overhead sheets*. Philips.

Polifroni, J., Seneff, S., Glass, J., and Hazen, T. J. (1998). Evaluation methodology for a telephone-based conversational system. In *First International Conference on Language Resources and Evaluation (LREC)*, pages 43–49.

Ramshaw, L. A. (1991). A three-level model for plan exploration. In *29th Annual Meeting of the Association for Computational Linguistics*, pages 39–46.

Rauterberg, M. (1993). Quantitative measures for evaluating human-computer interfaces. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 1 of *V. Methodologies*, pages 612–617.

Reiner, M. (1995). Tools for collaborative learning in optics. In Beun, R., Baker, M., and Reiner, M., editors, *Dialogue and instruction. Modeling interaction in intelligent tutoring systems. Proceedings of the NATO Advanced Research Workshop on natural dialogue and interactive student modeling*, pages 137–155.

Rist, T. and Andre, E. (1993). Designing coherent multimedia presentations. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 434–439.

Roe and Arnold (1988). *checklist of recommendations on human-computer interface design*.

Sadek, M. D., Bretier, P., and Panaget, F. (1997). ARTIMIS: Natural dialogue meets rational agency. In *international joint conference on artificial intelligence*, pages 1030–1035.

Schooten, B. W. v. (1997). Problemen met object-georienteerd programmeren. Student report for Cognitive Ergonomics.

Searle, J. R. (1983). *Intentionality, an essay in the philosophy of mind*. Cambridge university press.

Shannon, C. E. and Weaver, W. (1964). *The Mathematical Theory of Communication*. University of Illinois Press.

Shneiderman, B. (1998). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, Reading, MA, 3 edition.

Sikorski, T. and Allen, J. F. (1996). TRAINS-95 system evaluation. Technical Report TN96-3, University of Rochester, Computer Science Department.

Smith, R. W. (1996). Initiative-dependent features of human-computer dialogs in a task-assistance domain. In *Energy Information Management VI: Conference Papers*, volume I: Computers in Engineering.

Smith, R. W. (1997). An evaluation of strategies for selective utterance verification for spoken natural language dialog. In *Fifth Conference on Applied Natural Language Processing*, pages 41–48.

Smith, R. W. and Hipp, D. R. (1994). *Spoken Natural Language Dialog Systems: A Practical Approach*. Oxford University Press.

Stein, A. and Maier, E. (1995). Structuring collaborative information-seeking dialogues. *Knowledge-Based Systems, Vol. 8 Iss. 2-3*, pages 82–93.

Stent, A. J. and Allen, J. F. (1997). TRAINS-96 system evaluation. Technical Report TN97-1, University of Rochester, Computer Science Department.

Sutcliffe, A. G. and Faraday, P. (1993). Designing multimedia interfaces. *Lecture Notes in Computer Science*, 753:105–114.

Tillmann, H. G. and Tischer, B. (1995). Collection and exploitation of spontaneous speech produced in negotiation dialogues. In *ESCA workshop on spoken dialog systems: theories and applications*, pages 217–220.

Trabelsi, Z., Kotani, Y., and Nisimura, H. (1993). Heuristics for generating informative responses to failing user's queries in natural language database interfaces. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 362–367.

Trafton, J. G., Wauchope, K., and Stroup, J. (1997). Errors and usability of natural language in a multimodal system. In *Proceedings of the IJCAI'97 workshop on Intelligent Multimodal Systems*.

Traum, D. (1997). A reactive-deliberative model of dialogue agency. In Müller, J. P., Wooldridge, M. J., and Jennings, N. R., editors, *Proceedings of the ECAI'96 Workshop on Agent Theories, Architectures, and Languages: Intelligent Agents III*, volume 1193 of *LNAI*, pages 157–172, Berlin. Springer.

Tullis, T. S. (1993). Is user interface design just common sense? In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 9–14.

Uzilevsky, G. (1994). Ergosemiotics of user interface research and design: Foundations, objectives, potential. *Lecture Notes in Computer Science*, 876:1–11.

Veer, G. C. v. d., Hoeve, M., and Lenting, B. F. (1996a). Modeling complex work systems - method meets reality. In *Proceedings of the Eight European Conference on Cognitive Ergonomics*, pages 115–120.

Veer, G. C. v. d. and Lenting, B. F. (1995). *Cognitive Ergonomie en MCI, lecture notes from the 1995 student course*. University of Twente, faculty of WMW, department of ergonomics.

Veer, G. C. v. d., Lenting, B. F., and Bergevoet, B. A. J. (1996b). GTA: Groupware task analysis - modeling complexity. *Acta Psychologica*, 91:297–322.

Walker, M., Hindle, D., Fromer, J., Di Fabbrizio, G., and Mestel, C. (1997a). Evaluating competing agent strategies for a voice email agent. In *EUROSPEECH97: Proceedings of the European Conference on Speech Communication and Technology*.

Walker, M. and Whittaker, S. (1990). Mixed initiative in dialogue: An investigation into discourse segmentation. In *Proceedings of the 28th Annual Meeting of the Association of Computational Linguistics*.

Walker, M. A. (1989). Evaluating discourse processing algorithms. In *Proc. of the 27th ACL*, pages 251–261.

Walker, M. A. (1992). Redundancy in collaborative dialogue. In *Fourteenth International Conference on Computational Linguistics*.

Walker, M. A. (1994). Discourse and deliberation: Testing a collaborative strategy. In *Fifteenth International Conference on Computational Linguistics*.

Walker, M. A. (1996a). The effect of resource limits and task complexity on collaborative planning in dialogue. *Artificial Intelligence*, 85, Numbers 1-2:181–243.

Walker, M. A. (1996b). Limited attention and discourse structure. *Computational Linguistics*, 22-2.

Walker, M. A., Litman, D. J., Kamm, C. A., and Abella, A. (1997b). PARADISE: a framework for evaluating spoken dialogue agents. *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*.

Warren, C. P. (1993). The TOM approach to system development: Methods and tools for task oriented modelling of real-time safety critical systems. In *Proceedings of the Fifth International Conference on Human-Computer Interaction*, volume 2, pages 285–290.

Watt, S. N. K. (1997). Artificial societies and psychological agents. In *Software agents and soft computing*, pages 27–41.

Waugh, D. A. and Taylor, M. M. (1995). The role of feedback in a layered model of communication. In Beun, R., Baker, M., and Reiner, M., editors, *Dialogue and instruction. Modeling interaction in intelligent tutoring systems. Proceedings of the NATO Advanced Research Workshop on natural dialogue and interactive student modeling*, pages 215–230.

Wolf, C., Koved, L., and Kunzinger, E. (1995). Ubiquitous mail: Speech and graphical user interfaces to an integrated voice/e-mail mailbox. In *Interact '95*, pages 247–252.

Wright, P., Merriam, N., and Fields, B. (1997). From formal models to empirical evaluation and back again. In Palanque, P. and Paterno, F., editors, *Formal Methods in Human Computer Interaction*, chapter 14, pages 283–313. Springer-Verlag.

Yamada, K., Mizoguchi, R., Harada, N., Nukuzuma, A., Ishimaru, K., and Furukawa, H. (1993). Model of utterance and its use in cooperative response generation. *Lecture Notes in Computer Science*, 753:260–271.

Zhang, J. and Norman, D. A. (1994). Representations in distributed cognitive tasks. *Cognitive Science*, 18:87–122.

Zoest, A. v. (1978). *Semiotiek: over tekens, hoe ze werken en wat we ermee doen*. Ambo.

Index

Author Index