CUCS-309-87

The Use of Explicit User Models in Text Generation: Tailoring to a User's Level of Expertise

Cécile Laurence Paris

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Science

COLUMBIA UNIVERSITY
1987

© 1987
Cécile Laurence Paris
ALL RIGHTS RESERVED

ABSTRACT

The Use of Explicit User Models in Text Generation: Tailoring to a User's Level of Expertise

Cécile Laurence Paris

A question answering program that provides access to a large amount of data will be most useful if it can tailor its answers to each individual user. In particular, a user's level of knowledge about the domain of discourse is an important factor in this tailoring if the answer provided is to be both informative and understandable to the user. In this research, we address the issue of how the user's domain knowledge, or the level of expertise, might affect an answer. By studying texts we found that the user's level of domain knowledge affected the *kind* of information provided and not just the *amount* of information, as was previously assumed. Depending on the user's assumed domain knowledge, a description of a complex physical objects can be either parts-oriented or process-oriented. Thus the user's level of expertise in a domain can guide a system in choosing the appropriate facts from the knowledge base to include in an answer. We propose two distinct descriptive strategies that can be used to generate texts aimed at naive and expert users. Users are not necessarily truly expert or fully naive however, but can be anywhere along a knowledge spectrum whose extremes are naive and expert. In this work, we show how our generation system, TAILOR, can use information about a user's level of expertise to combine several discourse strategies in a single text, choosing the most appropriate at each point in the generation process, in order to generate texts for users anywhere along the knowledge spectrum. TAILOR's ability to combine discourse strategies based on a user model allows for the generation of a wider variety of texts and the most appropriate one for the user.

Table of Contents

1. Introduction 1
   - 1.1 Language generation and question answering 1
   - 1.2 User modelling in generation 2
   - 1.3 Research method and main contributions 4
   - 1.4 The domain 5
   - 1.5 System overview 9
   - 1.6 Examples from TAILOR 11
   - 1.7 Limitations 11
   - 1.8 A guide to remaining chapters 13
2. Related Research 17
   - 2.1 Related work in user modelling and generation 17
     - 2.1.1 Superposing stereotypes 17
     - 2.1.2 Modelling and using the user's domain knowledge 19
     - 2.1.3 Using knowledge about the user's plans and goals to generate responses 22
     - 2.1.4 Using reasoning about mutual beliefs to plan an utterance 24
     - 2.1.5 Dealing with misconceptions about the domain 25
   - 2.2 Related work in decomposing texts 26
     - 2.2.1 Decomposing a text using linguistic rhetorical predicates 26
     - 2.2.2 Decomposing a text using coherence relations 27
     - 2.2.3 Decomposing a text with rhetorical structure theory 28
   - 2.3 Related work in psychology and reading comprehension 29
   - 2.4 Summary 31
3. TAILOR's user model 32
   - 3.1 Identifying what needs to be in the user model 32
   - 3.2 Determining the level of expertise 35
     - 3.2.1 User type 35
     - 3.2.2 Role of the memory organization 36
     - 3.2.3 Question type and detecting misconceptions 38
     - 3.2.4 Inference rules and the radius of expertise 38
     - 3.2.5 Asking the user questions and using the previous discourse 39
   - 3.3 Conclusions 40
4. The research approach and the theoretical results 41
   - 4.1 Introduction 41
   - 4.2 Discourse strategies and their role in natural language generation 41
   - 4.3 The texts analyzed 42
     - 4.3.1 The textual analysis 44
     - 4.3.2 Analyses of entries from adult encyclopedias and the car manual for experts 47
     - 4.3.3 Texts from junior encyclopedias, high school textbooks, and the car manual for novices 53
     - 4.3.4 Need for directives 60
     - 4.3.5 Summary of the textual analysis 67
     - 4.3.6 Plausibility of this hypothesis 69
   - 4.4 Combining the two strategies to describe objects to users with intermediate levels of domain knowledge 70

4.5 Summary 72
5. The Discourse strategies used in TAILOR 74
5.1 Introduction 74
5.2 Constituency Schema 74
5.3 Process Trace 75
5.4 Requirements of the knowledge base 78
5.5 The process trace: a procedural strategy 79
5.5.1 Identifying the main path and different kinds of links 81
5.5.2 The main path 83
5.5.3 Deciding among several links 87
5.5.3.1 The side chain is long but related (attached) to the main path 87
5.5.3.2 There is an isolated side link 92
5.5.3.3 There are many short links 96
5.5.3.4 There is a long side chain which is not related to the main path 97
5.5.3.5 Substeps 98
5.6 Strategy representation 101
5.7 Open problems 110
5.8 Summary 111
6. Combining the strategies to describe devices for a whole range of users 112
6.1 Introduction 112
6.2 The user model contains explicit parameters 113
6.3 Generating a description based on the user model 115
6.3.1 Choosing a strategy for the overall structure of the description 116
6.3.2 Combining the strategies 118
6.3.2.1 Decision points within the strategies 119
6.3.2.2 Switching strategy when the constituency schema is chosen initially 119
6.3.2.3 Switching strategy when the process trace is chosen initially 122
6.4 Examples of texts combining the two strategies 122
6.5 Combining strategies yields a greater variety of texts 133
6.6 Conclusions 138
7. TAILOR system implementation 139
7.1 Introduction 139
7.2 System overview 139
7.3 System overview 141
7.4 The knowledge base and its representation 143
7.4.1 The generalization hierarchies 149
7.4.2 Limitations of the knowledge base 153
7.5 The user model 154
7.6 The textual component 156
7.6.1 Initially selecting a strategy 156
7.6.2 Finding the main path 157
7.6.2.1 Marking the side links 162

            - 7.6.3 Implementation of the Strategies 163
                - 7.6.3.1 ATN Arc types 165
                - 7.6.3.2 Traversing the graph 167
            - 7.6.4 Stepping through the Constituency Schema 168
                - 7.6.4.1 Predicate Semantics 170
            - 7.6.5 The ATN corresponding to the Process Trace 173
            - 7.6.6 Choosing an arc 175
        - 7.7 The Interface 178
        - 7.8 The surface generator 186
            - 7.8.1 The functional grammar and the unification process 187
            - 7.8.2 TAILOR's grammar 188
            - 7.8.3 TAILOR's unifier implementation 190
        - 7.9 Issues pertaining to domain dependency 192
        - 7.10 TAILOR as a question answering system 193
- 8. Conclusions 195
    - 8.1 Main points of this thesis 195
    - 8.2 Feasibility and extensibility of this approach 197
    - 8.3 Directions for future research 199
    - 8.4 Conclusions 200
- Appendix A. Examples of texts studied 202
    - A.1 Texts from high school text books, junior encyclopedias and manual for novices 202
    - A.2 Texts from adult encyclopedias and the manual for experts 206
- Appendix B. Input/Output examples from TAILOR 209

List of Figures

Figure 1-1: Example of a patent abstract 6
Figure 1-2: RESEARCHER and the TAILOR System 7
Figure 1-3: Two descriptions of a microphone 9
Figure 1-4: The TAILOR System 10
Figure 1-5: Short description of a telephone 12
Figure 1-6: Description of a telephone 13
Figure 1-7: Description of a receiver 14
Figure 1-8: Description of a vacuum-tube 15
Figure 1-9: Description of a pulse-telephone 16
Figure 3-1: Description of a telegraph from an Encyclopedia [Collier's 62] 33
Figure 4-1: Rhetorical predicates used in this analysis 45
Figure 4-2: Two descriptions of the filament lamps 46
Figure 4-3: The constituency schema 48
Figure 4-4: Description of a telephone from an adult Encyclopedia 49
Figure 4-5: Description of transformers from an adult encyclopedia 52
Figure 4-6: Description of a telephone from a junior encyclopedia 54
Figure 4-7: Organization of the description of the telephone from a Junior Encyclopedia 56
Figure 4-8: Description of transformers from a junior encyclopedia 57
Figure 4-9: The process trace algorithm 59
Figure 4-10: Including a subpart's process explanation while explaining the object's function 61
Figure 4-11: Including a subpart's process explanation after explaining the object's function 62
Figure 4-12: Decomposition of the telephone example from the junior encyclopedia text into rhetorical predicates 63
Figure 4-13: Decomposition of part of the telephone example from the junior encyclopedia text into coherence relations 66
Figure 4-14: Decomposition of the telephone example from the junior encyclopedia text into nucleus/satellite schemata 68
Figure 4-15: Text from the *Encyclopedia of Chemical Technology* 72
Figure 5-1: The Constituency Schema as defined by [McKeown 85] 75
Figure 5-2: The modified Constituency Schema 76
Figure 5-3: The process explanation follows the main path from the start state to the goal state 81
Figure 5-4: The process explanation follows the main path from the goal state to the start state 81

Figure 5-5: Main path for the loudspeaker 85
Figure 5-6: An analogical side link can produce a clearer explanation 88
Figure 5-7: The side chain is long but related to the main path 91
Figure 5-8: Including a long side chain that gets re-attached to the main path 92
Figure 5-9: Including a causal side link does not render the explanation clearer 94
Figure 5-10: Including a short enabling condition 95
Figure 5-11: The side chain is long and not related to the main path 99
Figure 5-12: Substeps arising because of subparts 100
Figure 5-13: The Constituency Schema 102
Figure 5-14: Stepping through the Constituency Schema 103
Figure 5-15: The Process Trace 105
Figure 5-16: Including substeps and an isolated side link 107
Figure 5-17: Process trace for the dialing mechanism, including a side chain that gets re-attached to the main path 109
Figure 5-18: Example of a feedback loop 111
Figure 6-1: Representing the user model explicitly 115
Figure 6-2: The Constituency Schema strategy and its decision points 120
Figure 6-3: The Process Trace strategy and its decision points 120
Figure 6-4: Simplified portion of the knowledge base for the telephone 124
Figure 6-5: Combining the strategies: using the constituency schema as the overall structure of the text and switching to the process trace for one part 125
Figure 6-6: Starting with the constituency schema and switching to the process trace for the new part 127
Figure 6-7: Switching to the process trace for the superordinate and two parts 128
Figure 6-8: Starting with the process trace and switching to the constituency schema for one part 130
Figure 6-9: Changing the parameter that determines the overall structure of a description 131
Figure 6-10: Description of the telephone. *Most* is set to half of the functionally important parts 132
Figure 6-11: Combining the strategies 136
Figure 6-12: Combining the strategies, using an entry point other than the beginning 137
Figure 7-1: RESEARCHER and the TAILOR System 140
Figure 7-2: The TAILOR System 142
Figure 7-3: The knowledge base used in TAILOR; parts hierarchies 145
Figure 7-4: The knowledge base used in TAILOR; generalization hierarchies 146

Figure 7-5: An object frame 147
Figure 7-6: Representation of a microphone's function 148
Figure 7-7: Representation of events and links between events 150
Figure 7-8: Representation of a microphone 151
Figure 7-9: Several generalization trees 152
Figure 7-10: A characterization of the User Model in TAILOR 154
Figure 7-11: More examples of user models in TAILOR 155
Figure 7-12: The decision algorithm 157
Figure 7-13: Importance scale used to find the main path 158
Figure 7-14: Links between events 159
Figure 7-15: Finding the main path for the loudspeaker 160
Figure 7-16: Knowledge base for the loudspeaker 161
Figure 7-17: Examples of propositions obtained from traversing an arc of the constituency schema 165
Figure 7-18: Example of a proposition obtained from traversing an arc of the process trace 166
Figure 7-19: Constituency Schema and its ATN 169
Figure 7-20: Including more information than strictly required by the predicates 171
Figure 7-21: Predicate semantics 172
Figure 7-22: Stepping through the Constituency Schema 173
Figure 7-23: ATN corresponding to the Process Trace 174
Figure 7-24: Description using the Process Trace 176
Figure 7-25: When to include a relation 178
Figure 7-26: Interface input and output 179
Figure 7-27: Translation of the various propositions 181
Figure 7-28: Constructing a sentence from the identification predicate 182
Figure 7-29: Constructing a sentence from the attributive predicate 183
Figure 7-30: Combining simple sentences into a complex sentence 185
Figure 7-31: Using the pronoun *this* in a process explanation 186
Figure 7-32: Embedded clauses and their use in TAILOR 189

Acknowledgments

First and foremost I would like to thank my advisors, Michael Lebowitz and Kathleen McKeown, who have provided much help and encouragement during my entire stay at Columbia. Discussions with them were a fruitful source of ideas and inspiration, and their tireless reading (and re-reading) of my thesis and other work has considerably improved my writing style.

John Kender, Jim Corter and David Krantz have been very helpful members of my thesis committee. Their comments and insights are greatly appreciated. Steve Feiner also made useful comments on a draft of the thesis. I am also deeply indebted to Clark Thomspon and Joseph Traub without whom I might not have gone to graduate school.

Discussions of my work with friends and colleagues at Columbia has always been stimulating and enjoyable. Special thanks are due to Ursula Wolz, Kenny Wasserman, Michelle Baker and Larry Hirsch. I also want to thank TjoeLiong Kwee for his help with the functional unification grammar.

I also very much appreciated the support and encouragement of Dayton Clark, Jim Kurose, Betty Kapetanakis, Channing Brown, Kevin Matthews, Dannie Durand, Moti Yung, Stuart Haber, Galina and Mark Moerdler, and my officemates Don Ferguson and Michael van Biema. Sincere thanks go out to all of them. Yoram Eisenstadter deserves special thanks for his constant encouragement, support and advice.

Finally and most importantly, many thanks to my family, which has been great in encouraging my work and tolerating my complaints, especially my parents, my brother Guillaume, and our ''little friends.''

To my father

This research was supported in part by the Defense Advanced Research Projects Agency under contract N00039-84-C-0165, and the National Science Foundation grant ISI-84-51438.

1. Introduction

A question answering system that provides access to a large amount of data will be most useful if it can tailor its answers to each user. In particular, a user's level of knowledge about the domain of discourse should be an important factor in this tailoring, if the answer provided is to be both informative and understandable to the user. The answer should not contain information already known or easily inferred by the user, and should not include facts the user cannot understand. This thesis demonstrates the feasibility of incorporating the user's domain knowledge, or *user's expertise*, into a generation system and addresses the issue of how this factor might affect an answer. My results are embodied in TAILOR, a computer system that takes into account this knowledge level to provide an answer that is appropriate for users falling anywhere along the knowledge spectrum, from naive to expert.

1.1 Language generation and question answering

One of the aims of natural language processing is to facilitate the use of computers by allowing the users to communicate with the computer in natural language. There are two important aspects to man/machine communication: understanding a query from a user and answering it. Generation is concerned with the latter. It is recognized that providing an answer is a complex problem. In order to be effective, an answer must be:

- informative: it must contain information the user does not already know
- coherent: it must be organized in some coherent manner
- understandable: it must be stated in terms the user understands and contain information that the user will be able to grasp
- relevant: it must provide information that will help users achieve their goals.

A generation system needs to determine both *what to include* in an answer, and *how to organize* the information into a coherent text.

In a domain containing a great deal of information, deciding what to include in an answer is an especially important task as a system cannot simply state all the facts contained in the knowledge base about an entity, but rather must select the most appropriate ones. Organizing the selected facts is also a problem, since they cannot all be output at the same time. The problem of text organization has been referred to as ''linearization,'' for it involves placing the selected facts into a sequence. One way to organize facts in a coherent manner is to employ a discourse strategy to dictate the overall organization of a text. TAILOR uses two such discourse strategies to guide its generation process. Once the content and organization of the response has been decided upon, a generation system must translate the answer into natural language, deciding what lexical items should be used for the different concepts represented and what syntactic structures are required to express them. I am mainly concerned with the first two aspects of generation: determining the content and organization of a response in the context of a question answering system.

1.2 User modelling in generation

An answer appropriate for one user may not be adequate for another. People make use of their knowledge about other participants in a conversation in order to communicate effectively. Users who are allowed to pose questions to a system in natural language will tend to attribute human-like features to the system, expecting it to respond in the same way a person would [Hayes and Reddy 79]. If not too costly, it would clearly be desirable for a computer system to have knowledge about the user to approximate more closely human question answering behavior. This knowledge, contained in a user model, would aid a system in making various decisions required in the course of generating an answer.

A user model can contain a variety of facts about a user, including:

- *The user's domain knowledge.* This refers to what and how much background knowledge the user already has about the domain under consideration. To construct an answer that is not obvious to the user and does not assume knowledge the user does not have, a system needs to know about a user's domain knowledge.
- *The user's goal in asking a question.* The goal can modify the meaning of the question and its response. An appropriate answer is one that addresses the goal of the user [Hobbs-Robinson78].
- *Specific beliefs the user has about the domain.* These are the facts that currently happen to be true in the "world." This differs from the user's domain knowledge as it refers to facts the user knows about that are true *now* as opposed to facts the user knows about that are *always* true in the domain. Mutual beliefs of the speaker and the hearer can be used to plan the production of a referring expression that can be unambiguously understood by the hearer.
- *Past history of interactions.* Recording past interactions can help a system learn about the user.

In this work, I am mainly concerned with the user's domain knowledge.

The tailoring of answers according to domain knowledge is used extensively by humans. An explanation of how a car engine works aimed at a child will be different than one aimed at an adult, and an explanation adequate for a music student is probably too superficial for a student of mechanical engineering. There is further evidence of this phenomenon in naturally occurring texts, where the type of information presented to readers varies with their probable level of domain knowledge. (I present such evidence in a later chapter.) To approximate human question answering, a question answering program would need to take into consideration the user's domain knowledge.

The need for a model of the user's domain knowledge in question answering systems has been noted by various researchers [Lehnert 77; McKeown 82]. The programs that have modeled the user's domain knowledge, however, did so only in order to generate more or less detailed texts (e.g., [Wallis and Shortliffe 82; Sleeman 85]), assuming the level of detail was the only parameter to vary. They did not address the issue of whether or not this assumption was valid. I do address this problem, identifying the role played by the user's level of knowledge in determining an answer. My primary domain in investigating this problem concerns the description of complex devices.

1.3 Research method and main contributions

To determine how people describe complex devices and see whether these descriptions differ with the readers' assumed level of knowledge about the domain, I analyzed various naturally occurring texts. I looked at texts aimed at readers on the two ends of the knowledge spectrum: naive and expert. The text analysis indicated that the user's level of expertise affects the *kind* of information and not just the *amount of detail* presented. This result is significant as it demonstrates that level of detail is not the only factor in tailoring a response to a user's level of knowledge.

I characterized these results in terms of discourse strategies used to present texts to readers with different knowledge levels. One of these strategies[1], the constituency schema, is composed of linguistically defined predicates and was identified in previous work on generation by [McKeown 85]. This strategy is a *declarative* strategy, i.e., it is based on an abstract characterization of patterns occurring in many texts and is independent on the structure of the underlying knowledge base. Rather, it imposes a structure on the knowledge base. The other strategy, the *process trace*, is a new type of strategy that I term a *procedural* strategy. I have developed a precise formalization of the process trace. This strategy consists of *directives*, or directions on how to trace the knowledge base. The structure of a text generated using this strategy mirrors the structure of the underlying knowledge base in ways dictated by the strategy. In contrast, texts produced by declarative strategies (such as the

---

[1] Discourse strategies will be presented at length in Chapter 4.

constituency schema) mirror the abstract patterns represented in the strategies. These two strategies will be presented in detail in Chapter 5.

I show how these strategies can be combined to provide answers to users whose domain knowledge falls anywhere along the knowledge spectrum, from naive to expert. I have implemented them in TAILOR, a program that generates device descriptions with differing content for users with varying expertise.

In summary, the main contributions of this research have been to:

- identify and formalize a new type of strategy consisting of directives rather than linguistically defined predicates
- show the feasibility of incorporating the user's domain knowledge into a generation system
- add a new dimension in tailoring by varying the kind of information included in the text as opposed to the amount of detail
- be able to combine the strategies in a systematic way in order to tailor descriptions to a whole range of users without requiring an *a priori* set of user stereotypes. This ability also gives rise to a greater variety of possible texts.
- implement a computer system that generates descriptions of complex devices. (These descriptions can be lengthy.)

1.4 The domain

My domain is that of RESEARCHER, a program developed at Columbia University to read, remember and generalize from patent abstracts. The abstracts describe complex devices in which spatial and functional relations are important [Lebowitz 83a; Lebowitz 85]. An example of a patent abstract is shown in Figure 1-1. The knowledge base constructed from reading patents is large and detailed. This domain is a challenging one for language generation as there are several different kinds of information and many details from which to select facts to present to the user, rendering the decision process a complicated one. TAILOR, the generation system introduced in this thesis, produces natural language descriptions of

---

Patent: US # 3899794, 12 Aug 1975

Title: Front Loading Disc Drive Apparatus
Inventor: Brown Leon Henry, Sylmar, CA, United States
Wangco Incorporated (US Corporation)

Apparatus for receiving and driving magnetic disc cartridges as peripheral computer memory units. Particular mechanisms are included which render the apparatus more effective and more compact than previously known corresponding devices of a comparable nature. These mechanisms cooperate to provide means for inserting the disc cartridge in a horizontal attitude, permitting the apparatus to be completely contained within a reduced vertical dimension and thus saving substantial space. These mechanisms are operatively coupled to the loading door so that, as the loading door is rotated through approximately 60 degrees to its open position, a pair of actuators coupled thereto are rotated through approximately 90 degrees to first lift and then translate the disc cartridge receiver forward to its fully extended position. During this motion, various door opener levers which are associated with the receiver for the purpose of opening the head entry door of the disc cartridge to the extent necessary to permit entry of the heads therein when the cartridge and receiver are in the retracted position for operation within the disc drive apparatus are withdrawn so that the head entry door may be closed when the cartridge is withdrawn from the receiver. When the cartridge is inserted within the receiver, the head entry door is opened to a first extent by a pivoted bail member and the reverse of the above-described operations occurs as the loading door is closed so as to retract the receiver with the disc cartridge therein to the operating position.

Figure 1-1: Example of a patent abstract

---

devices from RESEARCHER's knowledge base.[2] Figure 1-2 presents a block diagram of the system. Upon receiving a request for a description, TAILOR uses discourse strategies to guide its decision process and examines both the knowledge base and the user model to determine the content and organization of the text to be generated.

---

[2] As the research for building the parser for RESEARCHER is being done at the same time as this research, the knowledge base has been coded by hand in some cases.

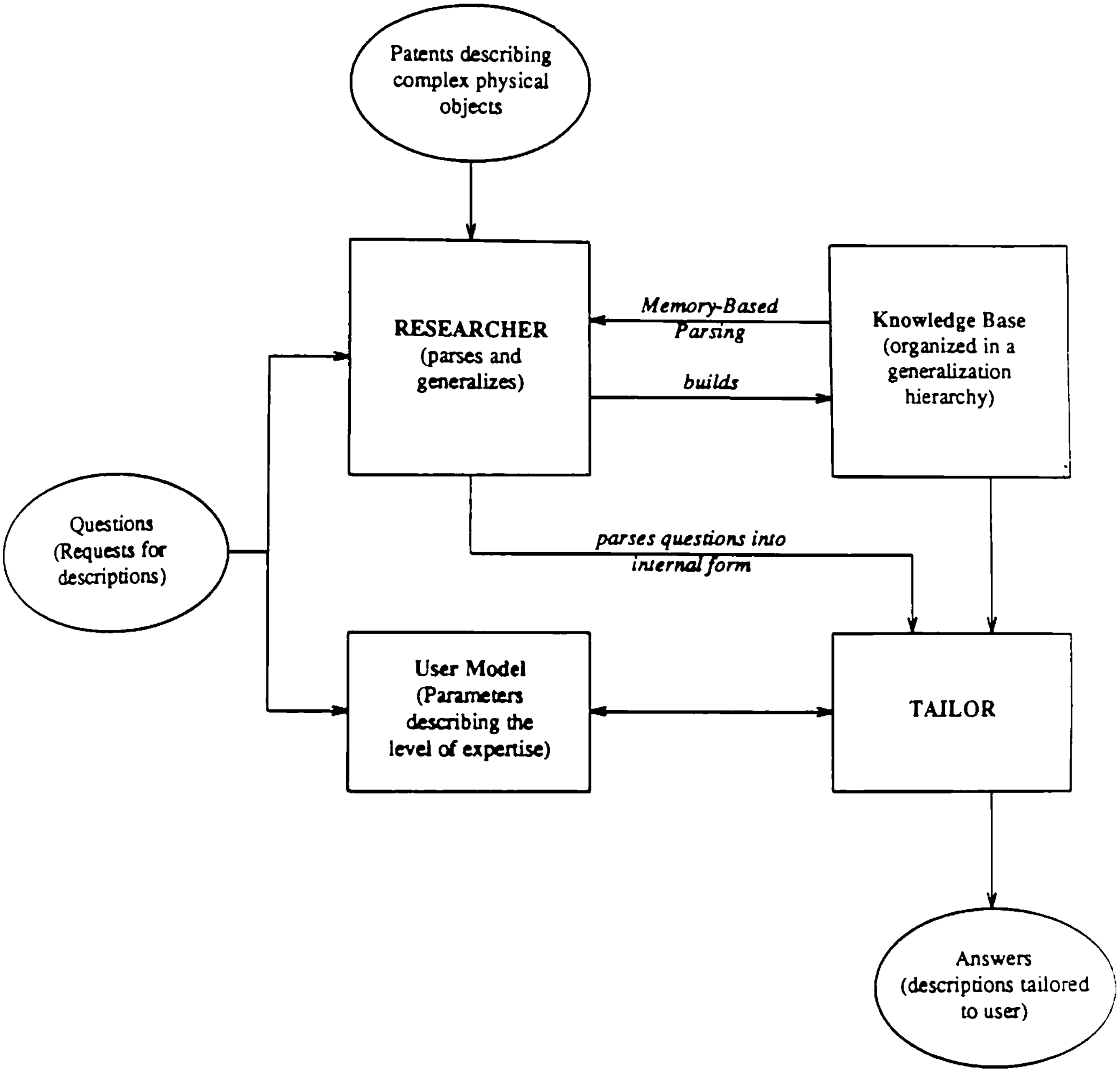

Patents describing complex physical objects
RESEARCHER (parses and generalizes)
Memory-Based Parsing
builds
Knowledge Base (organized in a generalization hierarchy)
Questions (Requests for descriptions)
parses questions into internal form
User Model (Parameters describing the level of expertise)
TAILOR
Answers (descriptions tailored to user)

Figure 1-2: RESEARCHER and the TAILOR System

The ability to generate descriptions is a good first step towards developing a question answering system for a knowledge base of complex devices for two reasons. First, users are likely to request object descriptions. Second, descriptions can also be used to answer other types of questions. For example, to compare two objects, it may be necessary to describe each of them as part of the text.

Generating descriptions is a difficult generation task in TAILOR's domain because a request for the description of an object cannot be answered by straightforward retrieval from the knowledge base. There are no clear constraints on what information should be included in the answer. This type of question is termed a *high-level question* [Tennant 78; McKeown 85]. To produce a description, a program cannot just state all the facts contained in the knowledge base about the object as there will typically be too many. A generation system will require guidance to select the appropriate facts to present to the user. Previous research efforts have developed discourse strategies to guide a system in choosing facts from a knowledge base in order to generate coherent texts (e.g., [McKeown 85]). In this domain, users will probably have different amounts of knowledge about the domain, so that coherence alone does not ensure that the text is the most appropriate one for a given user. Consider for example the two descriptions presented in Figure 1-3. These descriptions of a microphone were generated by TAILOR-87.

Both these descriptions present the information in a coherent manner but differ in content. Either may be appropriate for some user: the first one for a user who does not yet know how the microphone works, and the second for a user who is already familiar with the mechanism of the microphone. The second description would probably not be very informative to a user who did not know anything about microphones. A user model representing what the user presumably knows about the domain can thus help the system in choosing facts that the user understands and does

---

A microphone is a device that changes soundwaves into a current. Because a person speaks into the microphone the soundwaves hit the diaphragm of the microphone. This causes the diaphragm to vibrate. The diaphragm is made of metal and disc-shaped. When the intensity of the soundwaves increases the diaphragm springs forward. This causes granules of the button to be compressed. The compression of the granules causes the resistance of the granules to decrease. This causes the current to increase. Then, when the intensity decreases the diaphragm springs backward. This causes the granules to be decompressed. The decompression of the granules causes the resistance to increase. This causes the current to decrease. The vibration of the diaphragm causes the current to vary. The current varies, like the soundwaves vary.

A microphone is a device that changes soundwaves into a current. The microphone has a system to broaden the response and a metal disc-shaped diaphragm. The diaphragm is clamped at its edges. The system has a cavity and a button.

Figure 1-3: Two descriptions of a microphone

---

not already know (and cannot easily infer), thereby improving the resulting answer. The domain of complex devices is thus a domain very well suited to study of how a user's knowledge affects a description.

1.5 System overview

A block diagram of TAILOR is shown in Figure 1-4. TAILOR receives as input a request for a description and a set of parameters that describe a user's knowledge about the domain. This request is passed to the *textual component*. This component, the main concern in this work, determines the content and organization of the description to be generated. It is guided in its decision process by the user model and two discourse strategies. The two discourse strategies used in TAILOR are strategies that have been identified from text analyses. The strategies guide the system in

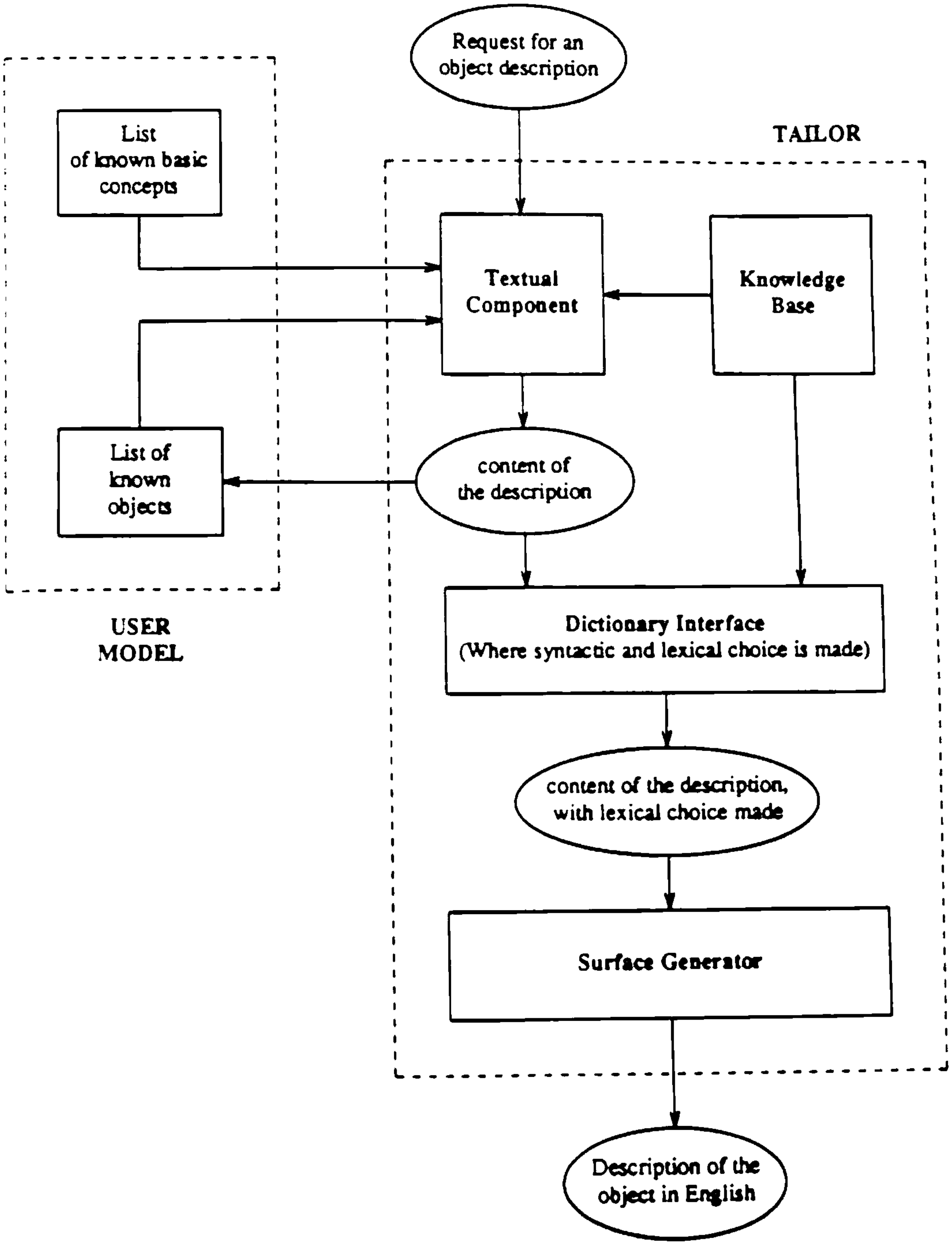

Request for an object description
TAILOR
List of known basic concepts
Textual Component
Knowledge Base
List of known objects
content of the description
USER MODEL
Dictionary Interface (Where syntactic and lexical choice is made)
content of the description, with lexical choice made
Surface Generator
Description of the object in English

Figure 1-4: The TAILOR System

choosing and organizing appropriate facts to include in a description. The strategy choice depends on the content of the user model.

The output of the textual component, a description in internal representation, is passed to the *tactical component*. The tactical component consists of a *dictionary interface* and a *surface generator*. The interface chooses the lexical items and syntactic structures. The surface generator produces English sentences using a functional unification grammar [Kay 79].

1.6 Examples from TAILOR

Sample texts generated by TAILOR are presented in the figures 1-5 through 1-9 below. Each one is preceded by a description of the user model for which the text was generated (i.e., a list of objects and concepts the user knows) and the name of the object being described. (More examples will be given throughout the thesis, as well as in the Appendix.)

1.7 Limitations

I do not examine the problem of determining how much the user knows about the domain, but take the user model as given. I will briefly discuss how it might be inferred in Section 3.2.

In order to focus on the role of a user's level of expertise in generation, other problems had to be ignored. I have not considered user characteristics other than domain knowledge, even though this is not the only factor which can influence an answer. In particular, I have not studied the influence of users' goals. Inferring the user's goal is another very hard problem. There has been much research on the subject [Allen and Perrault 80; Carberry 83; McKeown *et al.* 85]. It would, however, be interesting to study the interaction of the users' goals and domain knowledge in determining the content of an answer.

User Model:

Objects known?: *nil*
Concepts known?: *nil*

Describe telephone; (short description)

TAILOR-87 output:

```
 A telephone is a device that transmit soundwaves.
Because a person speaks into the transmitter of a
telephone a varying current is produced.  Then, the
current flows through the receiver.  This causes
soundwaves to be reproduced.
```

Figure 1-5: Short description of a telephone

In this work, requests for descriptions are the only type of questions studied. I believe that descriptions are a good starting point as they are required to answer other types of questions, but I feel that a user's domain knowledge will affect other types of questions as well.

I have made no attempt to parse questions from English input. (I will briefly discuss in Section 3 how a query posed to the system might also suggest the appropriate response level.) Moreover, while TAILOR does generate English sentences, I have studied neither the influence of a user's domain knowledge on lexical choice nor the complexity and subtleties of surface generation. My emphasis has been on deep generation.

User Model:

Objects known?: *loudspeaker, microphone*
Concepts known?: *nil*

Describe telephone;

TAILOR-87 output:

```
 A telephone is a device that transmit soundwaves.  The
telephone has a housing that has various shapes and
various colors, a transmitter that changes soundwaves
into current, a curly-shaped cord, a line, a receiver to
change current into soundwaves and a dialing_mechanism.
The transmitter is a microphone with a small disc-shaped
metal thin diaphragm.  The receiver is a loudspeaker with
a small aluminium diaphragm.  The housing contains the
transmitter and it contains the receiver.  The housing is
connected to the dialing_mechanism by the cord.  The line
connects the dialing_mechanism to the wall.
```

Figure 1-6: Description of a telephone

1.8 A guide to remaining chapters

In Chapter 2, I present an overview of related work in generation and user modelling. Even though I will not be addressing the problem of how to determine how much the user knows about the domain, I still have to know what *kinds* of knowledge a user possesses about a domain that can affect generation and be explicitly represented in a user model. This is described in Chapter 3. Instances of these kinds of knowledge will be the information contained in TAILOR's user model. Having identified what needs to be in the user model, I will take the user model as given, and study how a system can use the information contained in the user model to tailor the answer.

User Model:

Objects known?: *loudspeaker*
Concepts known?: *nil*

Describe telephone receiver

TAILOR-87 output:

```
A receiver is a loudspeaker with a small metal disc-
shaped diaphragm.  A receiver has a permendur ring-shaped
armature, a coil, a ring-shaped permanent magnet, a gap
and a small metal disc-shaped diaphragm.  The diaphragm
is mounted on the poles of the magnet.  The gap contains
air and it is between the diaphragm and the poles.  The
coil is mounted around the magnet.
```

Figure 1-7: Description of a receiver

In Chapter 4, I present the text analysis, showing how texts aimed at two distinct audiences (expert and naive) are organized differently and present different types of information to their readers. I also introduce the two distinct discourse strategies that TAILOR uses to describe complex devices. Each of these strategies is discussed in detail in Chapter 5. TAILOR can combine the two strategies to describe objects to users with intermediate levels of expertise, and, because of the explicit representation employed for the user model, TAILOR can generate descriptions tailored to a whole range of users, without requiring an *a priori* set of user types. This is explained in Chapter 6. TAILOR's implementation is presented in Chapter 7, and, finally, Chapter 8 presents a discussion of the feasibility of this approach, directions for future work as well as a conclusion.

User Model:

Objects known?: *nil*
Concepts known?: *electricity, voltage*

Describe a vacuum-tube

TAILOR output:

```
  A vacuum-tube is a device that produces a strong
current from a power-source across the anode and the
cathode.  Because a heat source heats the cathode
electrons accumulate on the cathode.  The cathode is made
of metal.  Electrons accumulating on the cathode enables
electrons to flow towards the anode.  Because the power-
source produces voltage across the anode and the cathode
electrons flow towards the anode.  This causes strong
current across the anode and the cathode to be produced.
```

Figure 1-8: Description of a vacuum-tube

User Model:

Objects known?: *microphone, loudspeaker*
Concepts known?: *electricity*

Describe a pulse telephone

TAILOR output:

```
 A pulse-telephone is a telephone with a pulse dialer.
The pulse-telephone has a pulse-dialer that produces
current pulses, when a person dials, a housing that has
various shapes and various colors, a receiver to change a
current into soundwaves, a curly-shaped cord, a line and
a transmitter that changes soundwaves into a current.
The pulse dialer is a dialing mechanism.  Because a
person dials the dial assembly of the pulse dialer turns
clockwise.  This causes the spring of the pulse dialer to
be compressed.  The spring is circular.  The compression
of the spring enables the spring to be decompressed.
Because the person dials, the person releases the dial of
the dial-assembly.  This cause the spring to be
decompressed.  The decompression of the spring causes the
dial assembly to turn counterclockwise.  This causes the
gear of the pulse-dialer to turn.  The gear is small.
The dial-assembly turns counterclockwise proportionally
to the way the gear turns.  Because the gear turns the
protrusion of the gear hits the lever of the switch.
This causes the lever to close the switch.  Because the
lever closes the switch current pulses are produced.  The
receiver is a loudspeaker with a small metal thin disk-
shaped diaphragm.  The dialing-mechanism is connected to
a wall by the line.  The housing is connected to the
dialing mechanism by the cord.  The housing contains the
transmitter and it contains the receiver.  The
transmitter is a microphone with a thin disk-shaped small
diaphragm.
```

Figure 1-9: Description of a pulse-telephone

2. Related Research

This work involves both generation and user modelling. This chapter presents other research in these areas, some of which have more emphasis on the user model, others on generating answers taking a user model into consideration. As user modelling is a new but growing field, I will present an overview of the work in this area. Although many researchers are working on generation, I will only discuss generation work that is closely related to this thesis, either in the text analysis employed to derive discourse strategies or because the decision process utilizes both a user model and discourse strategies. Research in reading comprehension and psychology is also of interest, as it provides insight into what might make an answer more understandable to users with different knowledge levels.

2.1 Related work in user modelling and generation

User modelling problems include the task of constructing and organizing a model. Constructing a user model can be done either by collecting information from a user, inferring facts from a dialog, or a combination of both. User modelling also includes issues of exploiting the user model to improve the system's answering abilities. All these aspects are important, and an ideal system would incorporate all of them. In this section, I present some of the major research that addresses these issues, starting with Rich's work, as it has been the basis for many other systems.

2.1.1 Superposing stereotypes

Rich [79] showed how a model of the user can be built by refining and intersecting various stereotypes and how a system can use such a model to tailor its answers to a user. GRUNDY, a system simulating a librarian, utilized this method to suggest books to its users.

GRUNDY had a generalization hierarchy of stereotypes, each containing a set of characteristics. Associated with a stereotype were *triggers* that signalled the appropriate use of a stereotype. Stereotypes were activated through these triggers when users were asked to describe themselves by typing a few words. Because of the generalization hierarchy, one stereotype could also activate another. The user model was built up by combining the characteristics of the active stereotypes. The user model thus contained a set of characteristics, taken from the active stereotypes. A *justification*, indicating from which stereotype the facet was borrowed, was associated with each characteristic, in case the system needed to remember how the information was derived.

Once the model was built, GRUNDY used it to select a book to present to the user. The most salient characteristics of the user were selected, and one was chosen at random to serve as a basis for selection. As the objects in the knowledge base (books) also had attributes that corresponded to the facets of the users' stereotypes, a set of books matching the chosen characteristic was selected. Each book of the set was then evaluated against the other salient characteristics of the user, and the best match was presented to the user.

GRUNDY also examined the user model to decide which aspects of the book to mention when presenting the book to the user. If the book was refused, GRUNDY would attempt to understand why by asking the user which characteristic of the book was disliked. Based on the answer, GRUNDY would try to alter the user model by changing the inappropriate characteristic.

In building GRUNDY, Rich was mainly interested in building the user model. My emphasis in this work differs from hers, as I am not interested in *building a user model*, but in *determining an answer based on a user model.* The user model employed in TAILOR is very different from the one used in GRUNDY, as it contains explicit information about the user's domain knowledge instead of various facets borrowed from stereotypes. (This will be presented in detail in Chapter 3.) Stereotypes have the disadvantages of being rigid and arbitrary. The ones set by the system implementer cannot easily be redefined. Stereotypes, however, might be useful as initial approximations of the user model which can be used until more detailed and explicit knowledge about the user can be gathered.

The way TAILOR decides what to present to the user differs from GRUNDY's since it is not based on attributes attached to items in the knowledge base. TAILOR relies on no specific information in the database to tell it what is appropriate for a given type of user. Rather, it uses a more complex set of criteria to choose relevant facts to present to the user, based on a characterization of what type of knowledge is appropriate in light of the user's domain knowledge.

2.1.2 Modelling and using the user's domain knowledge

Wallis and Shortliffe have used the naive/expert distinction in their work on providing explanations in the domain of medical expert systems [Wallis and Shortliffe 82]. The inference rules employed by the expert system were given a complexity factor, and users were assigned expertise levels. To generate explanations, the causal chain corresponding to the system's behavior was passed to the generator. The complexity measure of each rule in the chain was matched against the user's level of expertise to determine whether the rule should be included in the explanation or not. This procedure resulted in giving more or less detail depending on the user's domain knowledge.

Sleeman developed UMFE, a user modelling front end to be used to tailor expert systems' explanations [Sleeman 85]. As in [Wallis and Shortliffe 82], UMFE receives from an expert system the causal chain of inference rules which were activated in deriving a conclusion. The rules are assigned complexity and importance factors. UMFE determines which rules to present to the user based on these factors and the expertise level of the user. The emphasis in UMFE is on determining the level of sophistication of the user. This is done both by questioning the user and by employing inference rules. These rules relate concepts to each other based on their complexity factors to suggest additional concepts the user might know. These rules allow UMFE to ask the user a minimal number of questions.

The chains of inference rules employed by these expert systems are similar to the links used in TAILOR to generate a process trace. In both the program developed by Wallis and Shortliffe and UMFE, however, unlike in TAILOR, the content of the answer has already been decided upon by the time the user model is examined. The user model is utilized mainly to decide on the amount of detail to include in the explanation. The issue of whether the level of detail is the only important parameter to vary is not addressed. This is precisely the issue I confront in this work.

The CADHELP system, which serves as an interface to a computer aided design system, is also sensitive to some extent to the user's level of expertise as it is verbose with a new user and omits information as the user gains experience with the system [Cullingford *et al* 82]. CADHELP does not keep a user model *per se*, but only remembers the previous discourse. There is no characterization about what kind of information should be included for which type of user.

The HAM-ANS system has a model of the user's knowledge which is mainly used for resolution and production of anaphora [Jameson and Wahlster 82; Hoeppner *et al.* 84]. When asked a question, the system attempts to produce the smallest unambiguous answer possible. By using the system's ellipsis and anaphora resolution component (with a feedback loop) and the user model, the system checks whether a potential ellipsis or anaphora will be understood by the user, given the user's knowledge about the discourse. If the system determines that the answer can be understood in the current context, the answer is produced. Otherwise, the system tries to elaborate on its answer. TAILOR differs from HAM-ANS in that it uses its user model to decide on the *content* of an answer and rather than *phrasing*.

Chin is concerned with modelling and obtaining the user's domain knowledge about the UNIX system [Chin 86]. His system, KNOME, is part of UC, the UNIX Consultant [Wilensky *et al.* 84]. KNOME uses stereotypes for both the users and the knowledge base, which is a set of UNIX commands. Stereotypes for the commands in the knowledge base include *simple*, *mundane* and *complex*, while users are divided into four groups: *novice*, *beginner*, *intermediate* and *expert*. Each user category is expected (with some certainty factor) to know about some class(es) of commands. Unlike UMFE, KNOME does not ask the user any questions but tries to deduce the user's domain knowledge from what the user includes (or does not include) in a question posed to UC. To do this, KNOME relies on both the stereotype system and a few inferencing rules about what the user is likely to know.[4] KNOME infers the user's level of expertise by combining all the evidence it has about which facts the user knows or does not know. UC employs KNOME to decide how to answer a question, typically by omitting from the answer what it assumes the user already knows. For instance, UC does not include the example associated with a command when explaining the command, unless the user has been determined to be novice. While KNOME's double stereotype system seems to be successful in the UC domain, it is not as applicable in the domain of complex devices, where it is hard to partition the knowledge base into a few categories and decide that knowing about one type of

---

[4] Nessen [86] describes a system similar to KNOME, but in which the user model is continuously updated.

objects implies more expertise than knowing about another set of objects. For example, there is no reason to believe that knowing about microphones indicates more expertise about the domain of complex devices than knowing about telescopes. Another approach is thus required. Furthermore, I wanted to be able to tailor answers to users whose domain knowledge level falls anywhere along a knowledge spectrum without having to classify users into a few discrete stereotypes. Finally, in this work, I am more concerned with exploiting the user model whereas Chin is concerned with building it.

2.1.3 Using knowledge about the user's plans and goals to generate responses

A great deal of research is being conducted on determining users' plans and goals and using them to understand incomplete or incorrect sentences and generate helpful responses. Although I do not address this issue here, the user's goals can also play an important part in deciding what to include in an answer. Indeed, an answer for a user whose goal is to buy an object should include different kinds of information than an answer for a user who wants to repair this object. The ability to detect and address users' goals and plans is important and would need to be included in a full question answering system. I will therefore give a brief summary of the research done in this area of user modelling and generation, beginning with that of Allen and Perrault.

Allen and Perrault [80] examined the problems of generating appropriate responses to questions by inferring the questioner's goal. They showed that, by keeping a model of the questioners' beliefs and by being able to infer their plans and goals, a system can provide helpful and cooperative answers, as it can detect obstacles in the users' plans and provide information that will help accomplishing the desired goal. They developed a method that enables a system to derive the user's beliefs and goals. Using this method, a question answering system can build a user model containing the user's goals and beliefs and use it to answer questions in a cooperative fashion. The types of cooperative answers a system would be able to generate using this model include direct and indirect answers, as well as answers containing more information than requested in the question.

To detect the user's goals and plans, a system needs domain knowledge that includes *plans* and *goals* users may have in the domain of discourse, a formulation of *actions*, which have preconditions, substeps and effects, and *beliefs* and *wants* (intentions). In their system, Allen and Perrault used a standard planning formalism to represent plans and goals [Fikes and Nilsson 71], in which given an *initial state of the world* W and a goal G, a *plan* is a sequence of actions that transform W into G. Plans were domain specific and were used to derive the goal of a questioner. Because this knowledge was represented explicitly, the system was able to reason about what the user needed to know in order to achieve a goal. This fact is important since a system appears to be cooperative when it is able to provide information that will help the user achieve a goal.

Research on plans and goals and their use in cooperative discourse has continued since Allen and Perrault's work. Further plan inferencing models have been developed to allow for more complex sets of goals and plans [Carberry 83; Sidner 85; Litman 86; Carberry 87]. Morik has been looking at a similar problem, that of modelling a user's wants in order to produce cooperative responses [Morik 85; Morik 86]. With more emphasis on how to use the goal to select relevant information to present to the users, McKeown [85] and van Beek [87] generate explanations tailored to the users' goals, plans and intentions in a student advisory domain. Finally, many researchers are examining the problem of recognizing that a user's plan is incorrect and correcting it [Sidner and Israel 81; Pollack 86; Carberry 87; Quilici 87].

2.1.4 Using reasoning about mutual beliefs to plan an utterance

Appelt's generation system, KAMP, embodies a formal representation of the speaker's and hearer's mutual beliefs and uses a formal planning system to plan and produce utterances. KAMP was developed in a task domain where an expert is helping a novice assemble some piece of equipment. One of Appelt's emphases was on producing referring expressions that could be understood by the hearer. KAMP reasons about the knowledge of the speaker and hearer to make sure that, when producing an utterance, the speaker believes it will be understood by the hearer given both their beliefs. Axioms are used to prove that a generation plan formed by KAMP is correct, in that it will satisfy the speaker's goals, which must include being understood by the hearer. KAMP uses knowledge about the goals to be achieved and linguistic rules about English to produce sentences that satisfy multiple goals. KAMP relies on its planning system not only to plan the utterance, but also to generate English.

Although KAMP tailors an utterance according to the hearer's knowledge, the flavor of this work is different from TAILOR's. KAMP is very goal-oriented, and utterances are produced to satisfy the speaker's goals. The limited task domain provides a constrained framework for the utterance. The point during the assembly at which the dialog is taking place provides a constraint on the utterance, as there is usually one step to be accomplished at that time. The speaker, or program, need only produce one or two sentences corresponding to the next step in KAMP's plan. In TAILOR, there are no such constraints. Since the generator needs to select facts from the knowledge base to present to the user, the user model provides the framework that delineates a subset of the knowledge base to include in the text.

Hovy's generation system, PAULINE, incorporates the speaker's interpersonal goals towards the hearer to produce utterances with different content depending on various pragmatic situations. PAULINE mixes sentence planning and realization, allowing these goals to influence both the content and the phrasing on the sentence [Hovy 85; Hovy 87]. Unlike TAILOR, PAULINE does not take into consideration the user's domain knowledge in planning an utterance.

2.1.5 Dealing with misconceptions about the domain

Another important aspect of user modelling is to detect and correct users' misconceptions about a domain. Kaplan, Mays and McCoy address these issues for different classes of misconceptions and with different emphasis. Kaplan's system, CO-OP, deals with misconceptions that depend on the *content* of the database [Kaplan 79], while Mays designed a model aimed at recognizing misconceptions depending on the *structure* of the database [Mays 80a]. While the thrust of both Kaplan's and Mays' work was in detecting misconceptions, McCoy [86, 87] examined the problem of correcting misconceptions . She characterized in a domain independent manner the influences on the choice of additional information to include in answers. She also identified discourse strategies a system can use to produce answers correcting the misconceptions. McCoy's work is the most similar to mine, as she is concerned with exploiting user models and her generator employs discourse strategies.

Instead of relying on an *a priori* list of possible misconceptions, as do some Computer Assisted Instruction systems [Stevens *et al.* 79; Brown and Burton 78; Sleeman 82], McCoy classified object related misconceptions based on the knowledge base *feature* they involve. A feature of the knowledge base could be a *superordinate* relation or an *attribute*. Through studies of transcripts, she has identified what types of additional information should be contained in the answer corresponding to each type of object related misconception. A correction schema that dictates what kind of information to include in the answer is associated with each type of misconception. To vary answers depending on the *context* of the misconceptions, McCoy also allows for different ''object perspectives.'' The strategy chosen to correct a misconception is therefore dependent both on the misconception type and the active object perspective.

The present research differs from this body of work because I am not addressing the issue of detecting and correcting incorrect users' views of the domain, but am interested in providing an answer that is optimally informative given how much the user knows about the domain.

2.2 Related work in decomposing texts

To study the discourse strategies used in describing complex devices, I analyzed naturally occurring texts, decomposing them into rhetorical structures and identifying patterns that occurred across the texts. These patterns can be used to guide a generation system in choosing and organizing facts to construct a text. The main method for such text analysis is to decompose a text in terms of its rhetorical structure, identifying common combinations of predicates: the text is decomposed into different propositions (or clauses), and each proposition is classified as a type of *rhetorical predicate.* Rhetorical predicates are the means available to a speaker to present information. They characterize the structural purpose of individual clauses.

2.2.1 Decomposing a text using linguistic rhetorical predicates

In her work on generation, McKeown [85] used rhetorical predicates as defined by linguists [Shepherd 26; Grimes 75]. McKeown studied the problems of what to say when there are many facts to choose from and how to organize a text coherently [McKeown 85]. She examined texts and transcripts in order to determine whether there were standard patterns of discourse structure used in naturally occurring texts. Where patterns of discourse structure could be identified for various discourse goals, they were used to guide a generation system in choosing and organizing facts to construct a text. By decomposing the texts into their rhetorical structure, she found that some combinations of predicates were more likely to occur than others. Moreover, for each discourse situation (such as providing a *definition*), some combination was the most frequent. McKeown encoded these standard combinations as schemata that are associated with a particular discourse situation. The schemata may contain alternatives. Yet, they constrain the order in which the predicates appear. I have used McKeown's approach in my text analysis, and, in fact, TAILOR uses one of the schemata she posited. This approach will be presented at length in Chapter 5.

2.2.2 Decomposing a text using coherence relations

Another way to decompose a text that might give rise to a useful organizing structure is in terms of the coherence relations identified in [Hobbs 78]. In attempting to formalize the notion of coherence in a discourse and decide what makes a discourse coherent, Hobbs identified relations that relate the current utterance to the previous discourse. In constructing a text (or discourse), a speaker must decide in which way to expand the previous discourse segment. This decision is reflected in the coherence relation that connects the new sentence to the previous utterances. Hobbs defined four kinds of coherence relations:

- a *temporal relation* relates a sequence of events temporally, or causally.
- an *evaluation relation* provides a relation between two other segments of discourse.
- a *linkage relation* links a presumably unknown fact to what is already known to the hearer. Linkage relations include *background* and *explanation*.
- an *expansion relation* allows the speaker to go from the general to the specific, the specific to the general, or move from one specific instance to another one (either to elaborate or to contrast).

These relations are useful to point out the relationships among the sentences in the text. This top-down approach is very goal-oriented, as coherence relations reflect a communicative goal the speaker is trying to achieve. Many different coherence relations can be chosen at any point during the discourse construction, depending on the speaker's goals. As we will see in Chapter 5, it is hard to use these coherence relationships for text generation without constraints on how the relations should be arranged.

2.2.3 Decomposing a text with rhetorical structure theory

It is also possible to obtain a text decomposition using the nucleus/satellite schemata defined by Mann [84a]. Using these schemata, a text is decomposed into segments (or *text spans*), including a *nucleus* and one or more *satellites*. The nucleus segment serves to accomplish a goal the writer/speaker has in mind, while a satellite usually supports the claim given in the nucleus segment. The nucleus and satellites are connected with relations, such as *background* or *elaboration*. The schemata are recursive, so that each text span can be further decomposed into other schemata. Unlike McKeown's schemata, Mann's schemata do not restrict the textual order of the relations, and any schema can be expanded into any other at any point. To use these schemata for generation, one needs a way to dictate which relation to include when in order to provide a clear text. A control strategy must also determine which schema to expand, in which way to expand it, which satellite to choose, etc. Therefore, these schemata are mainly descriptive at this point, as are coherence relations.

2.3 Related work in psychology and reading comprehension

Much research has been conducted in psychology about aspects of man-machine interaction and the distinction between novices and experts. In studying the differences between novices and experts, researchers have mainly looked at the differences in learning style between these two groups, how memory is (re)organized as people acquire knowledge and how one goes from being a novice to becoming an expert. While not directly addressing the issue of how to tailor answers to users having different amounts of domain knowledge, this body of research is of interest as it confirms the validity of the method proposed here.

Of particular interest for this work is a study done by Egan and Gomez [82, 83] where they analyzed how individual differences affect difficulty in learning a text editor and showed how the amount of difficulty experienced by users is strongly correlated with user characteristics. Their study suggests that individual differences are very important and should be taken into consideration in designing systems, where it might be appropriate to display information differently depending on the users' characteristics. In particular, a user's level of expertise should be taken into consideration, which is exactly what this work proposes.

More directly connected with user's level of expertise are studies by Chi *et al* and Lancaster and Kolodner. In a study of categorization and representation of physics problems by experts and novices, Chi and her colleagues found that these two classes of students used very different ways of classifying physics problems [Chi *et al* 81]. Experts tended to use abstract physics principles, while novices used the problem's literal features, possibly indicating that novices lacked knowledge about physics principles. In a study in which problem solving capabilities were explored for users with different levels of expertise about the domain, Lancaster and Kolodner found that expert users have more knowledge not only about individual parts of complex devices but also about the causal models involved and the interconnections among parts [Lancaster and Kolodner 87]. Both these studies suggest that varying only the amount of information might not be enough to tailor an answer to a user's domain knowledge. In this work, another dimension along which to vary an answer is provided, namely the kind of information included in the answer.

Finally, research in reading comprehension emphasizes the importance of previous knowledge in comprehending a text. Davison [84] criticizes readability formulas, which are strictly based on syntactic structure and choice of vocabulary, claiming they do not adequately measure text difficulty or reading level because they do not measure the background knowledge a text requires in order to be understood. She further argues that the lack of background knowledge is often what makes a text hard to comprehend and shows how readers may fail to understand a text if they do not have required knowledge that is implicitly assumed.

Other researchers have also indicated that readers use their previous knowledge in order to understand new texts [Schank and Abelson 77; Anderson *et al.* 77]. The role of *schemata* (organized knowledge units in memory) in understanding a text and making inferences to complete the meaning of a text is emphasized in [Wilson and Anderson 86]. In this article summarizing research on the role of prior knowledge in understanding a text, Wilson and Anderson also point out that readers can fail to understand a text mainly because the text assumes knowledge that they do not have.

The results described above suggest that, in order to tailor a text or a response to a user's level of knowledge, it is not enough to simply use different words and grammatical constructions nor to vary only the amount of detail provided in an answer. What the user knows about the domain should play a significant role in deciding what to include in the answer by influencing the kind of information to present to the user. This is exactly what the method proposed in this work does.

2.4 Summary

This chapter presented the related research in generation, user modelling, psychology and reading comprehension. This work draws upon research in generation by using previously defined methods to analyze naturally occurring texts in order to construct discourse strategies. It is different from previous work on user modelling as it addresses the role of the user's domain knowledge in tailoring an answer, an issue not specifically addressed before. Research in psychology and reading comprehension provide further support for the approach proposed here.

3. TAILOR's user model

The *user model* contains all the knowledge a system has about the user. In TAILOR, the user model is *explicitly* encoded and contains exclusively information about the user's domain knowledge, as it is the focus of this work. This chapter describes the user model employed by TAILOR and explains how the information contained in the user model might be obtained.

3.1 Identifying what needs to be in the user model

Knowledge is multidimensional, in the sense that it is possible to be knowledgeable about a domain along several dimensions, such as historical, economical, functional, etc. In identifying the kinds of knowledge a user might have, however, I restrict myself to the types of knowledge that are explicitly represented in terms of our knowledge base.

Analysis of natural language texts suggests the existence of at least two kinds of domain knowledge that affect generation in this domain:

- *knowledge about specific items in the knowledge base.* I define ''knowing'' about an object to mean knowing about the existence of the object, its purpose and how this purpose is achieved (that is, how the various subparts of the object work together to achieve it). Knowing about an object thus means understanding the functionality of the object and the mechanical processes associated with it.
- *knowledge about various basic underlying concepts.* In a domain of complex physical objects, such concepts might include *electricity* and *voltage.*

Consider for example the text in Figure 3-1 The writer of this text must have assumed that the reader would know what a *switch* is, as well as what it meant to be *in series* and to *act as the return line*, thus understanding the concept of electricity. The text would probably have been different had the intended readers not have that knowledge.

Telegraph

A telegraph consisted of a key (switch) and an electromagnet (sounder) at each end, in series with a wire and a battery. The earth acted as the return line.

Figure 3-1: Description of a telegraph from an Encyclopedia [Collier's 62]

I define an *expert* user as one whose knowledge about the domain includes functionality of most objects and mechanical processes. An expert user does not necessarily know about *all* the objects contained in the knowledge base however, and thus, might ask questions about objects he or she does not know about. Given an object that is new but similar to a known one, an expert user has enough domain knowledge to infer how the parts of this new object work together to perform a function. For example, the expert user might already know about particular instances of this object, or, on the other hand, about generalizations of the object. A *naive* user is one who does not know about specific objects in the knowledge base and does not understand the underlying basic concepts.

A user is not necessarily naive or expert, however. For example, users may know about several objects in the knowledge base. Such users would not be considered naive, but, as there are many objects they do not know in the domain, they would not be considered experts either.

The level of expertise can be seen as a continuum from naive to expert. To really know and understand some of the objects in the knowledge base, a user needs to first understand the basic underlying concepts. When users have mastered these concepts, they become progressively more expert as they acquire knowledge about more objects in the knowledge base. Most users would be falling along the continuum of domain knowledge. Any user might lack knowledge about specific parts in the domain but understand various basic concepts. For example, a user may not know anything about telephones or similar devices, but still have knowledge about electricity and voltage. Although some combination of the two types of knowledge might not be realistic, the generation program that takes the user's domain knowledge into consideration ought to be able to handle any user model.

In this work, I will not attempt to determine some number of stereotypes that exist between the two extremes and categorize a user into one of these user types. Instead, I use explicit parameters to indicate the user's knowledge. These parameters correspond to the two kinds of knowledge outlined above. They are:

- a list of items in the knowledge base which are known to the user, representing the user's *local expertise*.
- a list of underlying basic concepts that the user understands.

As an example, a user model in TAILOR may indicate that the user only has local expertise with respect to disk drives. The terminology *naive* and *expert* is retained only for users at the two ends of the knowledge spectrum, only as a short-hand notation. The emphasis of my work is on studying how object descriptions can be varied when content of the user model varies.

By explicitly representing elements of a user's knowledge, I gain the flexibility needed to tailor descriptions to specific users falling anywhere along the knowledge spectrum, without requiring a predefined set of stereotypes. This will be explained in detail in Chapter 6.

3.2 Determining the level of expertise

Although the issue of determining the level of expertise of the user is not addressed in this work, it is obviously an important question that needs to be studied. Because the user model representing the user's level of expertise defined above is rather a coarse grained model, I believe that it may be possible to infer it from various factors, as outlined below.

3.2.1 User type

Before the program has been able to gather specific information about a user, user types might provide a good *initial heuristic*, as a starting point for building a more adequate user model. This section presents some user types that may provide some initial approximations of the user's domain knowledge. Other methods, described in the following sections, can be used subsequently to refine the user model, thus avoiding the problem of using strictly stereotypes.

Some classes of users may be likely to be naive while others may be likely to be expert. For example, three plausible categories of potential users have been identified for RESEARCHER:

- Inventors who created a device and want to check what has already been patented in the domain of their inventions. Inventors are experts in their field, but may be novices in other fields and as far as doing a patent search is concerned.
- Lawyers who perform patent searches for their clients. Unlike the inventors, they are experts with respect to doing a patent search, but may be novice with respect to the contents of the patents.
- General users who are people who want to know what kind of information is available from the database. Such a person could be either an expert or a novice.

Other user types might also be helpful to give *a priori* information on the probable level of expertise, as a class of users may be likely to know about some subset of objects. For example, students in astronomy are likely to know about telescopes, but they won't necessarily have any knowledge about any other objects in the knowledge base. On the other hand, an electrical engineering student is likely to understand concepts such as electricity and know about objects such as microphones. Identifying a user as part of these classes can give insight into how much that user might know, and provide a starting point for building the user model. This is similar to the approach taken in GRUNDY [Rich 79], although, in GRUNDY, the user model is formed only of stereotypes. This approach is also proposed by Cohen and Jones for building a user model in an educational diagnosis expert system, where psychologists might know one aspect of the knowledge base, while teachers might know another [Cohen and Jones 87].

3.2.2 Role of the memory organization

Intuitively, a person who talks only in very general terms about an object probably does not know much about it. Quite the contrary, knowledge about an *obscure* part of an object usually indicates that a person has some expertise about the domain. The specificity of a question can thus provide information about the user's domain knowledge. As an example, consider the questions Q1 and Q2.

Q1: How is the IBM 3380 disk drive different
from the IBM 3330 disk drive?

Q2: How are the three spaced bearings of a
disk drive head assembly connected to
the tracks ?

If a user knows about the IBM 3380 disk drive, it is likely that he or she has some knowledge about disk drives in general. Likewise, it is certainly not obvious to everyone that a head assembly should include "three spaced bearings" and therefore we might assume that the user has some knowledge about head assemblies. The bearings constitute *obscure* parts of a head assembly, and the fact that the user knows about them indicates a certain level of expertise in the domain.

Memory organization can play a role in determining that a part is obscure. Memory in RESEARCHER is organized in terms of generalizations [Lebowitz 83a; Lebowitz 83b]. As it reads patents, RESEARCHER looks for similarities among the objects described to make generalizations from them, and organize its memory around those generalizations. The resulting memory is largely hierarchical, consisting basically of several generalizations trees with individual instances at the leaves. The top node in the generalization tree contains information common to all instances of that generalization, while instance nodes only contain information special to that instance.

This complex, hierarchical structure can help determine which parts are obscure. When a user is knowledgeable about a part occurring deep in the generalization tree (the extreme case being a part which is particular only to an instance at the bottom of the tree), we might assume expertise in that subdomain.[5] On the other hand, if only parts at a top level node of the generalization tree are mentioned (i.e., the user knows only about information common to a whole class of objects), then we are probably dealing with a novice in that particular subdomain. This technique is helpful in determining the level of expertise of the user in any application using a hierarchical knowledge base.

Similarly, the depth in the components tree of the knowledge base can help

---

[5] This assumes that the knowledge base is fairly complete, in that it contains more than one object in the generalization tree.

evaluate the user's level of expertise, where a part occurring deep in the parts tree might be more obscure than a part occurring at a higher level.

3.2.3 Question type and detecting misconceptions

The kind of question asked might also give further information about how much the user knows. Consider for example the two questions:

Q1: How many bearings are there?

Q2: What is a bearing?

In Q1, the user most probably knows what a bearing is. In Q2, however, it is clear that this is not the case. The distinction must therefore be made between *knowing* and *mentioning* an object, and we must be careful in applying a method like the one outlined above for determining whether an object is obscure or not. The question type must be used in conjunction with the depth in the knowledge base.

Furthermore, a term may be misunderstood as in "Which filters have flying heads?," when the knowledge base indicates that a "filter" cannot have a "flying head." In this case, the user has a misconception. Techniques like those developed by Kaplan [82] and Mays [80b] can be applied to detect the misconceptions. The misconception can be used to infer information about the user's level of expertise.

3.2.4 Inference rules and the radius of expertise

Expertise in a subpart of a domain does not necessarily imply expertise in the whole domain. As an example, a user could be an expert in the domain of disk drives without knowing much about the computers that use them (or vice versa: a user could know about computers without really knowing much about disc drives). On the other hand, expertise about microphones might imply expertise about loudspeakers, as their mechanisms are similar. Knowing that the user is familiar with some items of the knowledge base might allow a system to infer expertise about other items. It is thus possible to have some inference rules that would allow a system to define an area of expertise for a given user, or a *radius of expertise*. These inference rules would relate items and concepts, as in [Sleeman 85]. A system could also have more general rules. For example, when a user mentions a part deep in the generalization tree, expertise with respect to a number of objects related to that part can be established. Or when an item is known to the user, it might be possible to infer that its superordinate in the generalization hierarchy is also known to the user. Similarly, it might be possible to infer knowledge about certain basic concepts, knowing the user has local expertise about some particular entities. TAILOR uses two general inference rules of this kind: when the user model indicates local expertise about an object, TAILOR assumes that the superordinate and subparts of the object are also known.

3.2.5 Asking the user questions and using the previous discourse

It is possible to ask the user a few questions about himself to get a starting point for the user model. This is the approach take in both GRUNDY [Rich 79] and UMFE [Sleeman 85] for example. After the discourse has begun, it may also be advantageous to use the past discourse to update the user model. In TAILOR, for example, once the system has provided a description to a user, it adds the object just described to the user model, assuming the user now knows about the object. While this approach is naive (as it assumes a perfect learner), it allows for the user model to get built up as the discourse progresses. It is possible to relax the assumption of a perfect learner. For example, if a user asks the same question twice, it is indicative that the answer was not understood, perhaps because it assumed knowledge the user did not have. The user model can be updated correctly to reflect this fact.

3.3 Conclusions

In this Chapter, TAILOR's user model has been presented, together with suggestions on how this model might be obtained from various sources (e.g., past discourse). It would, of course, be necessary to study in depth how the user's domain knowledge can be inferred from the sources outlined above and how these sources can be used together. While this is a hard problem, aspects of it have already been addressed in previous research (refer to 2.1.2 for a review of such research), and I believe that it is possible for a system to obtain the user model required by TAILOR.

4. The research approach and the theoretical results

4.1 Introduction

This chapter presents the research approach I employed to determine what strategies are used in human-authored texts to describe complex devices, and whether these strategies differ depending on the assumed level of expertise of the intended readers. In general, discourse strategies are best studied by analyzing texts written by people or conversations among two (or several) agents. I chose the former, because of the availability of texts describing objects. To see how the strategies vary depending on a reader's assumed level of expertise, I looked at a variety of texts, including encyclopedia entries, from both adult and junior encyclopedias, high school text books and manuals. In this chapter, I present examples of the texts studied, show how I analyzed the texts to capture their organizational structure, and present the two distinct discourse strategies found to describe complex devices.

4.2 Discourse strategies and their role in natural language generation

In order to generate paragraph-long answers, a system must both select facts to present to the user and organize them in a coherent manner. Discourse strategies, which characterize the structure of texts, have been identified, formalized and used successfully in computer systems to guide the generation of texts in deciding what to say and how to organize it [McKeown 82; McKeown 85; Mann 84a; McCoy 86; Kukich 85; Weiner 80].

Researchers have identified discourse strategies by decomposing naturally occurring texts with a common discourse goal into rhetorical structures and identifying patterns that occur across the texts. These patterns, which may contain options, are then captured formally in a discourse strategy for the discourse goal of the texts studied. One commonality of the strategies that have been proposed thus far is that the structure of the generated text is based on an abstract characterization of patterns occurring frequently in texts and is not primarily dependent on the underlying knowledge base from which the facts stated in the text are drawn. I term such strategies *declarative strategies*. Although the generated text is dependent on the knowledge base to the extent that the text can only contain facts included in the knowledge base, the structure of the text reflects more the abstract pattern embodied in the declarative strategy than the structure of the knowledge base.

Declarative discourse strategies are typically composed of *rhetorical predicates*, which are the means available to a speaker to present information. Rhetorical predicates characterize the structural purpose of individual clauses and have been discussed by a variety of linguists (e.g., [Shepherd 26; Grimes 75]) and computational linguists (e.g., [Hobbs 78; Hobbs 80; McKeown 82; Mann 84a]). Researchers construct declarative strategies by identifying the combinations of rhetorical predicates that are preferred over others in naturally occurring texts. For example, McKeown analyzed a variety of texts and found that certain combinations of predicates were associated with discourse purposes such as providing definitions, comparisons, and descriptions [McKeown 82]. Mann identified about 35 rhetorical nucleus/satellite combinations found in a diverse collection of texts, ranging from administrative memos to newspaper articles [Mann 84a]. Similarly, McCoy identified combinations of rhetorical predicates that are appropriate for correcting different types of user misconceptions [McCoy 86].

4.3 The texts analyzed

To develop effective strategies for tailoring a description to a particular level of expertise, I began by studying descriptions in a variety of naturally occurring texts: adult encyclopedias [Britannica 64; Collier's 62], junior encyclopedias [Britannica-Junior 63; New Book of Knowledge 67; Encyclopedia of Science 82], manuals [Chevrolet 78; Weissler 73] and high school text books (e.g., [Baker *et al.* 57; Verwiebe *et al.* 62]). These texts were chosen because they provide a good source of descriptions and because they seem to address audiences at the two ends of the knowledge spectrum: naive and expert. Texts from adult encyclopedias are directed at an audience much more knowledgeable in general than the audience addressed by high school text books and junior encyclopedias. Likewise, the Chevrolet repair manual is aimed at knowledgeable users (professional mechanics) while the other one claims to be directed towards novices. These texts thus constitute a good starting point for studying the differences between the descriptions given to naive users and those given to experts in a domain.

I studied descriptions of devices, taking the description of the same object in all sources whenever possible. This allows for good comparisons of the texts, both in terms of content and organization. To minimize the effects of stylistic differences on my results, texts from at least two different encyclopedias in each audience category were chosen. I examined about fifteen examples from each encyclopedia and textbook and a few from the manuals. The descriptions studied were generally several paragraphs in length.

Given the desire to focus on how a reader's assumed domain knowledge can affect a description, encyclopedia texts have an added advantage besides providing examples of descriptions. They are directed to a general audience and people turning to them do so for a variety of reasons. Yet, they all read the same texts and thus acquire the same information, regardless of their goal for reading these texts. Encyclopedia texts are not directly aimed to help achieve particular goals. Rather, they provide general information intended to be useful to achieve any goal a reader may have. Thus an encyclopedia must provide its readers with some information about an object, without knowing or attempting to address a reader's specific goals.

This property of encyclopedias gives me the opportunity to focus on how the level of expertise affects a description, without considering how the goals and specific beliefs of the reader could affect it. While a sophisticated question-answering system should take all these users characteristics into consideration in order to decide on an answer and it would be necessary to study how they affect each other, understanding these factors separately first will help us study their interaction later.[6] In this work, being more concerned with the role played by the user's domain knowledge, I simply assume that users want descriptions that allow them to build functional models of the object.

4.3.1 The textual analysis

I began by analyzing the different texts using methods developed by other researchers (e.g., [Hobbs 78; Hobbs 80; McKeown 82; Mann 84b]) to compare their organizational structures, hoping to find consistent structures in each group of texts. (See Section 2.2 for a description of these methods). My aim was to formalize these structures as discourse strategies to guide the generation process.

Following McKeown's approach, the texts were decomposed into different propositions (or clauses), and each proposition was classified as a type of predicate. The predicates were taken from [McKeown 85] and were based on Grimes's and Shepherd's definitions [Shepherd 26; Grimes 75]. They are shown in Figure 4-1.

The analysis showed that the texts fell into two groups: most of the descriptions in the adult encyclopedias entries and in the car manual for mechanics were organized around object subparts and their properties, while the descriptions in the junior encyclopedia entries and in the car manual for novices traced process information.

---

[6] Utilizing the user's goals and beliefs to provide an appropriate response is a hard problem and has been the focus of extensive research. See Section 2.1 for a summary of this research.

1. Identification: Description of an object in terms of its superordinate.
   Example: This bear is a panda bear.
2. Constituency: Description of the subparts or subentities.
   Example: The telephone consists of a transmitter, a receiver and a housing.
3. Attributive: Associating properties with an entity.
   Example: Beth's teddy bear is black and white.
4. Cause-effect: A cause-effect relationship between two events or relations.
   Example: The soundwaves strike the diaphragm and cause it to vibrate.
5. Analogy:
   Example: The X-ray tube is very similar to the cathode ray tubes you have been studying except that the electron beam is aimed at a metal target instead of a glass screen.
6. Elaboration:
   Example: The diaphragm was originally invented by Thomas A. Edison.
7. Renaming:
   Example: The current goes through the coil, called the 'primary coil'.

Figure 4-1: Rhetorical predicates used in this analysis

Consider for example the descriptions of filament lamps shown in Figure 4-2. The first text in this figure is from the *Collier's Encyclopedia* and the second from the *New Book of Knowledge (The Children's Encyclopedia)*. The first text concentrates on providing the parts of the filament lamp, while the other description concentrates on process information (e.g., an explanation of what happens in a lamp).

In the first sentence of the text for adult readers, the parts of the filament lamp are immediately introduced, with some attributes. In the text for the junior audience, after an prefatory sentence to introduce the term "filament lamp," the author explains what happens. Only one part gets explicitly mentioned as part of this explanation ("a thread").

from *Collier's Encyclopedia*, an adult encyclopedia

[Collier's 62]

Typical household incandescent lamps (general service) are constructed with the following parts: a coiled tungsten filament, a glass bulb to keep air out and inert gases in, and a base that serves as a holding device and connects the filament to the electric supply. These three parts vary in size and shapes with each different class of lamps, such as general service, reflector, showcase, streetlighting, automobile sealed beam, miniature flash lights and photograph lamps.

from the *New Book of Knowledge*, a junior encyclopedia

[New Book of Knowledge 67]

The type of electric lamp made by Edison is called a filament lamp. A filament lamp lights when a thread inside it heats up to incandescence - that is when it heats so brightly that it gleams with light.

Figure 4-2: Two descriptions of the filament lamps

These texts are typical of those found in the different sets of sources. The main difference between these two types of descriptions is in the kinds of information they provide (details about parts versus details about processes), although there are also some differences in the surface structure (that is, the vocabulary and syntax used). As the main concern of this work is in deciding what to include in a text and how to organize the facts coherently, the differences in syntactic structure and vocabulary will not be addressed.

Since two types of descriptions were found in texts, it is clear that there is more than one way to describe an object. A system that generates device descriptions should be able to provide these different kinds of descriptions. Furthermore, these two types of descriptions give a generation system another dimension along which it can tailor its answers to a user's level of domain knowledge: instead of simply varying the amount of detail provided, a system can also vary the kind of information.

To be able to generate these two types of descriptions, it is necessary to identify and formalize their different organizational structures. The difference between the two kinds of texts is captured in terms of two distinct discourse strategies: the *constituency schema* and the *process trace.*

4.3.2 Analyses of entries from adult encyclopedias and the car manual for experts

The texts from the adult encyclopedias and the Chevrolet manual can be characterized in terms of the *constituency schema*, a discourse strategy posited in [McKeown 85]. These texts provide details about the parts of an object and their properties (attributes).

In McKeown's analysis, the constituency schema was associated with the goals of providing definitions and describing available information. This is also the goal TAILOR has when providing a description. The constituency schema, which will be presented formally in the next chapter, is a declarative strategy that can be characterized by the four steps indicated in Figure 4-3.

Some of these predicates can be expanded into a schema, resulting in the recursive use of the schema. For example, instead of using the depth-attributive predicate for each subpart of an object, it is possible to use the constituency schema recursively to provide more details about each part.

To show how a text might be decomposed into rhetorical predicates and how a pattern can be abstracted from such decomposition, I present detailed analyses of some of the texts studied. To see how the descriptions from the adult encyclopedias

1. [Identify the object as a member of some generic class, using the *identification* predicate][7]
2. Present the constituents of the item to be defined (subparts or sub-entities), corresponding to the *constituency* predicate
3. Present characteristic information about each constituent in turn, corresponding to the *depth-attributive* predicate
4. [Present additional information about the item to be defined, corresponding to the *attributive* predicate.]

Figure 4-3: The constituency schema

match the constituency schema, the classification of the different propositions into rhetorical predicates is indicated in the Figures.

Figure 4-4 shows a text taken from the *Collier's Encyclopedia* that describes a telephone. In the first sentence, the telephone is described in terms of its subparts: the transmitter, the receiver and the housing. This proposition corresponds to the constituency predicate, the second step of the schema. The first step of the schema, identification in terms of a generic class, was omitted. Following this sentence, the two main subparts are described in turn: the transmitter, in sentences 2 through 8 and the receiver in sentences 9 to 13. Providing information about the subparts of an object is termed giving *depth-attributive* information. In this text, both parts are described in detail with a recursive call of the schema instead of one predicate, and depth-attributive information includes the use of both the constituency and attributive predicates.

For example, depth-attributive information about the transmitter is included in

---

[7] The steps included in square brackets are optional.

Text	**Predicates**
1) The hand-sets introduced in 1947 consist of a receiver and a transmitter in a single housing available in black or colored plastic.	Constituency (with attributive)
	Depth-Attributive for the transmitter:
2) The transmitter diaphragm is clamped rigidly at its edges	*Attributive*
3) to improve the high frequency response.	*Cause-effect*
4) The diaphragm is coupled to a doubly resonant system	*Attributive*
5) -a cavity and an air chamber-	*Constituency*
6) which broadens the response.	*Cause-effect*
7) The carbon chamber contains carbon granules,	*Constituency*
8) the contact resistance of which is varied by the diaphragm's vibration.	*Attributive /cause-effect*
	Depth-Attributive for the receiver:
9) The receiver includes a ring-shaped magnet system around a coil and a ring shaped armature of anadium Permendur.	*Constituency (with attributive)*
10) Current in the coil makes the armature vibrate in the air gap.	*Cause-effect*
11) An attached phenolic-impregnated fabric diaphragm,	*Attributive*
12) shaped like a dome,	*Attributive*
13) vibrates and sets the air in the canal of the ear in motion.	*Cause-effect*

Figure 4-4: Description of a telephone from an adult Encyclopedia

sentences 2 to 8. In (2), an attribute of the ''diaphragm'' is given, here a structural relation (e.g., ''the diaphragm is clamped''), followed by a cause-effect relation in (3) that justifies the attribute provided in (2) (e.g., ''it is clamped *to improve the high frequency response*''). In (4), more attributive information about the diaphragm is presented, still a structural relation, introducing a new subpart, the ''doubly-resonant system,'' again followed by a cause-effect relation in (6). The components of the newly introduced part, the doubly-resonant system, were also immediately introduced in (5) (e.g. ''a cavity and an air-chamber'').

Note that sometimes two predicates are combined in the surface structure into one clause. As an example, the first sentence clearly corresponds to the constituency predicate, as the parts or constituents of the telephone are given. A strict categorization would, however, decompose this sentence into first the constituency predicate, corresponding to the listing of the parts, and then the depth-attributive predicate, since attributes (properties) of the ''housing'' are provided: ''available in black or colored plastic.'' The two predicates were then merged in the surface structure. The same information could have been conveyed using two sentences: one using the constituency predicate followed by another using the attributive predicate to further describe the housing, as in: ''The hand-sets consist of a receiver, a transmitter and a housing. The housing is available in black or colored plastic.'' Immediately including properties results in a more concise text, however. A similar phenomenon appears in (9), where the magnet is described as being ''ring-shaped'' and ''around a coil,'' and the armature as being ''ring shaped'' and made of ''anadium Permendur.''

This effect can be achieved with a sophisticated surface generator capable of merging two predicates into a sentence. Although TAILOR does not have such an elaborate surface generator, it is able to mimic this phenomenon in the implementation of the predicate semantics by retrieving more information from the knowledge base than strictly required by the predicates. In other words, instead of merging two predicates at the surface structure level, the program will anticipate the use of two predicates one after the other and immediately retrieve the information for both. For example, in TAILOR's implementation, the semantics of the constituency predicate allow mentioning the parts together with several properties. This is in anticipation that the depth-attributive predicate (that can retrieve properties) will be chosen after the constituency predicate. This feature will be described in more detail in Chapter 7, where TAILOR's implementation is presented.

Another example of a description from an adult encyclopedia, shown in Figure 4-5, is a description of transformers from the *Encyclopedia Britannica*. The transformer is first identified as a member of a super class: ''inductors.'' This corresponds to the identification predicate, the first step of the schema. Identification here also includes the functionality. A description of the transformer's parts follows: first the parts are given (''two or more windings''), and depth-attributive information about the windings is provided in (5). The description then returns to the top level object being described, the transformer, and the paragraph ends by presenting properties associated with transformers. Two causal links are presented in this description. Instead of explaining how the transformers function, however, these links serve as justification for a property that was just given. For example, the cause-effect relation included in sentence (9), ''to minimize energy losses,'' justifies the property presented in sentence (7), ''is laminated,'' but does not explain how transformers transform alternating current.

The descriptions presented above, like the lamp description from the adult encyclopedia presented in Figure 4-2, are organized around the subparts and attributive information of the objects being defined, even though a causal link is occasionally presented. Attributive information includes spatial and structural

Text	Predicates
1) Transformers are special forms of inductors widely used in electronics	Identification
2) to transform alternating power to one or more circuits at the same frequency, but usually with changed values of voltage and current.	function (cause-effect)
3) They operate on the principle of electronic induction.	attributive
4) All transformers consist of two or more windings	Constituency
5) so arranged that a change of flux in one winding induces a change of flux in the other.	Attributive
6) They are usually provided with a high-permeability material	Attributive /cause-effect
7) to increase the efficiency of the flux linkage and	Cause-effect
8) this material is laminated or otherwise divided	Attributive
9) to minimize the energy losses that would occur in solid material.	Cause-effect

Figure 4-5: Description of transformers from an adult encyclopedia

relations, as in "the diaphragm is coupled to a doubly resonant system," and properties, as in "shaped like a dome." When a causal link is provided, it serves as a justification of a property just mentioned. For example, the causal link "to improve the high frequency response" in the description of the telephone (in Figure 4-4) elaborates on the structural relation "the diaphragm is clamped" and justifies this relation; it does not really explain the mechanism of the telephone.

4.3.3 Texts from junior encyclopedias, high school textbooks, and the car manual for novices

While most of the texts from the adult encyclopedias and the Chevrolet car manual can be characterized using the constituency schema, texts from the junior encyclopedias, high school textbooks and the car manual for novices could not be described using the analysis methods of earlier work. No known schema or other organizing structure consistently accounted for these descriptions. Moreover, identifying the rhetorical predicates of the sentences did not provide insights to a useful organizing structure. As I will show in the examples, the structure resulting from using rhetorical predicates is one that contains mainly cause-effect relations. Although this structure is a simple schema, it is not useful for generation as it does not provide enough constraints in deciding what to include in a text.

In looking for other types of organizing strategies, I discovered that the main strategy used in these descriptions was to trace through the processes that allow the object to perform its function. I termed this strategy a *process trace* and have developed a precise formulation, consisting of *directives*, or instructions, rather than predicates. This strategy is not an abstract pattern formed of rhetorical predicates, as is the constituency schema. Instead, the strategy is more like an algorithm that follows the underlying knowledge base very closely and gives directives on how to trace it to select textual content. I term this type of strategy a *procedural* strategy.

The description of the telephone in Figure 4-6 is taken from the *Britannica Junior Encyclopedia* [Britannica-Junior 63]. In this description, the author starts by explaining what happens when someone speaks into a telephone (sentence 1), and why a diaphragm located inside the transmitter vibrates (3). While introducing the diaphragm as part of a causal link, the author also includes properties about the diaphragm: it is ''aluminium'' and ''disc-shaped.'' Now that the diaphragm has been

Text	Predicates
1) When one speaks into the transmitter of a modern telephone, these sound waves strike against an aluminium disk	Cause-effect
2) or diaphragm	renaming
3) and cause it to vibrate back and forth	Cause-effect
4) in just the same way the molecules of air are vibrating.	Comparison between two actions
5) The center of this diaphragm is connected with the carbon button	Attributive
6) originally invented by Thomas A. Edison.	Elaboration
7) This is a little brass box filled with granules of carbon	Attributive
8) composed of especially selected and treated coal.	Attributive
9) The front and back of the button are insulated.	Attributive
10) The talking current is passed through this box so that the electricity must find its way from granule to granule inside the box.	Cause-effect
11) When the diaphragm moves inward under the pressure from the sound waves the carbon grains are pushed together	Cause-effect
12) and the electricity finds an easier path.	Cause-effect
13) Thus a strong current flows through the line.	Cause-effect
14) When a thin portion of the sound waves comes along, the diaphragm springs back,	Cause-effect
15) allowing the carbon particles to be more loosely packed, and	Cause-effect
16) consequently less current can find its way through.	Cause-effect
17) So a varying or undulating current is sent over the line whose vibrations exactly correspond to the vibrations caused by the speaker's voice.	Cause-effect between two actions
18) This current then flows through the line to the coils of an electromagnet in the receiver.	Cause-effect
19) Very near to the poles of this magnet is a thin iron disc.	Attributive
20) When the current becomes stronger it pulls the disc toward it.	Cause-effect
21) As a weaker current flows through the magnet, it is not strong enough to attract the disk and it springs back.	Cause-effect
22) Thus the diaphragm in the receiver is made to vibrate in and out....	Cause-effect

Figure 4-6: Description of a telephone from a junior encyclopedia

introduced as part of the process, it is described more fully in sentences (5) through (9) with various properties and structural relations. The description then reverts to tracing the process information, explaining how the soundwaves are transformed into varying current and describing how the telephone achieves its function of transmitting soundwaves.

As the vibration process in (3) can be decomposed into two substeps (i.e., "the diaphragm moves inward" and "the diaphragm springs back"), the author traces each one in turn (sentences (11) through (13) for the first substep, and (14) through (16) for the second). Sentence (17) summarizes these substeps, also indicating that the process trace will now continue at the top level, not the substep level. Sentence (18) continues the process trace by describing where current flows next.

In the remaining text, the author explains how the varying current is changed back into soundwaves in the receiver of the telephone. Again, the author traces through substeps when a process step can be divided: "the current strength varying" can be divided into the "current becomes stronger" (sentence 20) and "the current becomes weaker" (21). Each substep is traced in turn. The text organization for this description is summarized in Figure 4-7.

The organizing principle of this text is the mechanical process underlying the function of the telephone. The decomposition of the text into predicates, as shown in the second column in Figure 4-6, emphasizes that the text is *primarily*, although not exclusively, formed of causal links. In this text, the process description gets interrupted when descriptive information can be included about a subpart that was just mentioned as part of the process description. For example, attributive information about the diaphragm was provided after the diaphragm was introduced in the causal links. Furthermore, not only is the description made mainly through a

Text	Structure
1) When one speaks into the transmitter of a modern telephone, these sound waves strike against an aluminium disk 2) or diaphragm and 3) cause it to vibrate back and forth 4) in just the same way the molecules of air are vibrating.	Process trace at top level: *one speaks* ==> *soundwaves strike diaphragm* ==> *diaphragm vibrates*
5) The center of this diaphragm is connected with the carbon button 6) originally invented by Thomas A. Edison.	Attributive information about the diaphragm, a part just introduced during the process trace
7) This is a little brass box filled with granules of carbon 8) composed of especially selected and treated coal. 9) The front and back of the button are insulated.	Attributive information about the button
10) The talking current is passed through this box so that the electricity must find its way from granule to granule inside the box.	Process at top level: *current flows through the box*
	Diaphragm vibrates can be decomposed into two substeps: - diaphragm moves inward - diaphragm springs back
11) When the diaphragm moves inward under the pressure from the sound waves the carbon grains are pushed together and 12) the electricity finds an easier path. 13) Thus a strong current flows through the line.	Process trace for the first substep: *diaphragm moves inward* ==> *carbon grains pushed together* ==> *more current*
14) When a thin portion of the sound wave comes along, the diaphragm springs back, 15) allowing the carbon particles to be more loosely packed, and 16) consequently less current can find its way through.	Process trace for the second substep: *diaphragm springs back* ==> *carbon grains looser* ==> *less current*
17) So a varying or undulating current is sent over the line whose vibrations exactly correspond to the vibrations caused by the speaker's voice.	Summary; Back to tracing process at top level
18) This current then flows through the line to the coils of an electromagnet in the receiver.	Process trace at top level
19) Very near to the poles of this magnet is a thin iron disc.	Attributive information about the magnet
	Again, there are two substeps: - current increasing - current decreasing
20) When the current becomes stronger it pulls the disc toward it.	Process trace for the first substep: *current increases ===> disc is pulled*
21) As a weaker current flows through the magnet, it is not strong enough to attract the disk and it springs back.	Process trace for the second substep: *current decreases ===> disc is released*
22) Thus the diaphragm in the receiver is made to vibrate in and out....	Summary

Figure 4-7: Organization of the description of the telephone from a Junior Encyclopedia

process trace, but this process trace can be given in great detail by explaining substeps if there are any. The information contained in a description aimed at naive users corresponds to the causal links that connect the various processes contained in the knowledge base.

The description of transformers from the *Britannica Junior Encyclopedia* is shown in Figure 4-8. In this description, the author first explains the principle behind a transformer's mechanism in sentences (1) through (4). This causal explanation forms the emphasis of the description. The description continues with two illustrating examples in sentences (5) through (8), which still refers to the function of the transformer. Finally, after the parts (the coils) have been introduced, their properties are presented in sentences (9) through (12).

---

1) If two coils of wire are properly arranged close to one another, they will make a transformer. 2) When an alternating current is sent through one of the coils, called the 'primary coil', an alternating current will be created, or induced, in the second or 'secondary coil'. 4) The voltage of the current in each coil depends on the number of turns of wire around the coil. 5) For example, if the primary coil has 100 turns and secondary coil has 500 turns, the voltage of the current in the secondary coil will be five times as great as the voltage of the current in the primary coil. 6) Such a transformer is called a 'step-up' transformer. 7) By winding fewer turns of wire around the secondary coil, the voltage is decreased. 8) This makes a 'step-down' transformer. 9) The primary and secondary coils of transformers for power circuits are wound around an iron core. 10) The ''iron'' in the core is actually many layers of thin steel strips. 11) The same core serves for both windings. 12) Thin strips are used because magnetism can not change rapidly in a solid iron core and also because a solid core would be heated by the magnetism.

Figure 4-8: Description of transformers from a junior encyclopedia

---

The theme of the texts from this group has been the description of the function and the mechanical processes associated with the object being described. In this type of description, the object is mainly described in terms of the functions performed by its subparts in order to explain how the object achieves its purpose. The description traces through the process information instead of enumerating the object's subparts along with their attributes, as was the case in the descriptions matching the constituency schema. The object's subparts are mentioned only when the description of the mechanical processes involves them. As a result, not all the subparts are mentioned. For example, in the description of a lamp from a junior encyclopedia (shown in Figure 4-2), the base was not mentioned. Similarly, in the description of the telephone (shown in Figure 4-6), the housing was left out. When a part is mentioned during the process trace, the process description sometimes gets interrupted to include descriptive information concerning that subpart.

In the texts for adults and experts, the description were organized around parts, and process information was included mainly as a justification or elaboration of a property (or a part) just mentioned, not as an explanation of the object's function. In the texts for the junior audience, the reverse is true: a part is mentioned only when it is involved in the process explanation, and information about parts is provided after a part has been introduced during the process trace. The description is constructed around process information. The preceding observations can be formalized in the definition of the process trace. This strategy, which is described in more detail in the next chapter, is summarized in Figure 4-9.

This strategy is not an abstract pattern of predicates but rather an algorithm that dictates how to trace the knowledge base. Following this strategy, the emphasis of the description remains on the main path. This results in a coherent explanation of how the object performs its function. As indicated in step 3, it is possible to provide more information about a subpart, once it has been introduced as part of the process trace, as was done in the two examples presented in this section. As will be explained

1. Find the main sequence of events that take place when the object performs its function. I call this sequence the *main path.* I will explain in the next chapter how the main path can be obtained.

   Starting from the first event in this sequence:
2. Follow the next causal link on the main path.
3. [Give attributive information about a subpart just introduced][8]
4. [Include a side link, that is a link not on the main path, though related to it]
5. [Follow the substeps if there are any]
6. Go Back to (2).

Figure 4-9: The process trace algorithm

in Chapter 6, this step of the strategy can be replaced by a call to the constituency schema for a subpart, if a fuller structural description is desired. In step 4, it is possible to include a side link, that is a link that is not on the main path. This will be discussed in the next chapter. Finally, when a step can be divided into substeps, the causal explanation can be continued at the substep level, as indicated in step 5. Substeps happen, for example, in the following case: the step " ... causes the diaphragm to vibrate" can be divided into the two substeps: "the diaphragm moves inward" and "the diaphragm moves outward." In the description of the telephone, the causal explanation continued at the substep level. Substeps can also arise when a complex object is made of several other complex parts.

Substeps can be followed either when a decomposable step is encountered (as in the description of the telephone), or after traversing the main path entirely. The latter

---

[8] Brackets denotes optional steps.

appears desirable when there are many substeps, each leading to a long sequence of events, to avoid distracting the reader's attention from the main path for a long period. It also seems more desirable when the substeps arise because a subpart itself has process information associated with it. In that case, first an explanation of how the object parts work together to achieve the object's function is given. Then, if a long description is desired, it is possible to step through each of the parts, describing how it achieves its own function. This is similar to the schema recursion for the constituency schema in [McKeown 85]. Figure 4-10 illustrates how including a long subpart's process explanation while still explaining the top level object's function might be confusing as it might distract from the top-level object's process information. Figure 4-11 shows the description of the same object, with the subpart's process trace included after the main object's process trace. In both Figures, the process information for the subpart is shown in italics.

4.3.4 Need for directives

In Figure 4-6, I showed the decomposition of the description of the telephone from a junior encyclopedia into rhetorical predicates. The text and its decomposition into rhetorical predicates is repeated in Figure 4-12. This decomposition helped determine that this text was mainly composed of causal links. However, the structure resulting from this decomposition is an introductory sentence followed by a pattern of cause-effect links, as can be seen in the second column of Figure 4-6. A schema can be formed from this decomposition, namely a schema comprising simply causal links.

Such a schema is not very useful as an organizing strategy for generation as it cannot sufficiently guide the generation system in its decision process. Indeed, there may be quite a few causal links in the knowledge base and such a schema does not

---

Description of a tape recorder[9]

The tape recorder is a machine that records, stores, and reproduces sound. A tape recorder has a microphone which picks up sounds. The microphone changes sounds into electric currents. *The microphone converts soundwaves into current. When a person speaks into a microphone, the soundwaves strike against a diaphragm and cause it to vibrate back and forth in just the same way the molecules of air are vibrating. The talking current passes through a carbon chamber filled with granules, so that the electricity must find its way from granule to granule inside the box. When the diaphragm moves inward under the pressure from the soundwaves the carbon grains are pushed together and the electricity finds an easier path. Thus a strong current flows through the line. When a thin portion of the soundwaves comes along, the diaphragm springs back, allowing the carbon particles to be more loosely packed, and consequently less current can find its way through. So a varying or undulating current is sent over the line whose vibrations exactly correspond to the vibrations caused by the speaker's voice.* The electric currents go along a coil of wire which is wound round a metal bar, or core. When electricity flows in the wire, the metal core becomes a magnet. As soon as the electricity stops flowing, the core stops being a magnet. The metal core in the tape recorder becomes a magnet with each electric current from the microphone. This happens hundreds of times per second for most seconds. Every time the core becomes a magnet, it makes some of the particles on the tape into magnets. These particles remain magnets even after the tape has passed the core. They form a pattern on the tape. Different sounds produce different patterns. As the recording continues, the tape winds off one reel, or spool, onto another. Before the tape can be played back again, it must be wound onto the first spool again.

Figure 4-10: Including a subpart's process explanation while explaining the object's function

---

help in choosing one over another or in choosing the links that will result in a coherent process explanation. There is thus a need for directives that can guide the system in retrieving facts from the knowledge base.

---

[9] This text was formed by inserting the description of a microphone into the description of a tape recorder taken from the New Encyclopedia of Science.

Description of a tape recorder[10]

The tape recorder is a machine that records, stores, and reproduces sound. A tape recorder has a microphone which picks up sounds. The microphone changes sounds into electric currents. The electric currents go along a coil of wire which is wound round a metal bar, or core. When electricity flows in the wire, the metal core becomes a magnet. As soon as the electricity stops flowing, the core stops being a magnet. The metal core in the tape recorder becomes a magnet with each electric current from the microphone. This happens hundreds of times per second for most seconds. Every time the core becomes a magnet, it makes some of the particles on the tape into magnets. These particles remain magnets even after the tape has passed the core. They form a pattern on the tape. Different sounds produce different patterns. As the recording continues, the tape winds off one reel, or spool, onto another. Before the tape can be played back again, it must be wound onto the first spool again.

The microphone converts soundwaves into current. When a person speaks into a microphone causes the soundwaves strike against a diaphragm and cause it to vibrate back and forth in just the same way the molecules of air are vibrating. The talking current passes through a carbon chamber filled with granules, so that the electricity must find its way from granule to granule inside the box. When the diaphragm moves inward under the pressure from the soundwaves the carbon grains are pushed together and the electricity finds an easier path. Thus a strong current flows through the line. When a thin portion of the soundwave comes along, the diaphragm springs back, allowing the carbon particles to be more loosely packed, and consequently less current can find its way through. So a varying or undulating current is sent over the line whose vibrations exactly correspond to the vibrations caused by the speaker's voice.

Figure 4-11: Including a subpart's process explanation after explaining the object's function

It can be argued that the same holds for the constituency schema. Frequently, predicates will not sufficiently constrain the decision of what to include in the text.

---

[10] This text was formed by inserting the description of a microphone after the description of a tape recorder taken from the New Encyclopedia of Science.

Description of a telephone from a junior encyclopedia

1) When one speaks into the transmitter of a modern telephone, these sound waves strike against an aluminium disk 2) or diaphragm 3) and cause it to vibrate back and forth 4) in just the same way the molecules of air are vibrating. 5) The center of this diaphragm is connected with the carbon button 6) originally invented by Thomas A. Edison. 7) This is a little brass box filled with granules of carbon 8) composed of especially selected and treated coal. 9) The front and back of the button are insulated. 10) The talking current is passed through this box so that the electricity must find its way from granule to granule inside the box. 11) When the diaphragm moves inward under the pressure from the sound waves the carbon grains are pushed together 12) and the electricity finds an easier path. 13) Thus a strong current flows through the line. 14) When a thin portion of the sound wave comes along, the diaphragm springs back, 15) allowing the carbon particles to be more loosely packed, and 16) consequently less current can find its way through. 17) So a varying or undulating current is sent over the line whose vibrations exactly correspond to the vibrations caused by the speaker's voice. 18) This current then flows through the line to the coils of an electromagnet in the receiver. 19) Very near to the poles of this magnet is a thin iron disc. 20) When the current becomes stronger it pulls the disc toward it. 21) As a weaker current flows through the magnet, it is not strong enough to attract the disk and it springs back. 22) Thus the diaphragm in the receiver is made to vibrate in and out....

Text Decomposition

1: Cause-effect
2: Cause-effect
3: Cause-effect
4: Cause-effect
5: Depth-Attributive
6: Elaboration
7: Depth-attributive
8: Depth-attributive
9: Depth-attributive
10: Cause-effect
11: Cause-effect
12: Cause-effect
13: Cause-effect
14: Cause-effect
15: Cause-effect
16: Cause-effect
17: Cause-effect
18: Cause-effect
19: Depth-attributive
20: Cause-effect
21: Cause-effect
22: Cause-effect

Figure 4-12: Decomposition of the telephone example from the junior encyclopedia text into rhetorical predicates

For example, the ''identification'' predicate may not totally constrain the superordinate. If an object has several superordinates, as in the case of multiple hierarchies, one would need further directives to decide on which superordinate to choose. For example, one might need to know the goal of the user in asking for a description in order to select the appropriate hierarchy. Alternatively, given a deep generalization tree, it might not always be adequate to provide the immediate superordinate to fill the identification predicate. With the ''constituency'' and ''attributive'' predicates, a problem might arise when an object has too many parts and too many attributes to include them all. Again, a priority scheme would have to decide which to include. If represented explicitly, this priority schema would consist of directives telling what to retrieve from the knowledge base after the initial choice based on the predicates is made. Nevertheless, the predicates do indicate what kind of information to include in a specific way. The constituency schema captures the organization of the text and can guide the generation process in producing a coherent text (assuming, of course, that the knowledge base contains all the relevant information). The need for further directives only arises when the underlying knowledge base is very large and there are many choices for each predicate.

The situation is worse for a schema that would only indicate a pattern of cause-effect links. Without directives on how to traverse the knowledge base, a system cannot determine which links to choose as there are different kinds of cause-effects links, and there will be typically too many such links to include in the text. In this case, the predicates hardly constrain the decision process, and directives are immediately needed. This is why it is important to define a new strategy that consists of directives; this strategy, based on the underlying knowledge base, dictates which link to choose to form a process description.

Using the rhetorical predicates as in McKeown is not the only way to decompose a text in order to determine an organizing structure. Another way to decompose a text that might give rise to a useful organizing structure is in terms of the coherence relations identified in [Hobbs 78]. (The reader is referred to Section 2.2 for a description of coherence relations.)

Figure 4-13 shows the decomposition of part of the description of the telephone from the *Britannica Junior Encyclopedia* (from Figure 4-6) in terms of coherence relations. These relations are useful to point out the relationships among the sentences in the text. However, this top-down approach is very goal oriented, as coherence relations reflect a communicative goal the speaker is trying to achieve. Many different coherence relations can be chosen at any point during the discourse construction. As a result, to construct a coherent text from these relations, one would need to specify how they can be decomposed and in which order they should appear. Without these constraints, it would be hard to employ coherence relations to organize a text coherently when there are many choices in the knowledge base. The process trace provides these constraints.

Similarly, it is possible to obtain a decomposition of the texts using the nucleus/satellite schemata defined by Mann [84a] , also described in Section 2.2. The decomposition of the telephone example from the junior encyclopedia using Mann's schemata is shown in Figure 4-14. The pattern in this decomposition is one of inform/elaboration-background and condition/conditional nuclei, as shown in the figure. The elaboration and background, in the case of the junior encyclopedia texts, correspond to process information. Like the coherence relations, these schemata do not restrict the textual order of the relations, and any schema can be expanded into any other at any point. Therefore, to use these schemata for generation, one would need a way to dictate which relation to include when in order to provide a clear process description. A control strategy must also determine which schema to expand,

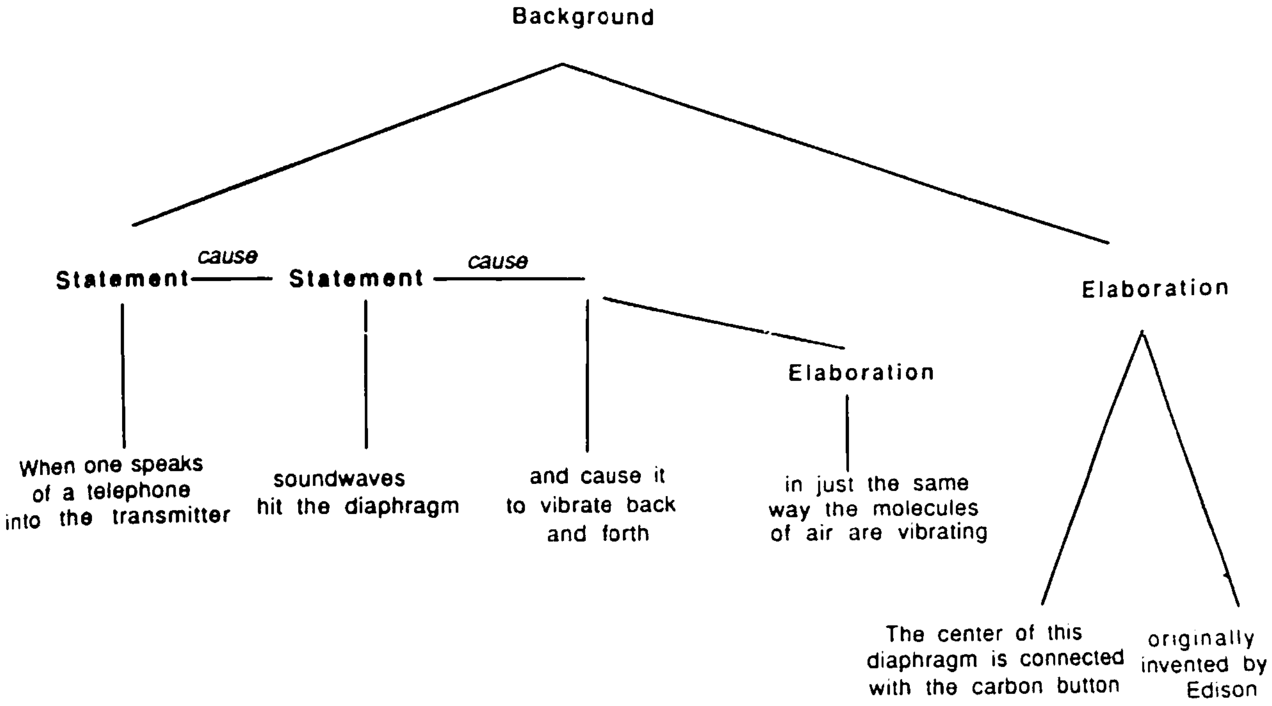

Background
Statement
cause
Statement
cause
Elaboration
Elaboration
When one speaks of a telephone into the transmitter
soundwaves hit the diaphragm
and cause it to vibrate back and forth
in just the same way the molecules of air are vibrating
The center of this diaphragm is connected with the carbon button
originally invented by Edison

Figure 4-13: Decomposition of part of the telephone example from the junior encyclopedia text into coherence relations

in which way to expand it, which satellite to choose, etc. Such a control strategy is precisely what the process strategy gives.

4.3.5 Summary of the textual analysis

There are two very different types of descriptions in naturally occurring texts that are characterized by means of distinct discourse strategies. One of these strategies, the process trace, is a new strategy that consists of directives instead of rhetorical predicates. These directives are necessary in order to sufficiently constrain the generation process. Identifying a new strategy is useful as the strategy can be added to the tools available to a generation system, thus increasing the kinds of texts that can be generated. This new strategy allows for the generation of process-oriented descriptions, as were found in the texts I examined.

As a result of the difference in organizing structures, the type of information that is included in the descriptions is also different: in the texts from the adult encyclopedias and the manual for experts, the information included is mainly structural, while in the texts from the junior encyclopedias and the manual for novices, the information included is mainly functional (or process oriented). One of the main differences between the two audiences at which the two groups of texts were aimed is their assumed level of domain knowledge. I postulate that the writers' choice of strategy might be based on the assumed domain knowledge of the expected readers.[11] If so, the reader's level of knowledge about the domain affects the *kind* of information provided as opposed to just the *amount* of information, as was previously assumed [Wallis and Shortliffe 82; Sleeman 82]. This is significant as it adds a dimension (the different kinds of information available) along which a system can tailor its

---

[11] No psychological experiments have been conducted to confirm this hypothesis. It is based on observations about what happens in natural texts.

1) When one speaks into the transmitter of a modern telephone, these sound waves strike against an aluminium disk 2) or diaphragm 3) and cause it to vibrate back and forth 4) in just the same way the molecules of air are vibrating. 5) The center of this diaphragm is connected with the carbon button 6) originally invented by Thomas A. Edison. 7) This is a little brass box filled with granules of carbon 8) composed of especially selected and treated coal. 9) The front and back of the button are insulated. 10) The talking current is passed through this box so that the electricity must find its way from granule to granule inside the box. 11) When the diaphragm moves inward under the pressure from the sound waves the carbon grains are pushed together 12) and the electricity finds an easier path. 13) Thus a strong current flows through the line. 14) When a thin portion of the sound wave comes along, the diaphragm springs back, 15) allowing the carbon particles to be more loosely packed, and 16) consequently less current can find its way through. 17) So a varying or undulating current is sent over the line whose vibrations exactly correspond to the vibrations caused by the speaker's voice. 18) This current then flows through the line to the coils of an electromagnet in the receiver. 19) Very near to the poles of this magnet is a thin iron disc. 20) When the current becomes stronger it pulls the disc toward it...

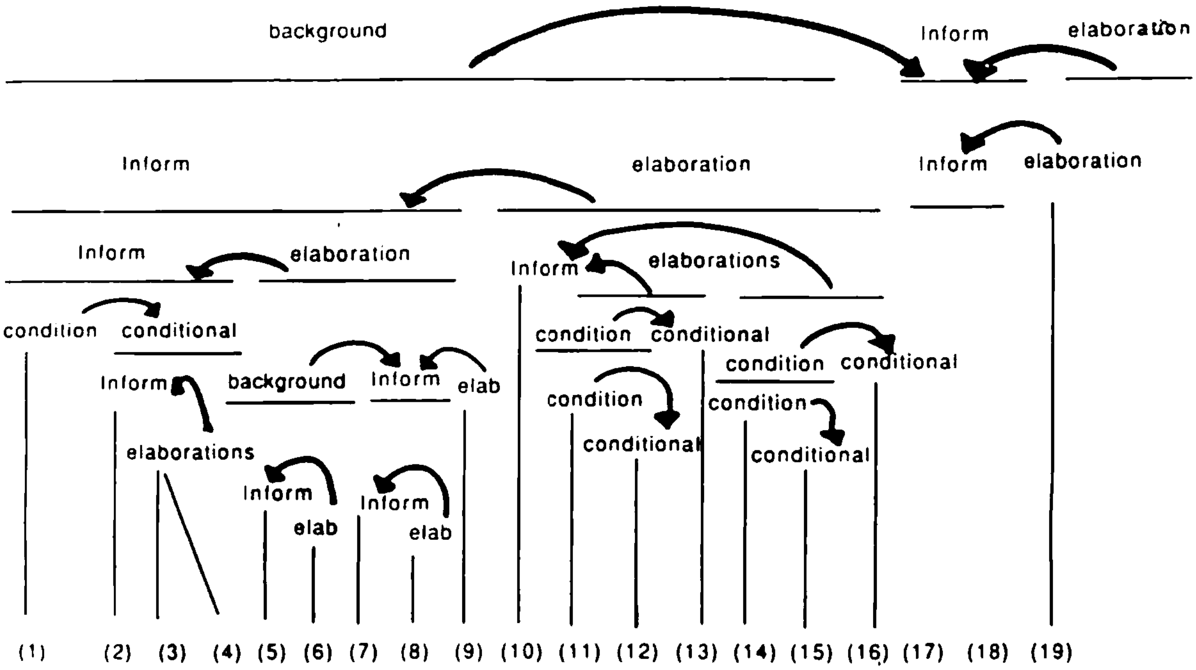


Figure 4-14: Decomposition of the telephone example from the junior encyclopedia text into nucleus/satellite schemata

answers to users having different levels of domain knowledge. I will show in Chapters 6 and 7 how a user's level of expertise can be incorporated in a generation system and how it can guide the system in choosing a discourse strategy.

4.3.6 Plausibility of this hypothesis

Recall that I assumed that the goal of a description is to allow the user to form a mental model of the functionality of the object. As seen in Section 2.3, research in psychology indicates that expert users have more knowledge not only about individual components but also about the causal models involved and the interconnections among parts [Chi *et al* 81; Lancaster and Kolodner 87]. Expert users, then, are likely to have functional knowledge about the domain and to know how parts might interact with each other. As they can use this knowledge when reading the description,[12] they should be able to pull together all the parts provided in the description in order to "understand" the description as a whole. That is, they should be able to figure out how the parts fit together to form an object capable of performing a function. Since the reader is able to construct a mental model, it is unnecessary to include process information in the description. Assuming that the reader will be able to infer the processes involved, providing such useless information would contradict the principles of cooperative behavior [Grice 75].

On the other hand, a user who does not have enough knowledge to infer the processes linking the parts would be unlikely to understand a mostly structural description of an object and to be able to construct a functional mental model of the object from such a description. For this sort of naive user, if the description is to be informative and understandable, it must describe how the parts perform the function

---

[12] Research in reading comprehension indicates that readers indeed use their previous knowledge in order to understand new texts [Schank and Abelson 77; Anderson *et al.* 77; Wilson and Anderson 86].

of the object. The description must therefore include process information. Previous research in reading comprehension strengthens my belief that a user who does not have knowledge about the functions of the various parts will not be able to make sense of a description centered around parts [Davison 84; Wilson and Anderson 86]. Criticizing readability formulas, Davison argues that these formulas often fail to correctly assess a text's difficulty because they do not measure the background knowledge a text requires in order to be understood; lack of background knowledge is often what makes a text hard to comprehend. Wilson and Anderson also point out that readers can fail to understand a text mainly because the text implicitly assumes knowledge that they do not have. A description that explicitly provides causal information might thus be appropriate for a naive user.

To summarize, I suggest that the user's domain knowledge should affect the content of a description with respect to the kind of information to include in a text, and not just the level of detail. I propose using the process trace when the user is relatively naive about the domain of discourse, and the constituency schema when the user has expertise about the domain. Extensive testing and thorough psychological experiments would be required to decide whether this is a good strategy choice. In any case, tailoring by changing the kind of information provided gives more flexibility in tailoring texts than was previously allowed for.

4.4 Combining the two strategies to describe objects to users with intermediate levels of domain knowledge

The two strategies presented account for the main differences found between the adult and junior encyclopedia entries and can be used to describe objects to naive or expert users. Users are not necessarily either strictly naive or expert in a domain however. Users may have "local expertise" in that they may know about a few objects in the domain but not others [Paris 84]. Such users would not be considered naive, but, as there are many objects they do not know in the domain, they would not be considered experts either. A combination of the two strategies presented for naive and expert users seems appropriate to describe objects to users with intermediate levels of expertise in order to provide them with the proper mix of structural and functional information.

As an example, suppose a description of a telephone is provided to a user who knows how a microphone works, but not how the telephone functions. In describing the telephone to this user it is necessary to first describe how the parts of the telephone work together, using the process trace strategy. It is not necessary, however, to describe the microphone using the process trace. Attributive information (or a call to the constituency schema) should be enough to describe the microphone. For the other part the receiver, the process strategy is still appropriate in order to explain its mechanism.

Descriptions combining the two strategies occur in naturally occurring texts. Consider for example the text shown in Figure 4-15, taken from [Chemical 78]. The description starts with the constituency schema but ends with a process trace, which is underlined in the figure: the ''ir spectrometer'' is first described in terms of its parts; each part is then described in turn (depth-attributive); finally, the authors switch to a process trace to describe the ''thermocouple detector,'' probably assuming it is unknown to the reader. To fully understand this text, the reader must already know (or be able to infer information) about the ir spectrometer's purpose, the ''ir radiation'' and the ''monochromator.'' I show how the strategies can be mixed in more detail in Chapter 6.

---

(1) The ir spectrometer consists of three essential features: a source of ir radiation, a monochromator and a detector. (2) The primary sources of ir radiation are the Globar and Nernst glower. (3) The Globar is a silicon carbide rod heated to 1200 degrees C. (4) The Nerst glower is a rod containing a mixture of yttrium, zirconium, and erbium oxides that is heated electrically to 15000 degrees C. (5) Earlier ir spectrometers contained prism monochromators but today gratings are used almost exclusively. (6) Most detectors in modern spectrometers operate on the thermocouple principle. (7) Two dissimilar metal wires are connected to form a junction. (8) Incident radiation causes a temperature rise at the junction and the difference in the temperature between head and tail causes a flow of current in the wires which is proportional to the intensity of the radiation.

Text decomposition

1. Constituency
2. Depth identification for the ir radiation
3. Depth identification for the ir radiation
4. Depth identification for the ir radiation
5. Depth identification for the monochromators
5. Depth identification for the monochromators
7. Depth identification for the thermocouple spectrometer
8. Process trace for the thermocouple principle.

Figure 4-15: Text from the *Encyclopedia of Chemical Technology*

4.5 Summary

I analyzed texts from adult and junior encyclopedias, high school text books and manuals for experts and novices, in order to compare their organizing strategies. In doing so, I found that the entries from the adult encyclopedias and the manual for experts could be characterized by the *constituency schema*, a discourse strategy developed in [McKeown 85]. These descriptions are organized around subparts and contain mainly details about subparts and their properties. On the other hand, the texts from junior encyclopedias, the high school text books and the manual for novices did not fit any known organizing strategy. I found that these descriptions tended to be organized around the process information associated with the objects being described, so that these descriptions contain more details about functional information. I called this strategy the *process trace*. This is a new type of strategy in that it consists of directives on how to trace the knowledge base rather than of rhetorical predicates. The differences in organizing strategies in these two categories of descriptions are important as they add a new dimension along which tailoring a description to the user's level of expertise can be done.

5. The Discourse strategies used in TAILOR

5.1 Introduction

In the previous chapter, I have shown that two distinct discourse strategies can be used to generate descriptions of complex devices: the constituency schema and the process trace. A description using the constituency schema is organized around the properties and parts of the object being described. The process trace revolves around the processes allowing the object to perform its function. The two types of descriptions can be used to generate descriptions for different users, depending on their assumed level of knowledge about the domain. In this chapter, each strategy is presented in detail.

5.2 Constituency Schema

The constituency strategy was described by McKeown [85]. It is a declarative strategy as it represents an abstract characterization of patterns occurring in many texts. As the strategy has been presented at length in [McKeown 85], it is discussed only briefly, and the change that was made to adapt it to TAILOR is described.

The constituency schema, as identified by McKeown, is represented formally in terms of its rhetorical predicates in Figure 5-1 below, using McKeown's notation: ''{}'' indicates optionality,''/'' indicates alternatives, ''+'' indicates that the item may appear 1-n times, ''*'' indicates that the item may appear 0-n times, and '';'' is used to indicate that it was not possible to unambiguously classify the propositions appearing in the text as one predicate.

Following this strategy, an object is first described in terms of its subparts or subtypes. Then, the focus of the description can either stay on the object itself by providing attributive information or switch to the subparts (or subtypes), using the

---

Constituency Schema

Constituency (description of subparts or sub-types.)
Attributive* (associating properties with an entity) / Cause-effect*/
{ Depth-identification / Depth-attributive
 { Particular Illustration / Evidence }
 { Comparison ; Analogy } }+
{ Attributive / Explanation / Amplification / Analogy }

Figure 5-1: The Constituency Schema as defined by [McKeown 85]

---

depth-attributive or depth-identification predicate. Finally, more attributive information might be presented about the original object being described.

TAILOR uses a slightly altered version of McKeown's schema. It includes the identification predicate at the beginning of the schema, as, in the texts analyzed, objects were very often first identified in terms of a superordinate. This predicate is optional, however. This was the only change that needed to be performed, as the texts examined matched the rest of the constituency schema as defined. The resulting schema, together with an example, is shown in Figure 5-2.

5.3 Process Trace

The process trace dictates how to traverse the knowledge base in order to describe the mechanisms that allow a device to perform its function. A description following a process trace thus explains a device's mechanism. It would be desirable to introduce a device before explaining its mechanism, however. As a result, the first step of the

{Identification (description of an object in terms of its superordinate)}
Constituency (description of subparts or sub-types.)
Attributive* (associating properties with an entity) / Cause-effect*/
{ Depth-identification / Depth-attributive
{ Particular Illustration / Evidence}
{ Comparison ; Analogy} }$^+$
{ Attributive / Explanation / Amplification / Analogy }

Text	Predicates
Abacus [Collier's 62]:	
1) An abacus is a manually operated storage device that aids a human operator.	Identification
2) It consists of a row of any number of parallel wires, rods and grooves	Constituency
3) on or in which slide small beads or blocks.	
4) The strung beads are divided into two sections by means of a bar perpendicular to the rods.	3-7: Depth-attributive
5) One section has one or two beads, representing 0 and 5, depending on their position along the rod.	
6) The second section has four or five beads, representing units.	
7) Each bar represents a significant digit, with the least significant digit on the right.	

Figure 5-2: The modified Constituency Schema

process trace will also be to identify a device in terms of its superordinate, together with a summary of its function if possible. This step is similar to the first step of the constituency schema. After this prefatory sentence, the device's mechanism is explained. This is the main part of the process trace, and it is the part that is described in detail in this chapter.

The process trace can be seen as following a sequence of events, called the *main path* going from a *start* state to a *goal* state. In a device description, this chain of events describes how the object performs its function, from the beginning (start state) to the end (goal state), thus providing an explanation of that process.

To use the process trace, the main path of links from the start to the goal must be identified. Given the main path, the strategy then decides what to include in the description as the program traces along the path. A link that is not on the main path but that is attached to it (that is, a link relating an event on the main path to an event not on the main path) is termed a *side link*. When an event can be decomposed into another series of events, these events are called *substeps*. The process trace makes decisions about which link to include based on the type of side links or substeps it encounters. These decisions are explained later in this chapter.

Although the process trace will be described in terms of TAILOR's domain, the domain of complex devices, it should be possible to use it in any domain where processes occur and must be explained. The following section describes the assumptions made of a knowledge base in designing the process trace.

5.4 Requirements of the knowledge base

The process trace is dependent on the knowledge base only to the extent that it needs information about processes. A knowledge base with information about processes presumably describes the *events*, or actions, that occur. In TAILOR, these are the events that occur when an object performs its function. The knowledge base also most probably contains information about relationships among events, such as causal or temporal relationships. A process description is represented by relationships among various events in the knowledge base. To use the process trace on an arbitrary knowledge base, one needs to be able to identify these relationships. There are three constraints:

- It must be possible to retrieve information about the events contained in the knowledge base and to identify the relationships among various events.
- If the knowledge base contains many such relationships, the relation types need to be ranked to enable the program to decide which relations to include in the text. As the different relation types are known at the time the knowledge base is constructed, the ranking can be specified at that time. This ranking must be done before using the process trace.
- The beginning and end of a process explanation have to be identifiable. This constraint will become clearer when the process trace is described.

To summarize, if information about events, their relationships and the start and goal states is available, the process trace can be used to produce a process explanation.

Given knowledge bases other than TAILOR's with different types of links and different representations, the implementation of the strategy (the functions which retrieve the information from the knowledge) would have to be adjusted to reflect the changes, but the process trace would remain unchanged.

5.5 The process trace: a procedural strategy

Unlike other existing strategies (such as the constituency schema), this new strategy does not impose a structure on the underlying knowledge base, but follows it closely. I termed this type of strategy a *procedural* strategy. In this sense, a process trace is similar to the strategy identified in a study done by Linde and Labov [75], wherein they asked people to describe their apartments. They discovered that people generally take a visual tour of their apartment and describe it as they go along. Linde and Labov formalized this result in a discourse strategy, which I refer to as the *apartment strategy*. The apartment strategy follows the lay-cut of the apartment being described, tracing the imaginary tour the speaker is taking through the apartment. The speaker may either start by describing a main hall and then describe the rooms off that hall, or may go from one room to the next if they lead to one another. Unlike the process trace, the apartment strategy was never formalized as part of a generation system to produce text. The apartment strategy provides a physical description and not a functional one. Furthermore, the apartment strategy results in a *complete* description of the apartments (with branching points and backtracking), in that every room is generally included, although not all with the same level of detail.[13] The apartment strategy did not account for describing apartments that are too complex to be described in that way. In the process trace, however, not every causal link will be included. In a complex knowledge base, one can expect to have many links branching out of the main path. To keep the process explanation coherent, a process trace cannot follow every causal link in the knowledge base that may branch out of the main path. Therefore the strategy needs to dictate which side links to include in the text, which to ignore, and when to mention a link it has chosen to include.

---

[13] Closets were not necessarily included.

The next sections describe the different types of links that may exist, how a program can identify the main path, and various choices the process trace needs to make. The process trace was identified from the texts studied. Decisions about which link types to include and when to include them were made with the assumption that to remain coherent, the process explanation needs to focus on the main path; that is, on the main sequence of events that take place in order for the device to perform its function. The process description found in the texts that were studied focused on this main sequence of events. As these texts presented the main sequence of events in sequential order, that is from start to finish, the process trace will duplicate this behavior. Note that it would not be hard to change the algorithm to produce a process explanation in which the events are presented in another order. For example, it might also be reasonable to traverse the main path in reverse, from goal to start. A process explanation produced that way is probably less clear than one in which the events are presented in sequential order from start to goal, however. This is illustrated by the two functional descriptions of how a door lock works, shown in Figures 5-3 and 5-4. Both descriptions follow the main path. In Figure 5-3 the main path is given from the start to the goal, while it is given beginning with the goal state in Figure 5-4. I believe that the explanation in the first figure is clearer, as it provides events in the order in which they occur.

The texts studied did not often include other links unless they were enabling conditions for events on the main path. I took these observations into consideration when formalizing the process trace. The various decisions that are presented in the next sections were thus based primarily on the texts examined but also on intuition about what makes a text coherent and experimentation with TAILOR.

Door Locks [The Way Things Work 72]

When the key is turned, it first presses the tumbler - which is kept engaged with the bolt by the pressure of a spring - upward and thus releases the bolt. The key bit (the lateral projection at or near the end of the key) then engages with the first notch on the underside of the bolt. Further rotation of the key causes the bolt to slide until the catch of the tumbler engages with the next notch on the top of the bolt.

Figure 5-3: The process explanation follows the main path from the start state to the goal state

Door Locks

The catch of the tumbler engages with a notch on the top of the bolt. This occurs when the bolt slides, which is caused by further rotation of the key. The first notch of the underside of the bolt is engaged by the key bit, after the tumbler is released. This is enabled by the tumbler being pressed upward when the key is turned.

Figure 5-4: The process explanation follows the main path from the goal state to the start state

5.5.1 Identifying the main path and different kinds of links

Process information consists of links between events in the knowledge base that express different relations between events. The knowledge base created by RESEARCHER contains three types of relations among events, and therefore three types of links. These links seem typical of a knowledge base. They are:

- *Control* links: these links indicate causal relations between events, such as *cause-effect*, *enablement*, (which are termed here *positive* control links), as in "X causes Y" or "X enables Y", or *interruptions*, such as "X interrupts Y," (which is called a *negative* control link).
- *Temporal* links: these links indicate temporal relations between events, such as "X happens *at the same time* as Y."
- *Analogical* links: these indicate analogies between events, such as "X is *equivalent to* Y," or "X *corresponds to* Y."

Because these links represent relations between events, they could all be potentially mentioned in the process description. As they are potentially too numerous, there is a need to choose those that will produce a coherent process explanation. Because they express different kinds of relations, it is possible to rank them in order of importance. This ranking must be done in order for the strategy to decide which links to include in producing a process explanation, when it has to choose among several.

While the ranking does not need to be a strict ranking, if all types are equally important and can all be included in the text, the program will have to simply pick one arbitrarily. A strict ranking thus helps the decision process as it allows a program to choose the "most important" link first.

Based on texts, the ranking was determined to be as follows in the domain of devices:

1. Control links: as these links represent causality between events, they are the most important links for describing a process. Among those, the causal links are the most significant, as they represent the strongest positive control relationships among events: to describe the main sequence of events that take place when an object performs its function, a causal relationship is better than a mere enablement relation. In the texts studied, causal links were indeed emphasized.
2. Temporal links: when no control relation among events can be found, the temporal relations among events become the next most important links for producing an explanation of a chain of events in sequential order.

3. Analogical links: when no control or temporal relation exist among events, analogical relationships indicating how events compare with each other become important.

5.5.2 The main path

The *main path* is the main chain of events that occur from the start to the goal. In TAILOR's knowledge base, the main path describes the sequence of events that are performed when an object achieves its function. This sequence is assumed to be linear. If another series of events happens simultaneously, it will be considered as a side chain.

Consider, for example, the subset of TAILOR's knowledge base representing the *loudspeaker* shown in figure 5-5.[14] To simplify the figure, not all the relations have been included. In this figure and the other figures in this chapter, each labeled box represents a frame containing information about an object. For clarity, only the labeled boxes are shown here, instead of all the information contained in a frame. The boxes are connected with non-directed arcs that represent a parts hierarchy. So, for example, the ''loudspeaker'' is shown to have the following parts: an ''armature,'' a ''magnet,'' an ''air gap'' and a ''diaphragm.''

In the figures, *events* and *structural relations* among the objects are represented by directed arcs between objects. The ''current'' ''varies'' is an example of an event. The ''coil'' being ''wound around'' the ''magnet'' is an example of a structural relation between two objects.

Links between events are shown with arcs between the directed arcs (arcs between

---

[14] Because this work is being conducted at the same time as the research for RESEARCHER's parser, the knowledge base was built by hand. It is, however, faithful to the representation built by RESEARCHER. The knowledge base used in TAILOR is represented using a frame based knowledge representation, where items and relations are frames with standard slots. The representation details will be shown in section 7.4.

events). The thick directed arcs between events represent the main path, that is the main sequence of events that occur when the object (the loudspeaker) performs its function, namely, in the figure:

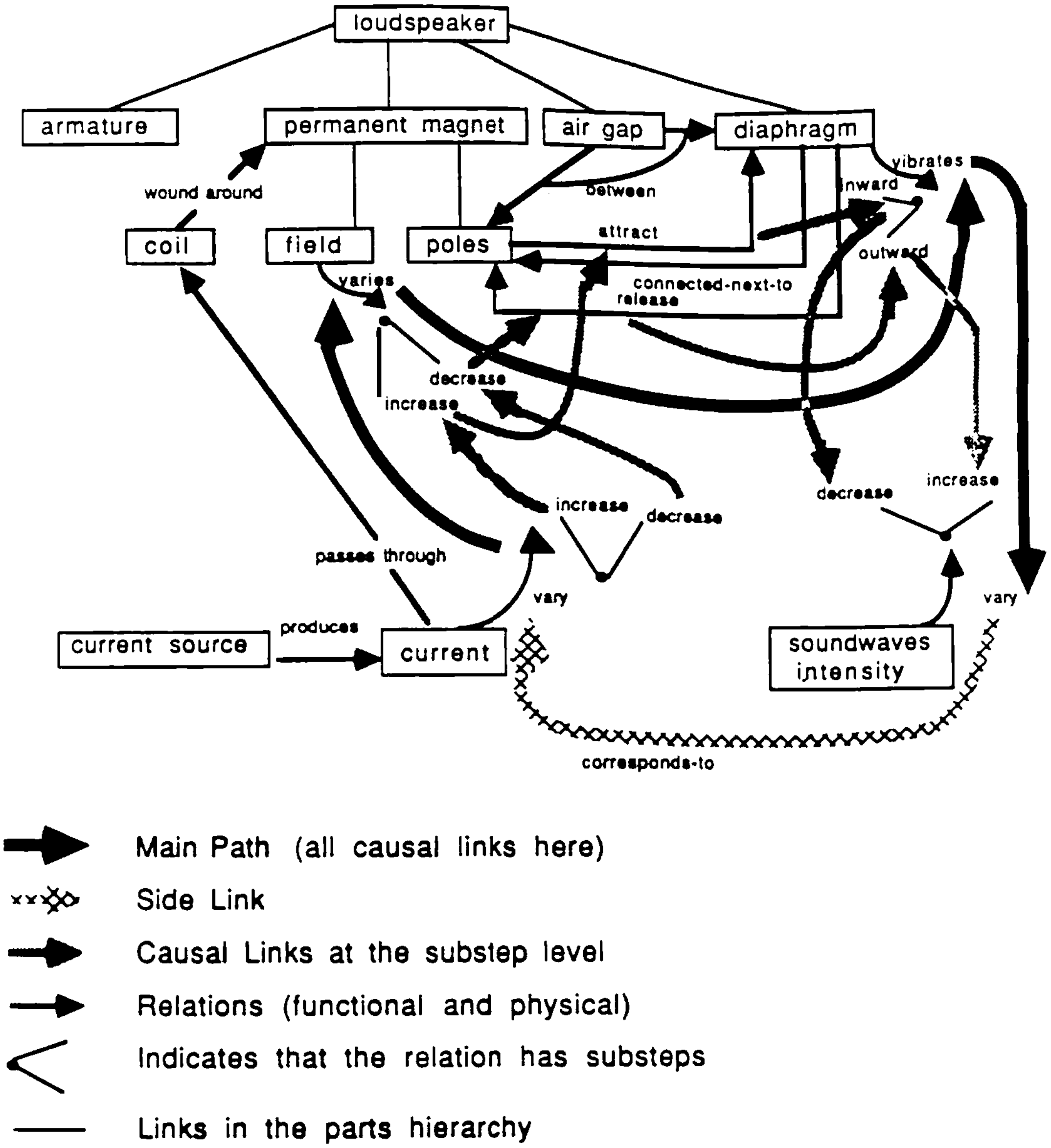
- The varying current causes the field of the electromagnet to vary;
- The varying field causes the diaphragm to vibrate;
- The diaphragm vibrating causes soundwave intensity to vary.

In this case, all the links on the main path are *control* links. There is also a side link, the *analogical* link ''corresponds to.''

Some of the events in the figure can be broken up into substeps (e.g., ''varies'' can be decomposed into ''increases'' and ''decreases''). These substeps are also shown in the figure. The causal links at the substep level are represented with lighter directed arcs between events. They represent the sequence of events that occur when an event is decomposed into its substeps.

Since the program needs to identify the main path before starting generation, TAILOR includes an algorithm for obtaining it. This algorithm is essentially a depth first search with ordered backtracking. It can be summarized as follows:

- Fetch the start and goal states. In TAILOR, these are obtained from the object frame. In RESEARCHER's representation, the function of a device is seen as an action performed on an input to produce an output. The function (or purpose) of a device is represented by a relation between two entities or possibly two other events [Baker and Danyluk 86]. For example, the function of the loudspeaker is to ''change current into soundwaves.'' This is represented by the relation corresponding to ''change,'' relating two entities: ''current,'' the input, and ''soundwaves,'' the output. The relation corresponding to the device's function and the input and output objects can be retrieved from a device's frame. The corresponding goal and start states can be obtained from examining the events these objects are involved in.
- Search starting from the goal state:
  - Take all the links (relations) in which the goal event participates. If the link is a causal or temporal one, consider the links in which



Main Path: 1- The varying current CAUSES the field of the electromagnet to vary
2- Varying field CAUSES diaphragm to vibrate
3- The diaphragm vibrating CAUSES soundwave intensity to vary

Side Link: Soundwaves intensity varies CORRESPONDS-TO current varies

Figure 5-5: Main path for the loudspeaker

the goal event is the ''caused'' state or the state which happens last.

- Take the most important of these links (given the ranking introduced in Section 5.5.1). If several links have equal ranking, one is chosen at random. Given the chosen link, the search is repeated. The goal state is now the second event involving the chosen link. When the original start state is reached, the process stops. If the most important link does not lead to the start state, the program backtracks and starts over with the next most important link.

While searching to obtain the main path, the program also keeps track of all the other links it encounters (i.e., it tags them). After the main path has been identified, the program traverses each marked link to check both whether it leads to a side chain and whether its side chain is reattached to the main path. This will be important in deciding whether to include the side chain or not.

As mentioned in Section 5.4, the start and goal states of a process explanation have to be identifiable to use the process trace on an arbitrary knowledge base. Without a goal state, the program would be unable to search for the main path, as there would be no beginning point for the search. Without the start state, the search for the main path would end only if there are no further relationships between the event reached and other events. A threshold could be added to ensure a halt to the search, but, at this point, no mechanism is provided to infer when the best time to stop would be. While this is the main limitation of the process trace, it is not unreasonable to assume that there would be a beginning and an end to a process explanation, whether given or inferred.

5.5.3 Deciding among several links

In general, to produce a coherent text, one should avoid continual side-tracking on different subjects. For the process trace, this means staying roughly on the main path, without continuously following side links. In other words, the focus of the process description should be on the main path. I have determined from texts, however, that including a side link is sometimes useful or necessary as in the following situations:

- The side link introduces an analogy that provides a clearer process explanation. An example is shown in Figure 5-6. This was done in the junior encyclopedias texts presented earlier.
- The side link introduces an important side effect that the user should know about. This can happen in a medical domain, for example, when the side effect of a drug that is prescribed is of vital importance.

Side links can have various structures with respect to the main path. Based on their structure and type, I classified the different overall structure of side links that were encountered in the knowledge base into five cases, each of which requires different treatment:

- there is a long side chain that is related to the main path
- there is an isolated side link
- there are many short links
- there is a long side chain that is not related to the main path
- there are substeps.

The cases and their associated algorithms are presented below.

5.5.3.1 The side chain is long but related (attached) to the main path

In this case, the side link is a long chain of events that is attached to the main path. This case is illustrated in Figure 5-7 and can be characterized by the diagrams (1-a) and (1-b) below:

Description of the loudspeaker

Consider the subset of the knowledge base given in Figure 5-5. The following description simply traces through the main path, that is the main sequence of events that take place when the loudspeaker changes current into soundwaves.

```
The variation of the  current causes the field  of
the magnet to vary.  This causes the diaphragm  to
vibrate.  The  vibration of  the diaphragm  causes
the intensity of the soundwaves to vary.
```

This second description is still a description of the loudspeaker. This time, however, the side link *corresponds to* is included. This side link was included in the texts from the junior encyclopedia. Including this side link produces a clearer explanation of the process, as it provides a link (''corresponds-to'') between the input (''the current'') and the output (''the soundwave intensity'').[15] The side link is shown underlined in the text:

The variation of the current causes the field of the magnet to vary. This causes the diaphragm of the loudspeaker to vibrate. The vibration of the diaphragm causes the intensity of the soundwaves to vary. <u>The intensity varies, like the current varies</u>.

Figure 5-6: An analogical side link can produce a clearer explanation

---

[15] This text is generated by TAILOR, when a short description is asked for (i.e., no substeps are given).

Diagram 1-a

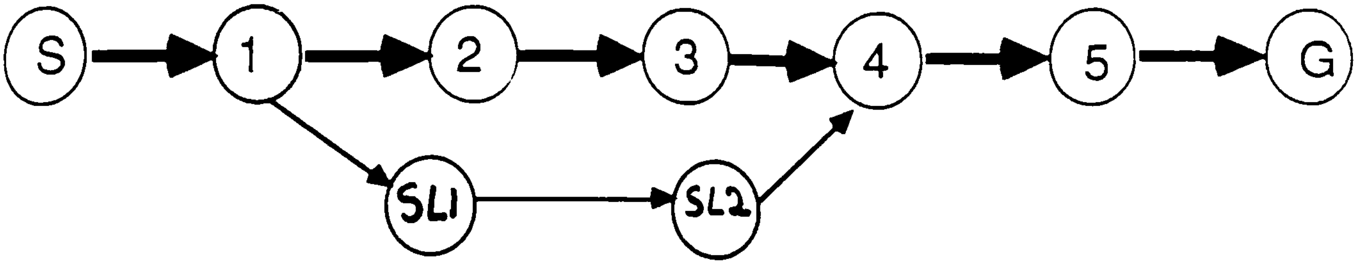
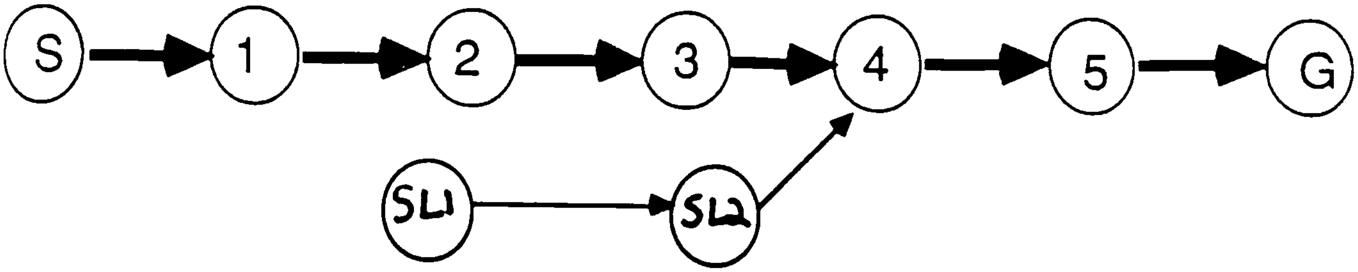


Diagram 1-b

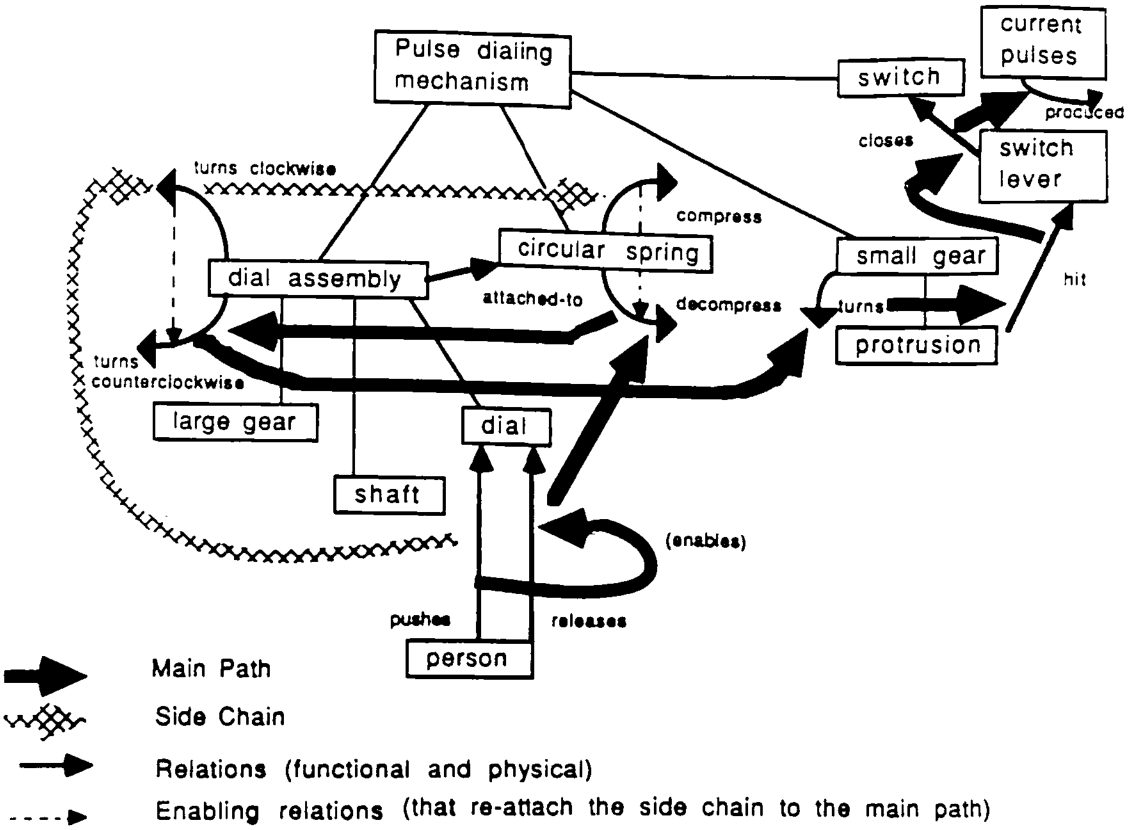


In both diagrams, the main path is represented by the thick line ("➡") while the side chain is drawn as a thin line ("→"). The nodes represent the specific events along the chain. In the diagrams, the main path from S (the start) to G (the goal) is composed of the events (1) through (5). In (1-a), there is a side chain coming off the main path at (1). This side chain gets re-attached to the main path at event (4). In (1-b), the side chain does not start from an event on the main path, but gets attached to the main path at (4).

Recall that the main path is obtained by searching from the goal state to the start state and choosing the most important links available at each point in the knowledge base. During this search, the sequence of events (1) - (2) - (3) - (4) was found to be the main path. The link (SL2) was tagged during the search, and the sequence (SL1) - (SL2) - (4) was determined to be a side chain.

A significant case of this structure is when the side link leads to an event which is an *enabling* condition to an event on the main path, as if (SL2) was an enabling condition for event (4). This is illustrated in Figure 5-7. In this figure, the main path is represented with thick directed lines between events. The thick broken line represents the side chain. Finally, the dotted lines (.......) show the *enabling* relations that re-attach the side chain to the main path. For example, a precondition for the ''spring decompressing'' is the ''spring compressing.'' So, although this event does not belong on the main path, it is an important link to mention as it is an enabling condition for an event on the main path.

If a side chain results in an event that enables an event on the main path, that side chain is mentioned in the description. The process trace specifies tracing through the side chain first, to provide the enabling condition before actually giving the causal link. The side chain is traced using a focus shift from the main path to the side chain. After the chain is traversed, the program reverts to tracing the main path, returning focus to the main path. The description of the dialing mechanism shown in Figure 5-7 is given in Figure 5-8 below.



Main Path:
[finger pushes dial] ENABLES [finger releases dial]
CAUSES [circular spring decompresses]
CAUSES [dial turns counter clockwise]
CAUSES [small gear turns]
CAUSES [protrusion hits switch-lever]
CAUSES [switch closes]
CAUSES [current pulse produced]

Side chain coming off *[finger pushes dial]*:
[finger pushes dial] CAUSES [dial turns clockwise]
CAUSES [circular spring compresses]

The side chain is related to the main path, as the event *the circular spring compresses* is an *enabling* condition for the event *circular spring decompresses* in the main path.

Figure 5-7: The side chain is long but related to the main path

Description of the pulse dialing mechanism

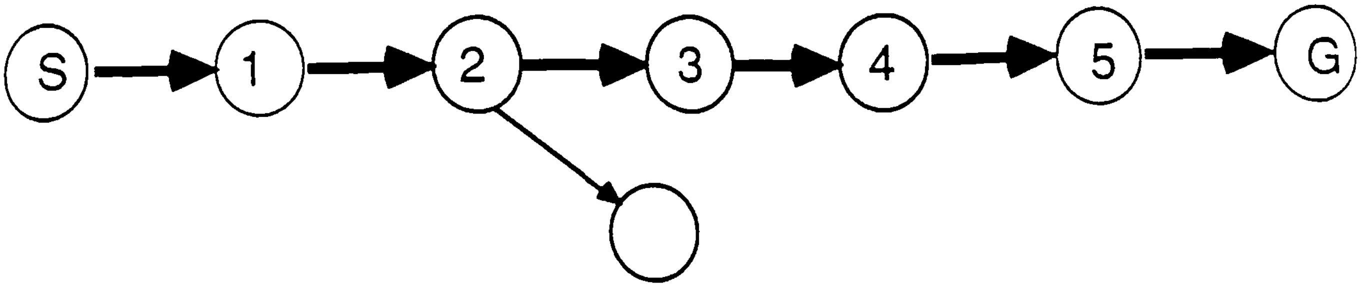
Because the person dials, the dial assembly of the pulse dialer turns clockwise.	Long-side chain
The dial assembly turning clockwise causes the spring to be compressed. The compression of the spring enables the spring to be decompressed.	Point at which it gets reattached
The person dialing enables the person to release the dial. This causes the spring to be decompressed.	Main path
The decompression of the spring causes the dial assembly to turn counterclockwise. This causes the gear of the pulse dialer to turn. The dial assembly turns counterclockwise proportionally to the way the gear turns.	Analogical side link
Because the gear turns, the protrusion of the gear hits the lever of the switch. This causes the lever to close the switch. Because the lever closes the switch, current pulses are produced.	Main path

Figure 5-8: Including a long side chain that gets re-attached to the main path

5.5.3.2 There is an isolated side link

There are two possibilities in this case. Each will be examined in turn. In the first case, the diagram is as follows:

Diagram 2-a:



Given such an isolated side link in the knowledge base, the decision whether to include it is based on the link type. In the link is an important link to mention as part of a description, it is included in the process trace.

In TAILOR's domain, the only links which fit this description are analogical links, as they provide a clearer explanation of the process, as previously observed. This case was illustrated in Figure 5-6. Indeed, a side link that represents a causal relationship that is not useful to the goal does not help understand a process explanation, as illustrated in Figure 5-9, where, the explanation of how a car moves when the ignition switch is turned is given. The causal side links are indicated in italics in the figure. Including these causal side links hinders the readers' ability to understand the explanation as additional irrelevant information is included.

On the other hand, a side link that represents an analogical link might allow readers to associate the current process explanation with an already known one and thus help them understand the process explanation. The side link that is judged important to mention is included with a temporary focus shift. If the surface generator is capable of producing complex sentences, the side link can be included in a relative or subordinate clause to avoid a focus shift.

In other domains, links indicating side effects might be important and would be included. If the link is not an important link to mention, the program continues to follow the main path.

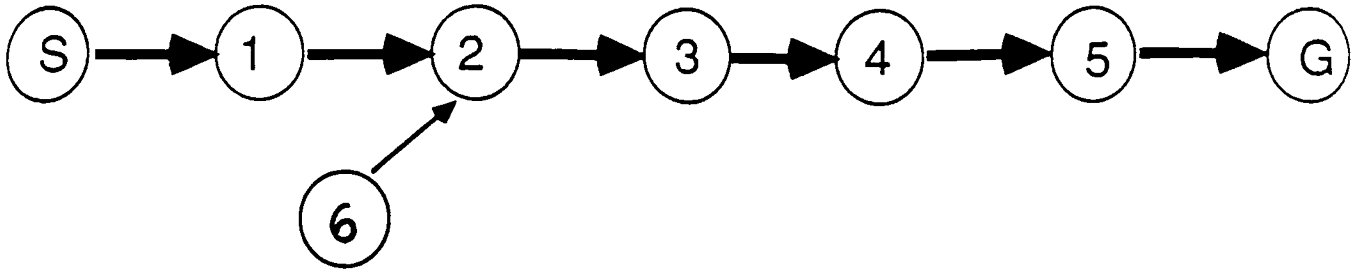
Explaining why the car moves when the ignition switch is turned on

When the key is turned into the ignition switch, current flows through the starter relay coil and causes a magnetic field. This causes the starter relay contacts to close, and current to flow to the starter motor. The starter motor turns. This causes the crankshaft to turn and the engine turns over. *This causes the generator to turn, causing electricity to be produced, enabling the headlights to turn on.* The engine turning over causes the car to move.

Figure 5-9: Including a causal side link does not render the explanation clearer

In the second case of this link structure, the diagram is as follows:

Diagram 2-b



If event (6) is the cause or an enabling condition for (1), it needs to be mentioned. The strategy dictates to mention it immediately after mentioning event (1), as in the previous case. An example of this case is presented in Figure 5-10, where the side link is shown in italics.

Description of an amplifier

An amplifier is a device that controls a strong current with a weak current. The variation of the weak current causes the voltage across the cathode and the grid to vary. This causes the strong current to vary. *The strong current was produced by a battery.*

Figure 5-10: Including a short enabling condition

5.5.3.3 There are many short links

This is similar to the case presented in 5.5.3.2, since the side link does not lead to a side chain. However, instead of having only one isolated short side link along the main path, there are now many short links. This case can represented with the following diagrams:

Diagram 3-a

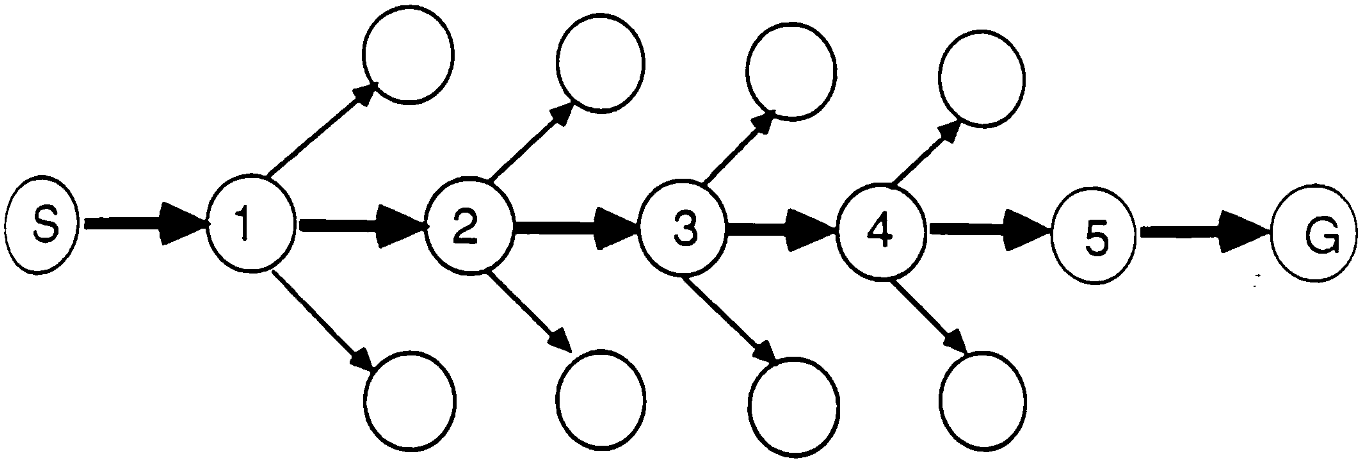
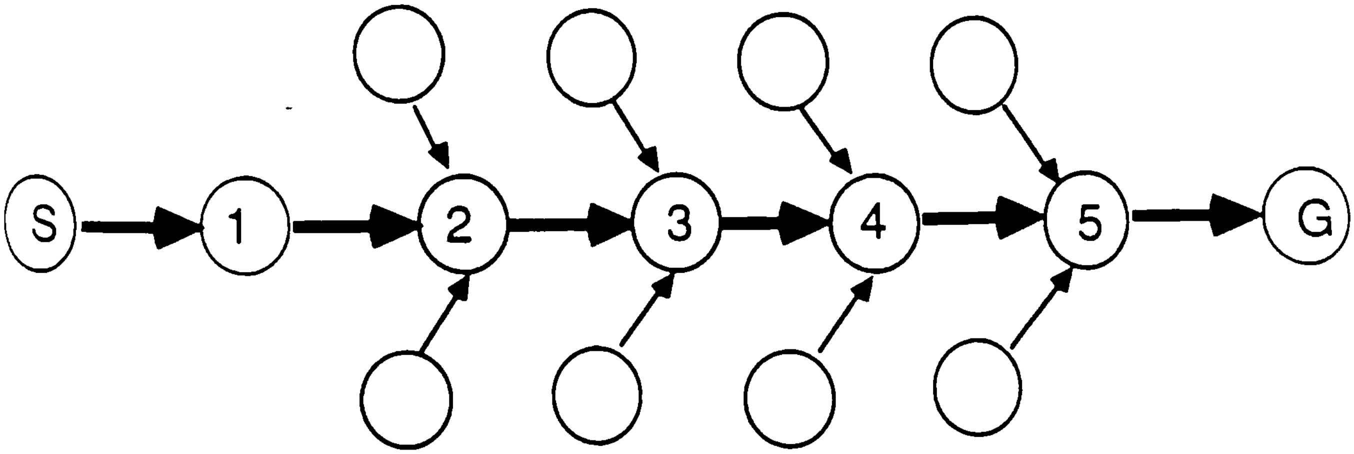


Diagram 3-b

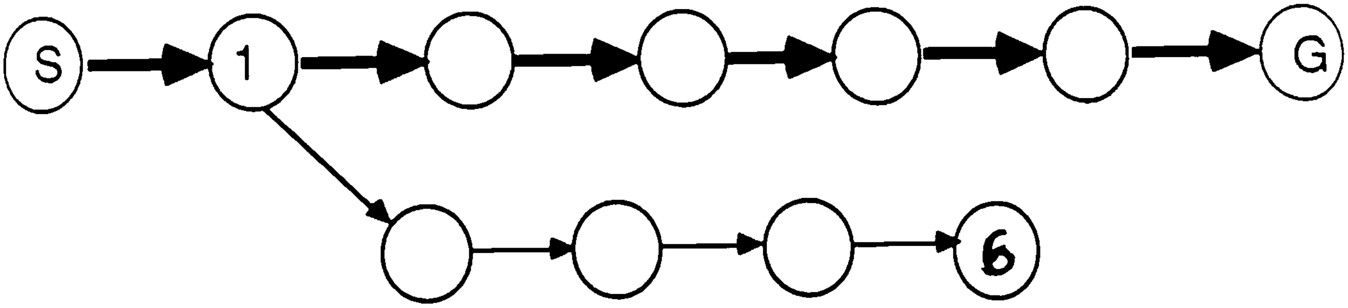


In each case (diagram 3-a, 3-b, or a combination of both), all these short side links cannot be included using temporary focus shifts as in 5.5.3.2, as doing so would result in a text with constantly shifting focus. Using complex sentences to avoid focus shifts would likewise produce a text in which the main path would be lost among the side links, resulting in a less clear description. As a result, if the links are judged to be important to mention, they are grouped together after the main path is described. In TAILOR, a parameter defines the number of short side links required to trigger this case.

5.5.3.4 There is a long side chain which is not related to the main path

Here, there is a long side chain that leads to an event unrelated to the main path. This case can be characterized by the diagram:

Diagram 4



There is a side link coming off the main path at event (1). This link leads to a chain that results in event (6), an event unrelated to the main path. In general, this side link is ignored. An example of this case is shown in Figure 5-11, where an explanation of why the light in a car turns on when the door opens is desired. The start state in this explanation is the event *door opens*, while the goal state is the event *light goes on.* To explain this process, there is no need to also explain why the buzzer goes on when the door opens. This is a side chain. It is not included in the text. In some rare cases, the side link may be essential. As mentioned earlier, this might happen in a medical domain, where side effects of drugs might be important. A side link might also be essential when it indicates a hazard, in which case it should be included as a warning. Such a side chain could then be included after tracing through the main path, using a focus shift.[16] Another example of such a case is when there are multiple goals that need to be described. Note that event (1) has to be re-introduced before following the side chain, if the side chain is described.

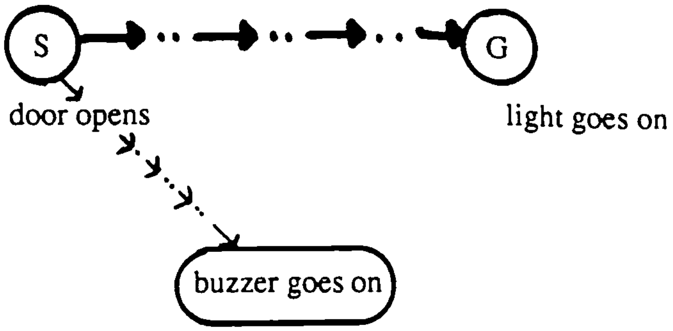
5.5.3.5 Substeps

A single event might be decomposed into substeps. For example, recall the representation shown in Figure 5-5. The event *the diaphragm vibrates* consists of (1) *the diaphragm moving forward* and (2) *the diaphragm moving backward.* The causal link between the event *diaphragm vibrating* and *soundwave intensity varying* can be decomposed into two chains of events: one from *the diaphragm moving forward* and the other from *the diaphragm moving backward.* In this case, substeps are traversed if they are not too lengthy or numerous. (This is controlled by a parameter, which, in my implementation, is set arbitrarily to allow for two substeps, each having at most four links.) Including longer substep chains is undesirable for two reasons: (1) the generated text becomes too long, and (2) in the domain of complex physical devices, the process may be described at too fine a level of detail. Tracing substeps is done by shifting focus until returning to the main path.

---

[16] In our domain, this never happens, since it is not necessary, when explaining the function of a device, to explain all the events that also happen to take place but do not take an active part in the process sequence.

Suppose I want to describe why the light inside a car turns on when the door opens. The *start* state is the event corresponding to the *opening of the door*. The *goal* state is the *light turning on.*



A side effect of *opening the door* results in the *buzzer turning on*, after a sequence of other events.
When explaining why the light turns on, shifting focus to describe why the buzzer turns on is unnecessary and confusing. If this side chain was judged important, it would be possible to include it by mention it at the end after proper re-introduction.

Figure 5-11: The side chain is long and not related to the main path

Substeps can also arise when a complex object is made of several other complex parts. Tracing through the main path of an object corresponds to describing how the parts achieve the object's function. It is possible to repeat the strategy for each of the subparts. This is similar to the schema recursion for the constituency schema presented in [McKeown 85]. An example is shown in Figure 5-12. In this description of a telephone, the process trace is first chosen to explain how the telephone transmits soundwaves in terms of the functions of its subparts, the transmitter and the receiver. The process trace then continues by explaining how the transmitter performs its function; that is, the step ''changes soundwaves into current'' is decomposed into its substeps.

Description of an object made up of several functional complex parts

This description is generated by TAILOR to describe a telephone, when the user model is empty (i.e., the user is naive) and more information is requested about the transmitter.

The telephone is a device that transmits soundwaves. Because a person speaks into the transmitter of the telephone, a current is produced. Then, the current flows through the line into the receiver. This causes soundwaves to be reproduced.

Process description at the top-level

More about (transmitter, receiver) ?: transmitter

Process description for the *transmitter*. The function of the transmitter is decomposed into its substeps.

The transmitter is a microphone with a small diaphragm. Because a person speaks into the microphone, soundwaves hit the diaphragm of the microphone. This causes the diaphragm to vibrate. When the intensity of the soundwaves increases, the diaphragm springs forward. This causes the granules of the button to be compressed. The compression of the granules causes the resistance of the granules to decrease. This causes the current to increase. Then, when the intensity decreases, the diaphragm springs backward. This causes the granules to be decompressed. The decompression of the granules causes the resistance to increase. This causes the current to decrease. The vibration of the diaphragm causes the current to vary. The current varies, like the intensity varies.

Figure 5-12: Substeps arising because of subparts

5.6 Strategy representation

The constituency schema and the process trace are both represented in TAILOR as augmented transition networks (ATN) [Woods 73]. The constituency schema was represented using this formalism in [McKeown 85], with each arc representing a rhetorical predicate. Using the same formalism for the two strategies obtains the control structure necessary to switch from one strategy to the other, thus gaining the ability to employ both strategies in the same description.

The ATN corresponding to the constituency schema is shown in Figure 5-13. The arcs correspond to the predicates of the schema, thus defining the type of information to be taken from the database to include in the description. The predicates are:

- the *identification* predicate, which represents the more general concept of which the present object is an instance.
- the *constituency* predicate, which gives the components of an object, if there are any.
- the *attributive* predicate, which provides different attributes of an object (such as its shape or material).
- the *cause-effect* predicate, which provides some causal relations between entities or relations.

When several facts in the knowledge base can match the predicates or when several predicates are available, one is chosen based on factors such as focus. This will be explained in detail in Chapter 7.

The process of traversing the ATN for the constituency schema strategy to describe a *microphone* is shown in Figure 5-14. First, the identification predicate is applied to the loudspeaker. It provides the superordinate of the object together with the function of the object. Second, the arc corresponding to the constituency predicate is taken, and the parts of the loudspeaker are retrieved, together with their properties. Finally, the depth-attributive predicate is matched against the knowledge base for each subpart.

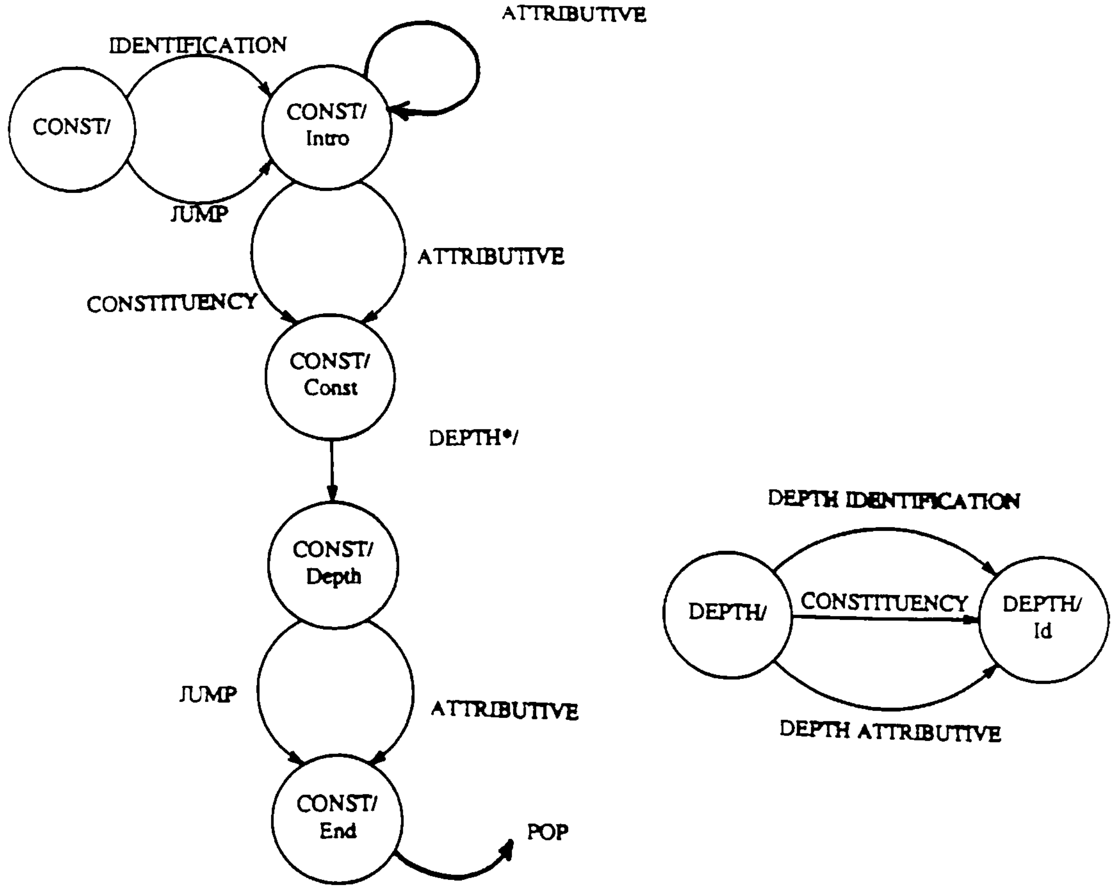


Figure 5-13: The Constituency Schema

```
; Stepping through the Constituency Schema to describe
; a LOUDSPEAKER
Applying the predicates to LOUDSPEAKER:

  Identification predicate: DEVICE; (used-for:
                              change current into soundwaves)

  Constituency   predicate: ARMATURE
                              (shape ring, material permendur)

                            COIL

                            MAGNET
                              (mobility permanent, shape ring)

                            DIAPHRAGM
                              (shape dome, material paper,
                               size large, thin)

                            GAP

  Depth-attributive for DIAPHRAGM: mounted on the poles of the magnet

  Depth-attributive for GAP: contains air
                             is between poles and diaphragm

  Depth-attributive for COIL: mounted on the magnet
```

English output:

```
The loudspeaker is a device that changes current
into soundwaves. The loudspeaker has a
permendur, ring-shaped armature, a coil, a
ring-shaped permanent magnet, a gap and a paper
thin large dome-shaped diaphragm. The diaphragm
is mounted on the poles of the magnet. The gap
contains air. The gap is between the poles and
the diaphragm. The coil is mounted on the
magnet.
```

Figure 5-14: Stepping through the Constituency Schema

The information included in a description generated using the constituency schema is dictated by the rhetorical predicates contained in the schema and does not depend on the structure of the knowledge. The semantics of the predicate dictate what information to retrieve, and the graph (or schema) dictates the order of the predicates. This is a property of declarative strategies.

The network for the process trace is shown in Figure 5-15. In this network, the arcs dictate how to trace the knowledge base to form a process description, mainly by following the causal links in the knowledge base. Following the process trace, the knowledge base is traversed in a specific order. These arcs are not rhetorical predicates as in the network corresponding to the constituency schema. Rather, the arcs indicate the following actions:

- *Next-main-link*: This arc dictates to follow the next link on the main path.
- *Long-side-chain?*: This arc tests for a long side chain that gets re-attached to the main path and needs to be followed first. If the test returns True, the subnet *long-chain* is called. This subnet dictates to follow the links of the side chain.
- *Short-Side-link?*: A test is made to see whether there is a side link coming off an event at this point. The test includes checking the importance of the side link and the length of its associated side chain. If this link is important and short, the arc will be taken, i.e., the side link will be included. Since the side links were marked while the program was obtaining the main path, it is also possible to check for the number of short side links. It there are many side links, they will be grouped at the end instead of being mentioned at this point.
- *Attributive*: This arc is similar to the attributive predicate in the constituency schema. Hardly any description is purely process oriented; information about parts is presented when the parts are mentioned during the process description. If information about a part just introduced is available in the knowledge base, this arc will be taken.
- *Substeps?*: If an event at this point can be divided into substeps and the description does not have to be short, each substep with its associated chain is followed. To traverse the substeps, the subnet (or subroutine) *substep* is called for each substep. This subroutine is very similar to the main graph but does not allow for a further decomposition of events.

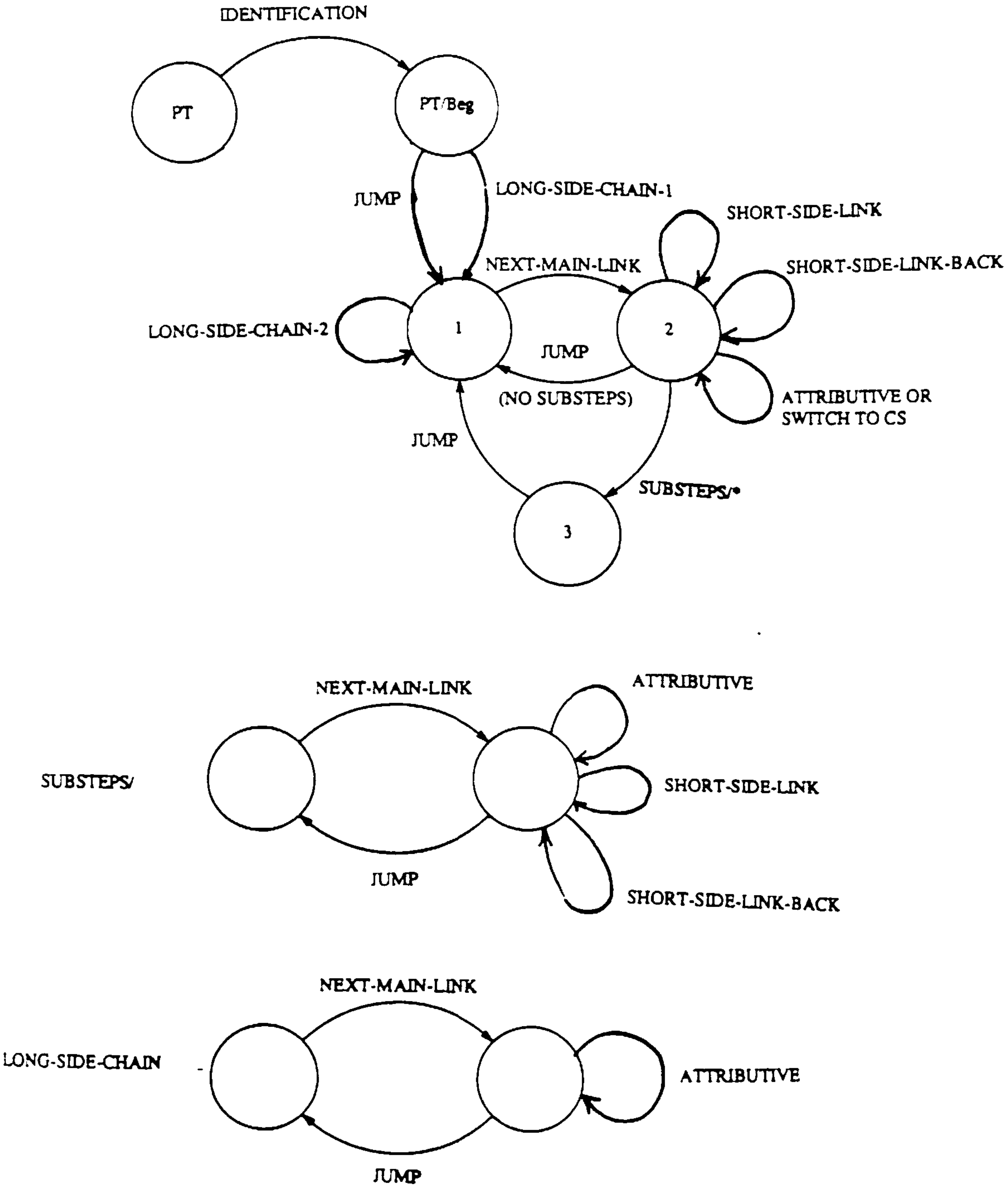


Figure 5-15: The Process Trace

17 If a short description is desired, the program chooses not to follow the substeps.

The process of stepping through the ATN for the process trace strategy is shown in Figure 5-16. In this figure, I show the process description for the loudspeaker. This description includes explaining the causal links at the substep level and includes a short side link, an analogical link. This process description is obtained by applying the process strategy to the subset of the knowledge base previously shown in Figure 5-5. The sentences that resulted from following the main path are underlined. The causal links (M-CAUSES) are given in curly brackets, together with the two events they relate.

The descriptions shown in this section are actually only a subset of the descriptions TAILOR generates. Full descriptions include a statement to first introduce the object and attributive information about some subparts. This information is omitted here as the emphasis is on tracing links between events. Full descriptions are presented in other parts of the thesis.

In Figure 5-17, I show the process description for the dialing mechanism, which involves following a side chain. The corresponding English translation from TAILOR, shown in the comments, was presented previously in Figure 5-8. The knowledge base for the dialing mechanism was shown in Figure 5-7. In tracing the process description, it is necessary to first mention the side chain before mentioning the events on the main path, as the events on the side chain are preconditions for the events on the main path.

A text generated using the process trace follows the knowledge base very closely. The arcs of the schema do not dictate which information to include, as they did in the

---

[17] A user can explicitly ask for a further decomposition if required.

Process trace for the loudspeaker

Substeps and an important isolated side link (an analogical link)

Main Path

First Causal Link:

```
{M-CAUSES} relates the two relations:          ; The variation of the
           [CURRENT] VARIES                    ; current causes the field
           [FIELD] VARIES                      ; of the magnet to vary.
```

Substeps: first substep: "current increases"
second substep: "current decreases"

***Causal link in the* first substep:**

```
{M-CAUSES} relates the two relations: ; When the current
           [CURRENT] INCREASES        ; increases  the field
           [FIELD] INCREASES          ; increases.
```

***Next causal link in the substep chain*:**

```
{M-CAUSES} relates the two relations: ; Because the field
           [FIELD] INCREASES          ; increases the poles of
           [POLES] ATTRACT [DIAPHRAGM ; the magnet attract the
                                      ; diaphragm of the
                                      ; loudspeaker.
```

***Next causal link in the substep chain*:**

```
{M-CAUSES} relates the two relations: ; The poles attracting the
           [POLES] ATTRACT [DIAPHRAGM ; diaphragm causes the
           [DIAPHRAGM] MOVES-FORWARD  ; diaphragm to move
                                      ; forward.
```

***Causal link in the* second substep:**

```
{M-CAUSES} relates the two relations: ; The current decreasing
           [CURRENT]  DECREASES       ; causes the field to
           [FIELD] DECREASES          ; decrease.
```

***Next causal link in the substep chain*:**

```
{M-CAUSES} relates the two relations: ; This causes the poles
           [FIELD] DECREASES          ; to release the
           [POLES] RELEASES [DIAPHRAGM]; diaphragm.
```

***Next causal link in the substep chain*:**

```
{M-CAUSES} relates the two relations: ; Because the poles
           [POLES] RELEASES [DIAPHRAGM]; release the diaphragm
                                      ; the diaphragm
           [DIAPHRAGM] MOVES-BACKWARD ; move backward.
```

Back to the main path

***Next causal link in the main path*:**

```
{M-CAUSES} relates the two relations:          ; The variation of the
           [FIELD] VARIES                      ; field causes the
           [DIAPHRAGM] VIBRATES                ; diaphragm to vibrate.
```

(Continued in the next Figure)

Figure 5-16: Including substeps and an isolated side link

Figure 5-16 Continued

Substeps: first substep: “diaphragm moves forward
second substep: “diaphragm moves backward

Causal link in the **second substep:**

```
{M-CAUSES} relates the two relations: ; Because the diaphragm
     [DIAPHRAGM]  MOVES-BACKWARD       ; springs backward the
     [SOUNDWAVE-INTENSITY] INCREASES   ; soundwave-intensity
                                       ; increases.
```

Causal link in the **first substep:**

```
{M-CAUSES} relates the two relations: ; The diaphragm springing
     [DIAPHRAGM]  MOVES-FORWARD        ; forward causes the
     [SOUNDWAVE-INTENSITY] DECREASES   ; soundwave-intensity
                                       ; to reduce.
```

Back to the main path

Next causal link in the main path:

```
{M-CAUSES} relates the two relations:         ; The vibration of the
          [DIAPHRAGM] VIBRATES                 ; diaphragm causes the
          [SOUNDWAVE-INTENSITY] VARIES         ; soundwave-intensity to
                                               ; vary.
```

Isolated short side-link

Analogical link:

```
{M-CORRESPONDS-TO} relates the two relations: ; The soundwave intensity
          [SOUNDWAVE-INTENSITY] VARIES         ; varies like the current
          [CURRENT] VARIES                     ; varies.
```

English output:

The variation of the current causes the field of the magnet to vary. When the current increases the field increases. Because the field increases, the poles of the magnet attract the diaphragm of the loudspeaker. The poles attracting the diaphragm causes the diaphragm to spring forward. The current decreasing causes the filed to decrease. This causes the poles to release the diaphragm. Because the poles release the diaphragm, the diaphragm springs backward. The variation of the field causes the diaphragm to vibrate. Because the diaphragm springs backward, the soundwave-intensity of the soundwaves increases. The diaphragm springing forward causes the soundwave-intensity to reduce. The vibration of the diaphragm causes the soundwave-intensity to vary. The soundwave-intensity varies, like the current varies.

[Figure 5-16 Cont'd: Including substeps and an isolated side link]

Long side chain

First Causal Link of the side-chain:

```
  {M-CAUSES} relates the two relations:      ; Because a person dials, the
      [ONE] PUSHES [DIAL]                    ; dial assembly of the pulse
      [DIAL-ASSEMBLY] TURNS-CLOCKWISE        ; dialer turns clockwise.
```

Next Causal link of the side chain:

```
  {M-CAUSES} relates the two relations:      ; The dial assembly turning
      [DIAL-ASSEMBLY] TURNS-CLOCKWISE        ; clockwise causes the
      [CIRCULAR-SPRING] COMPRESSED           ; spring to be compressed.
```

Point at which the side chain is reattached to the main path:

```
  {M-ENABLES} relates the two relations:     ; The compression of the
      [CIRCULAR-SPRING] COMPRESSED           ; spring enables the
      [CIRCULAR-SPRING] DECOMPRESSED         ; spring to be decompressed.
```

Main path

First Link:

```
  {M-ENABLES} relates the two relations:     ; A person dialing
      [ONE] PUSHES [DIAL]                    ; enables the person  to
      [ONE] RELEASES [DIAL]                  ; release the dial.
```

Next Causal Link:

```
  {M-CAUSES} relates the two relations:      ; This causes the
      [ONE] RELEASES [DIAL]                  ; spring to be
      [CIRCULAR-SPRING] DECOMPRESSES         ; decompressed.
```

Next Causal Link:

```
  {M-CAUSES} relates the two relations:      ; The compression of the spring
      [CIRCULAR-SPRING] DECOMPRESSES         ; causes the dial assembly to
      [DIAL-ASSEMBLY] TURN-COUNTERCLOCKWISE; turn counterclockwise.
```

Next Causal Link:

```
  {M-CAUSES} relates the two relations:      ; This causes the gear
      [DIAL-ASSEMBLY] TURN-COUNTERCLOCKWISE; of the pulse dialer
      [SMALL-GEAR] TURNS                     ; to turn.
```

Analogical Link:

```
  {M-PROPORTIONAL}                           ; The dial assembly turns
      [DIAL-ASSEMBLY] TURN-COUNTERCLOCKWISE; counterclockwise proportion-
      [SMALL-GEAR] TURNS                     ; ally to the way the gear turns.
```

Next Causal Link:

```
  {M-CAUSES} relates the two relations:      ; Because the gear turns, the
      [SMALL-GEAR] TURNS                     ; protrusion of the gear hits
      [PROTRUSION] HITS [SWITCH-LEVER]       ; the lever of the switch.
```

Next Causal Link:

```
  {M-CAUSES} relates the two relations:      ; This causes the lever
      [PROTRUSION] HITS [SWITCH-LEVER]       ; to close the switch.
      [SWITCH-LEVER] CLOSES
```

Next Causal Link:

```
  {M-CAUSES} relates the two relations:      ; Because the lever closes
      [SWITCH-LEVER] CLOSES                  ; the switch, current
      [CURRENT-PULSE] PRODUCED               ; pulses are produced.
```

Figure 5-17: Process trace for the dialing mechanism, including a side chain that gets re-attached to the main path

---

constituency schema, but rather help follow the knowledge base in a coherent manner.

5.7 Open problems

In formalizing the process trace, I have assumed that the start and goal states of the sequence were known. The algorithm outlined in this section finds the main path of events by tracing the events in the knowledge base from the goal state to the start state. If no start state is provided, the search will end when the program reaches an event with no further links. If the knowledge base is very complex, this may result in a very long search. If no goal state is provided, the program cannot search for the main path and will not be able to provide a process explanation. It is unclear what strategy should be used to decide when to start and end a process description should the program have no clue as to what the start and goal states are.

The other assumption made in formalizing this strategy is that there is only one main chain of events that allow a device to perform its function. This appeared to be the case in the texts I examined. If there are several, the first one found will be chosen as the main path, and the other one will be treated as a side chain. It is unclear whether that is correct or whether there are other criteria that would allow the program to choose among several equally likely or concurrent main paths.

Lastly, while the program used to obtain the main path is able to detect loops in the main path, the resulting text at this point will not indicate that a loop was found. This can happen in case of a feedback loop. Consider for example the description of an oscillator shown in Figure 5-18. Although the program detected a loop, it is not explicitly stated in the text. Although a few changes would make the identification of loops in the generated texts possible, it is not clear what would be the best way to explain such feedback loops.

Oscillator

An oscillator is a device that produces a varying current when a battery produces a current. Because the battery produces a current, the transistor turns on. The transistor turning on is also caused by the capacitor discharging through the resistor. The transistor turning on causes the capacitor to charge. This causes the transistor to turn off. Because the transistor turns off, the capacitor discharges through the resistor. The resistor has low resistance. The capacitor discharging through the resistor causes the varying current to be produced.

Figure 5-18: Example of a feedback loop

5.8 Summary

In this chapter, I have presented in detail two discourse strategies that can be used to produce descriptions of complex devices. The constituency schema is a declarative strategy identified by McKeown [85]. It consists of rhetorical predicates. Unlike other discourse strategies, the process trace is procedural in nature, in that it consists of directives for tracing the underlying knowledge base. By representing the procedural strategy in the same formalism as the declarative strategy, I am able to to use the two strategies together to produce a single text. This will be explained in Chapter 6.

6. Combining the strategies to describe devices for a whole range of users

6.1 Introduction

In the two previous chapters, I have proposed using the constituency schema to describe an object to an expert user and the process trace for a naive user. Chapter 5 presented these strategies in detail. TAILOR can select a strategy based on the user's assumed domain knowledge level. Users are not necessarily strictly naive or expert, however, but can fall anywhere along a knowledge spectrum between the extremes of naive and expert.

In Chapter 3, I defined an expert user as one who knows many of the objects contained in the knowledge base and the basic underlying concepts necessary to understand the devices' mechanisms. A naive user, on the other hand, does not understand the basic underlying concepts and does not know about any of the objects included in the knowledge base. These two types of users clearly represent extremes. Many users are likely to fall somewhere between these extremes, as they might have some knowledge about the domain, too much to be classified as naive users but not enough to be considered experts. A generation system that can only tailor to the two extreme user types is limited and not appropriate for many users. The ability to generate descriptions aimed at users intermediate amounts of knowledge as well as novices and experts is required.

One way to address this problem is to define several stereotypes and categorize users in terms of these types. A discourse strategy can be assigned to each user type, and tailoring can be done based on the stereotype assigned to a particular user.

One problem with this approach is the necessity to define a set of somewhat arbitrary stereotypes, arbitrary both in terms of their number and in what they represent. Furthermore, users are forced into stereotypes, and a generation system comprising these user types can tailor its answers only to these predefined categories and is thus still limited.

In this chapter, I show how, by explicitly representing a user's domain knowledge, TAILOR is able to generate descriptions for a whole range of users rather than just an *a priori* set of user stereotypes. Based on the user model, TAILOR can merge the two strategies automatically in a systematic way to produce a wide variety of descriptions for users who fall anywhere between the two extremes of naive and expert. Therefore, TAILOR can also adapt itself to the user's continuously changing expertise.

I also show how, by representing both strategies using an augmented transition network, it is possible to easily combine the strategies in many different ways. This can be generalized to any number of strategies and thus gives a generation system greater flexibility, allowing it to generate a wider variety of texts than otherwise possible.

6.2 The user model contains explicit parameters

Besides the aforementioned problems with stereotypes, it would be difficult in the domain of complex devices to categorize users in terms of several types ranging from naive to expert. Deciding on the number and type of stereotypes that might exist between the extremes would require partitioning the knowledge base into several categories and deciding that knowledge about one type of objects indicates more expertise than knowledge about another set of objects. As discussed in Chapter 3, this would not be appropriate for a complex domain such as ours.

Instead of attempting to define various stereotypes for users falling inside the knowledge spectrum, I chose to explicitly represent a user's knowledge about the domain. This gives TAILOR the ability to generate descriptions to a whole range of users, ranging from naive to expert, and not just to a predefined set of user types.

In Chapter 3, I introduced the kinds of user domain knowledge that might affect the type of descriptions that can be provided in the domain of complex physical objects. They were:

- knowledge about specific items in the knowledge base. Users who fall along the knowledge spectrum may know about some items in the knowledge base and not others. I term this having *local expertise* about some objects [Paris 84].
- knowledge about various underlying concepts.

A user model in TAILOR contains explicit information about these two types of knowledge. It indicates the objects about which the user has local expertise, and a list of basic concepts the user understands. By using these explicit parameters to indicate the user's level of expertise, TAILOR does not require a preconceived set of user types, and there is no need to fit users into known types. The user model parameters can be set to any value, and TAILOR will generate a description accordingly.

Examples of user models are shown in Figure 6-1. In (a), the user has local expertise about the microphone and understands the concept of electricity. Lack of expertise is assumed about other objects and concepts. In (b), the user is a naive user, and the user model is empty. In (c), the user has local expertise about both the microphone and the telephone. The concepts of electricity and magnetism are also understood.

(a) The user has local expertise about the microphone and understands the basic concepts:

Local expertise: *microphone*
Basic concepts: *electricity*

(b) The user is a naive user:

Local expertise: *nil*
Basic concepts: *nil*

(c) The user has local expertise about the microphone and the telephone and understands the electricity and magnetism:

Local expertise: *microphone, telephone*
Basic concepts: *electricity, magnetism*

Figure 6-1: Representing the user model explicitly

6.3 Generating a description based on the user model

Before starting to generate a text, TAILOR must decide whether to use the constituency schema or the process trace for the overall framework of the text. While constructing the description, TAILOR can switch to the other strategy, thus combining the two strategies in one text. This will be done for users with intermediate levels of expertise. There is therefore a need to specify 1) how to choose an initial strategy and 2) how to combine the strategies. In TAILOR, the decision on which strategy to use at any point is based on the user model.

6.3.1 Choosing a strategy for the overall structure of the description

To choose the initial strategy, TAILOR first checks whether the frame representing the object to be described in the knowledge base contains process information. If the object has no such information in its frame, the constituency schema is chosen by default. This will happen in two cases. First, when the object's function cannot be decomposed into a sequence of events. For example, a ''cover'' has no mechanism associated with it. Second, when the object's function and the way it is performed is identical to that of its superordinate in the generalization hierarchy. For example, the function of the *pulse telephone* is the same as that of a *telephone*. Process information in this case is stored in the frame of the superordinate. The *pulse telephone* frame contains the part that differs from the telephone, namely the *pulse dialer*. Although the constituency schema will be chosen for the initial strategy of the description of a pulse telephone, it will be possible to switch to the process trace to explain the function of its superordinate. Similarly, it will be possible to switch to the process trace to explain the function of the ''new'' (or different) part, i.e., the pulse dialer. Examples will be given shortly to illustrate this.

If there was process information (or a mechanism) associated with the object, it is possible to describe the object with either strategy. TAILOR examines the user model to check the user's local expertise. If the user model does not indicate any local expertise about either the object to be described or its superordinate in the generalization hierarchy, the process trace is chosen. Otherwise, the constituency schema is chosen. (If local expertise is indicated about the superordinate of the object, the user will most likely be able to infer the processes for the object given his or her knowledge about the superordinate, and, therefore, the constituency schema is still adequate for the description.)

The constituency schema is also chosen when the user model indicates local expertise about all or most of the functionally important subparts of the object to be described.

A distinction is made between parts that play a significant role in the mechanical process of a device and other parts because knowledge about the former will help comprehend the mechanism of the device. For example, the *housing* of a telephone does not play an important role in allowing the telephone to transmit soundwaves. Knowing about the *housing* or about a *cover*, the housing superordinate in the generalization hierarchy, does not help a user understand how the telephone works. Understanding how a microphone and a loudspeaker work, on the other hand, is useful since the transmitter of a telephone is essentially a microphone, and the receiver a loudspeaker.

Functionally important parts in TAILOR are defined to be the parts that are mentioned when the object's mechanism is explained or the main parts that have a mechanism themselves. (''Main parts'' denotes the first level in the parts hierarchy.) For example, the pulse dialer is a functionally important part of the pulse telephone, as it has a mechanism associated with its frame in the knowledge base.

If the user already knows about most of the functionally important subparts of a device, he or she is likely to be able to infer how the parts fit together functionally (as was pointed out in the discussion of the strategies, in Chapter 4), so it is not necessary to include a process description for the object. The overall description of a text in that case will be based on the constituency schema. ''Most'' is represented in TAILOR as a parameter that can be set to any value for experimental purposes.[18] If the user does not have local expertise about most of the subparts of the device, the

---

[18] Its particular value in the current implementation is half the number of functionally important parts.

process trace is chosen, and a process explanation of how the subparts function together is provided.

To summarize, the constituency schema is chosen as the overall structure of the text in the following cases:

1. The object has no mechanism associated with its frame in the knowledge base;
2. The user model indicates local expertise about either the object or its superordinate in the generalization hierarchy;
3. The user model indicates that the user knows about most of the functionally important parts of the object.

The process trace is chosen as the initial strategy in all other cases.

Whether the constituency schema or the process trace is chosen as the overall strategy for the description, a mixture of structural and process information can still be provided depending on the user model. For example, if the constituency schema is chosen initially, the process trace can be used for the unknown parts, providing the user with a process explanation for these parts. Likewise, if the process trace is taken at first, structural information about the parts already known to the user is still included. ''Most'' only determines the initial strategy (i.e., the overall structure of the text). It does not determine whether a mixture of structural and functional information will be included.

6.3.2 Combining the strategies

To generate a description aimed at users with intermediate levels of expertise, the two strategies must be combined. Consequently, it is necessary to specify:

- *when* it is possible to switch from one strategy to the other
- *the conditions* under which it is possible to switch.

Although it would be hard without a thorough psychological study to specify exact conditions necessary for switching from one strategy to the other, I have identified heuristics that determine plausible ways to mix the strategies.

6.3.2.1 Decision points within the strategies

Whenever an object is introduced and needs to be described, the system must decide whether to provide chiefly structural or functional information. Thus, each time an object needs to be described, a strategy must be chosen, and a strategy switch might occur. This gives us some clear decision points:

- Within the constituency schema:
  - After the *identification* predicate: once the parent of an object in the generalization hierarchy (the superordinate) has been introduced, a process trace can be provided for the superordinate.
  - After the *constituency* predicate: after mentioning the parts of an object, the constituency schema dictates filling the depth identification predicate for each subpart. Instead, a functional description of one or more of the parts can be given. This was done in the *Encyclopedia of Chemical Technology* text presented in Figure 4-15 in Chapter 4, for example, where *detectors* were described with a process explanation.
- Within the process trace:
  - When a part is introduced while traversing the causal links, the process strategy dictates to include attributes of this part to describe it. Here, we could also choose to describe the part more fully with the constituency schema.
  - When the subparts have to be described, the constituency schema can be used to provide structural information about them instead of including functional information.

Figure 6-2 and 6-3 summarizes the two strategies with the decision points.

6.3.2.2 Switching strategy when the constituency schema is chosen initially

Like the initial decision for a strategy, the decision to switch strategy is mainly based on the user model. If the initial strategy is the constituency schema, TAILOR considers switching to the process trace for the superordinate and for the subparts of the device being described. The test for switching is largely the same as the initial

Constituency Schema (with decision points)

```
Identification (introduction of the superordinate)
    If there is no local expertise for the superordinate
        do a Process Trace for the superordinate before proceeding.
Constituency (description of the subparts)
    For each part, do:
        If there is local expertise on this part (or its superordinate),
         do Depth-identification
        Else do a  Process Trace for the part
Attributive
```

Figure 6-2: The Constituency Schema strategy and its decision points

Process Trace (with decision points)

```
Next causal link
Properties of a part mentioned during the process trace
    If a fuller description of the part is desired, do
        Constituency Schema for the part
Substeps
Back to next causal link

Repeat for each of the subparts:
    If there is local expertise on this part (or its superordinate),
     do Constituency Schema
        Else do a  Process Trace
```

Figure 6-3: The Process Trace strategy and its decision points

test. TAILOR first checks whether the object under consideration (either the superordinate or each of the subparts) has some process information in its frame, or in its superordinate's frame. If so, TAILOR examines the user model to see whether the object or its superordinate is included in the user's local expertise. Switching is considered only if it is not included.

Two more conditions are tested before switching occurs. If the process explanation of the object includes basic concepts of which the user lacks knowledge, the process explanation would involve steps the user is not likely to comprehend. As a result the process explanation will not be understandable. For example, if the process explanation of a part involves the concepts of electricity and magnetism, it will not be informative if the user does not understand them. In that case, switching does not occur.

TAILOR keeps a record that switching was not allowed because of lack of knowledge about basic concepts, and, after the initial description is given, the user is given the option to ask for the process description for the part, even though it contains concepts he or she might not understand.

The last condition that needs to be satisfied before the switch can occur is a test on the length of the text constructed so far and, in case of subparts, the number of parts that need to be described using the process trace. This is done to avoid generating very long texts. A variable, which can be arbitrarily set to any value, sets a threshold beyond which the process trace will not be included at this point.[19] TAILOR remembers that the process explanations for some parts were not included because of length, and, after the text corresponding to the initial strategy is produced, TAILOR asks the user whether it should provide these explanations. When all the conditions are satisfied, TAILOR decides to switch to the process trace; otherwise, it continues with the constituency schema.

---

[19] Note that this test can also be used to test schema recursion while going through the constituency schema, when the user model indicates local expertise about a part and switching is not allowed.

6.3.2.3 Switching strategy when the process trace is chosen initially

If the process trace is chosen for the overall structure of the text, a test similar to the one described above is performed when an object has been introduced as part of the process explanation. In this case, however, switching is considered only if the user model indicates local expertise about the part just introduced. As above, a test on the length of the text generated so far and the number of parts to be described using the constituency schema is also done to determine whether the switch should occur when the part is mentioned as part of the process or, at the user's request, after the initial text is produced. Again, when all the conditions are satisfied, TAILOR switches to the constituency schema for the part.

6.4 Examples of texts combining the two strategies

In this section, I present a number of examples of texts that demonstrate that the ability to combine the strategies based on the user model allows TAILOR to generate a wide variety of texts tailored to users along the knowledge spectrum.

The first three texts are based on the telephone. This part of the knowledge base is shown in Figure 6-4. It has been greatly simplified to include only the generalization hierarchies and the main parts of the telephone. The generalization hierarchies are indicated in thick broken lines. The other lines represent *parts-of* relationships. The mechanism of objects that have process information in their frame is summarized in the labelled box in angle brackets ''[ ].'' For example, the telephone has process information that allows it to ''transmit soundwaves.''

As shown in the figure, the *telephone* has two functionally important parts: the transmitter and the receiver. The microphone is the transmitter's superordinate in the generalization tree, while the loudspeaker is the receiver's superordinate. Also indicated in the figure, the *pulse telephone* is an instance of a telephone, with a special part, the pulse dialer.

Figure 6-5 presents a description of the telephone. The next two figures provide descriptions of the pulse telephone. In these figures, the process trace is shown underlined, and the right hand column indicates the structure of the text and when switching occurs.

In Figure 6-5, the user model indicates local expertise about the *loudspeaker*. As *most* of the functionally important parts are known to the user,[20] TAILOR initially chooses the constituency schema. As a result, the overall structure of the text is a description organized around the subparts of the telephone and their attributes.

The *telephone* is first identified with its purpose, and its subparts are introduced. Then, the constituency schema directs TAILOR to provide attributive information about each of the subparts. At this point, it is also possible to switch to the process trace for some subparts. TAILOR examines each subpart to decide whether there is a need to switch strategy for any of them. Two of the subparts have a mechanism associated with them (or with their parent in the generalization hierarchy): the transmitter and the receiver. It is thus possible to describe either of these parts with the process trace. Because the user model indicates local expertise about the *loudspeaker*, the receiver's superordinate, TAILOR decides to use attributive information and not a process explanation to describe this part. As no local expertise about the transmitter or the microphone is indicated in the user model, TAILOR chooses to provide process information for the transmitter, switching momentarily to the process trace. After the process explanation for the transmitter is completed, TAILOR returns to the constituency schema, and attributive information is provided about each of the other subparts.

---

[20] *Most* is set to *one half* of the functionally important parts in these examples.

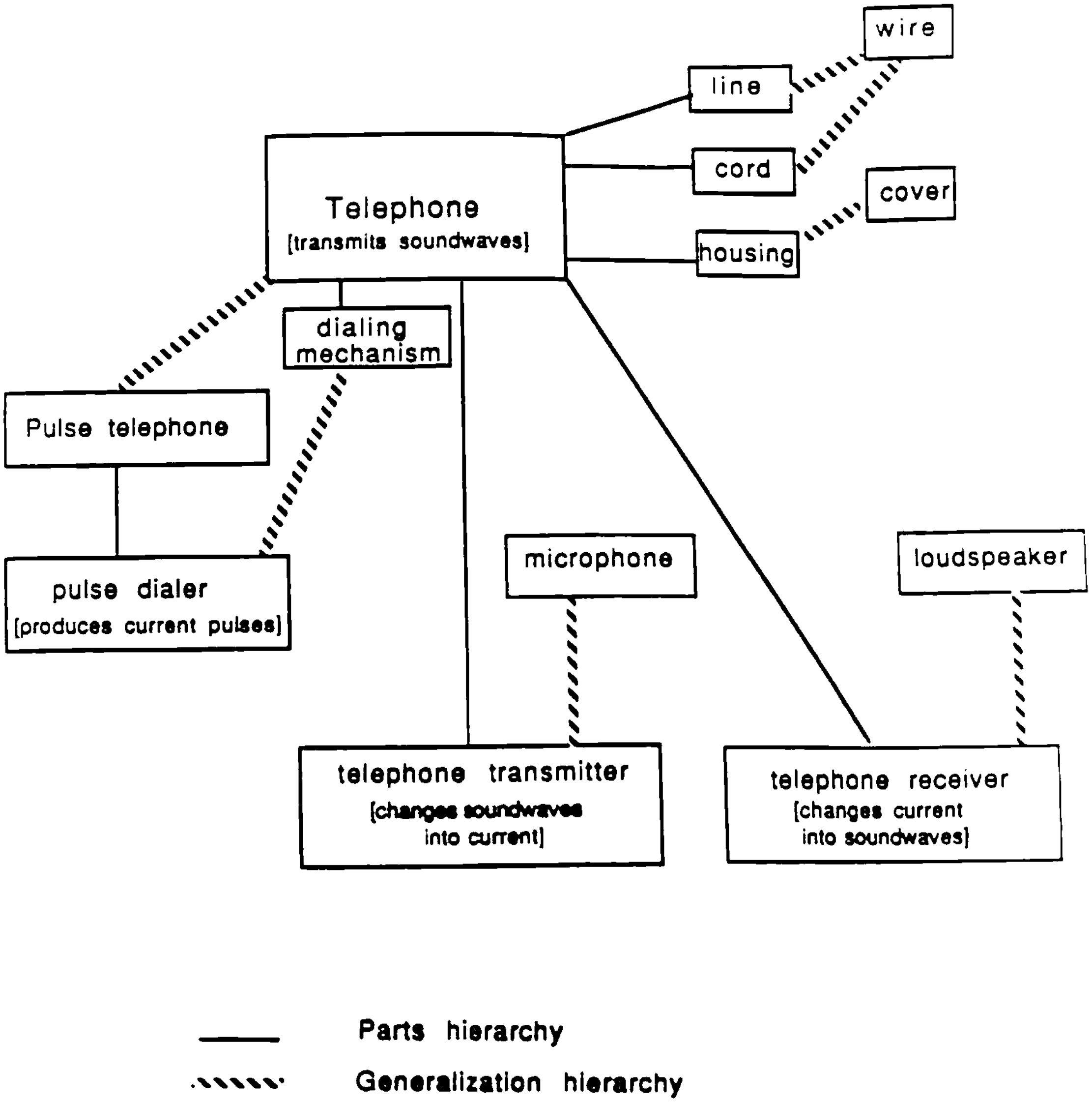


Figure 6-4: Simplified portion of the knowledge base for the telephone

Telephone

User Model: Local Expertise -- *Loudspeaker*
Basic Concepts -- *electricity*

The telephone has two main functional parts: the transmitter (an instance of a microphone) and a receiver (an instance of a loudspeaker). Because the users knows one of the two parts of the telephone, the system decides on the *constituency schema strategy* at first. However, before providing structural information about each subpart, the system consults the user model and decides to switch strategy to describe the process of the transmitter since the user has no local expertise about it.

TAILOR-87 output:

```
 The  telephone  is  a  device  that  transmits    Identification
soundwaves.  The telephone has a housing that
has  various  shapes  and  various  colors,  a     Constituency
transmitter   that   changes   soundwaves   into
current,  a  curly-shaped  cord,  a  line,  a
receiver to change current into soundwaves and
a dialing-mechanism.   The transmitter is a        Switch to process
microphone with a small diaphragm.  A person       trace for the
speaking  into  the  microphone  causes  the       transmitter
soundwaves  to  hit  the  diaphragm  of  the
microphone.     The  soundwaves  hitting  the
diaphragm causes the diaphragm to vibrate.  The
vibration of the diaphragm causes the current
to  vary.    The  current  varies,  like  the     Attributive information
intensity  varies.      The   receiver   is   a   for the other subparts
loudspeaker with a small aluminium diaphragm.
The  housing  contains  the  transmitter  and  it
contains   the   receiver.     The   housing   is
connected to the dialing-mechanism by the cord.
The line connects the dialing-mechanism to the
wall.
```

Figure 6-5: Combining the strategies: using the constituency schema as the overall structure of the text and switching to the process trace for one part

The next two texts are descriptions of a pulse telephone, given different user models. In the first text, Figure 6-6, the user model indicates local expertise about telephones.

The pulse telephone does not have a mechanism associated with its frame, as the process mechanism is associated with the telephone, its parent in the generalization tree. As a result, the constituency schema is chosen for the overall structure of the text. The pulse telephone is first introduced in terms of its superordinate, the telephone. At this point, TAILOR has the option of switching to the process trace to describe the telephone, but, as the user model indicates expertise about this object, the program does not do so. Instead, the description continues with the constituency schema. The subparts of the pulse telephone are provided. Now, TAILOR has the option of switching strategies for each subpart. Because local expertise was indicated about the *telephone*, TAILOR assumes the user also has knowledge about the telephone's parts, the transmitter and the receiver.[21] The pulse telephone has a part that the telephone does not have, however, namely the pulse dialer, an instance of a *dialing mechanism*. No expertise is indicated in the user model about this part, and TAILOR switches to the process trace to describe it. Attributive information is provided for the other parts.

In Figure 6-7, the user model shows local expertise about the *pulse dialer*. As in the previous description, the constituency schema is chosen by default for the overall structure of the description, as there is no process explanation associated with the pulse telephone. The description thus starts by introducing the object's superordinate, as in Figure 6-6. Switching is now considered to describe the telephone. Unlike in

---

[21] This is one of the inference rules employed to obtain information about the user's domain knowledge, as described in Section 3.2.

Telephone with a pulse dialer

User Model: Local Expertise -- *Telephone*
Basic concepts? -- *electricity*

The *constituency schema* is chosen at first as no process mechanism is associated with the pulse telephone. When the superordinate (the telephone) is introduced, switching does not occur because the user model indicates local expertise about it. The pulse telephone has one part that the telephone does not have, that is the pulse dialer. As this part is new and no local expertise is indicated in the user model about it, TAILOR switches to the *process trace* to describe this part.

TAILOR-87 output:

The pulse-telephone is a telephone with a pulse-dialer.	Identification
The pulse-telephone has a pulse-dialer to produce current-pulses, when a person dials, a housing that has various shapes and various colors, a receiver to change current into soundwaves, a curly-shaped cord, a line and a transmitter that changes soundwaves into current.	Constituency
The pulse-dialer is a dialing-mechanism. A person dialing causes the dial-assembly to turn clockwise. The dial is round. The dial assembly turning clockwise causes the spring to compress. The spring is circular. A person dialing enables the person to release the dial. This causes the spring to decompress. The decompression of the spring causes the dial to turn counterclockwise. This causes a gear to turn. The gear is small. The gear turning causes the protrusion of the gear to hit a lever of the switch. This causes the switch to close. This causes current-pulses to be produced.	Switching to the process trace for the pulse dialer
The transmitter is a microphone with a small diaphragm. The receiver is a loudspeaker with a thin small metal disc-shaped diaphragm. A dialing-mechanism is connected to a wall by the line. The housing is connected to the dialing-mechanism by the cord. The housing contains the transmitter and it contains the receiver.	Attributive information for the other parts

Figure 6-6: Starting with the constituency schema and switching to the process trace for the new part

Telephone with a pulse dialer; long description requested

User Model: Local expertise -- *pulse dialers*
Basic Concepts -- *electricity*

TAILOR-87 output:

The pulse-telephone is a telephone with a pulse-dialer.	Identification
The telephone is a device that transmits soundwaves. Because a person speaks into the transmitter of the telephone, a varying current is produced. Then, the varying current flows through the line into the receiver. This causes the soundwaves to be reproduced.	Switch to process trace for the telephone
The pulse-telephone has a pulse-dialer that produces current pulses when a person dials, a housing that has various shapes and various colors, a receiver to change current into soundwaves, a curly-shaped cord, a line and a transmitter that changes soundwaves into current.	Constituency
The receiver is a loudspeaker with a thin small metal disc-shaped diaphragm. A dialing-mechanism is connected to a wall by the line. The housing is connected to the dialing-mechanism by the cord. The housing contains the transmitter and it contains the receiver. The transmitter is a microphone with a small diaphragm.	Attributive information for the parts
The receiver is a device that changes a varying current into soundwaves. The variation of the varying current causes the field of the magnet to vary. The magnet is permanent and ring-shaped. The field is magnetic. The variation of the field causes the diaphragm to vibrate. The vibration of the diaphragm causes the soundwave intensity to vary. The current varies, like the soundwaves intensity varies.	Process trace for the receiver
The transmitter is a device that changes soundwaves into a varying current. Because a person speaks into the transmitter, soundwaves hit the diaphragm of the transmitter. This causes the diaphragm to vibrate. The vibration of the diaphragm causes the current to vary. The current varies like, the soundwave intensity varies.	Process trace for the transmitter

Figure 6-7: Switching to the process trace for the superordinate and two parts

Figure 6-6, no local expertise about the telephone is indicated in the user model. Therefore TAILOR switches to the process trace to describe it. After the telephone's mechanism has been presented, TAILOR returns to the constituency schema, and the subparts of the pulse telephone are introduced. Now TAILOR has the options of describing both the telephone transmitter and the telephone receiver using the process trace. As the text constructed so far is longer than the threshold allowed and there are two parts to describe, however, TAILOR chooses to continue with the constituency schema. Since a long description was explicitly requested, TAILOR describes these two subparts with the process trace afterwards. Because of the length of the text, substeps are preempted when going through the process trace for the parts.

Another text, shown in Figure 6-8 describes a radio station. In this example, the process trace is chosen for the initial description of the object, and switching to the constituency schema occurs for one of the parts. The user model indicates local expertise about radio transmitters. The process trace is chosen for the overall structure of the text, because the user does not know most of the parts of the radio station. The description starts by introducing the object with its purpose and following the main sequence of events that occur when the object performs its function. When the radio transmitter is introduced as part of the process explanation, TAILOR switches to the constituency schema to describe it, as the user model shows local expertise about radio transmitters. After the radio transmitter is described with its subparts, the process explanation is resumed.

Finally, in the last text shown in Figure 6-9, the same user model as in Figure 6-5 is used for a description of the same object, the telephone, but the value of *most* was changed from Figure 6-5. *Most* now requires *more than half* of the functionally important parts to be known to the user for the constituency schema to be chosen. The description from Figure 6-5 is repeated in Figure 6-10 for ease of comparison.

Radio Station

User Model: Local expertise -- *radio transmitter*
Basic Concepts -- *electricity*

A radio station has a few functionally important parts, only one of which is known to the user. The *process trace* is thus chosen as the overall structure of the text. TAILOR switches to the *constituency schema* for the radio transmitter as the user model indicates local expertise about it.

TAILOR-87 output:

Process Trace

A radio-station is a device that broadcasts a signal when a human speaks into a microphone. Because the human speaks into the microphone, a current is produced. Then, the current flows through the wire into the radio-transmitter. The radio-transmitter has an oscillator that produces a varying current when a battery produces another current, an amplifier that produces a strong varying audio-frequency signal from a varying audio-frequency signal, a mixer, and an amplifier to produce a strong radio-frequency signal from a radio-frequency signal. An amplifier has a vacuum-tube to produce a strong current from a power source and a metal grid. The oscillator has a high-resistance resistor, a battery, a capacitor, a low-resistance resistor and a transistor. The current flowing through the wire into the radio-transmitter causes the strong varying radio-frequency signal to be produced. Then, the strong varying radio-frequency signal flows into the antenna of the radio-station. The antenna is long and made of metal. Because the strong radio-frequency signal flows into the antenna, the strong radio-frequency signal is broadcasted.

Switch to constituency schema for the radio transmitter

Continue the process trace

Figure 6-8: Starting with the process trace and switching to the constituency schema for one part

Telephone

User Model: Local expertise -- *loudspeaker*
Basic Concepts -- *nil*

The value of *Most* was changed to require more than one half of the functionally important subparts to be known to the user before the constituency schema could be chosen. As a result, the process trace was chosen here. The constituency schema is used to describe the receiver because the user model indicates local expertise about the loudspeaker.

TAILOR-87 output:

```
 The telephone is a device that transmits
soundwaves.  Because a person speaks into the
transmitter of the telephone a varying current
is produced.  Then, the varying current flows
through the line into the receiver.  The
receiver has a thin small metal disc-shaped
diaphragm, a gap, a ring-shaped permanent
magnet, a coil and a ring-shaped permendur
armature.  The diaphragm is mounted on the
poles of the magnet.  The gap is between the
poles and the diaphragm and it contains air.
The coil is wound around the magnet.  Because
the varying current flows through the line into
the receiver, soundwaves are reproduced.
```

Process trace
for the telephone

Switch to constituency
schema for the
loudspeaker

Process trace

More about (transmitter)?: `nil`

Figure 6-9: Changing the parameter that determines the overall structure of a description

---

Telephone

The telephone is a device that transmits soundwaves. The telephone has a housing that has various shapes and various colors, a transmitter that changes soundwaves into current, a curly-shaped cord, a line, a receiver to change current into soundwaves and a dialing-mechanism. The transmitter is a microphone. A person speaking into the microphone causes the soundwaves to hit the diaphragm of the microphone. The soundwaves hitting the diaphragm causes the diaphragm to vibrate. The vibration of the diaphragm causes the current to vary. The current varies, like the intensity varies. The receiver is a loudspeaker with a small aluminium diaphragm. The housing contains the transmitter and it contains the receiver. The housing is connected to the dialing-mechanism by the cord. The line connects the dialing-mechanism to the wall.

Figure 6-10: Description of the telephone.
Most is set to half of the functionally important parts

In figure 6-9, the description starts with the process trace and switches to the constituency schema for the loudspeaker. In this text, a process mechanism is provided for the telephone, explaining how its main parts work together to allow the telephone to transmit soundwaves. As the user is already familiar with the loudspeaker, a fuller structural description is provided for the receiver, an instance of a loudspeaker. This text would be appropriate for a user who would not be able to infer how the telephone functioned knowing only how the loudspeaker functions.

Compare this text with the one in Figure 6-10. In this text, only a structural description is provided for the telephone. A process explanation is provided for the transmitter, however, because this part is unknown to the user. This text would be appropriate when the user is able to infer how the telephone's mechanism, knowing how the receiver works and given the process explanation for the transmitter. It is unclear which of these texts is "best" for a user who has local expertise about loudspeakers. This is an indication that testing is needed to set the value of *most*. It is likely, however, that no unique value actually exists: it might depend on the kind of object being described, the user's learning style, and other factors.

In either case the mixture of functional and structural information seems appropriate given the user's domain knowledge. These two texts are included to illustrate how, although *most* chooses the overall structure of the description, a mixture of process and structural information is still provided.

The texts presented in this section seem to be appropriate for the users they are tailored to as they provide functional details when the user lacks such information and structural details when the user probably already has the functional understanding of the object. The heuristics provided here allow the system to generate reasonable descriptions given the user's domain knowledge. Note, however, that as is generally the case in text generation, there are no clear "best" texts, and extensive testing would be required to determine the descriptions effectiveness.

6.5 Combining strategies yields a greater variety of texts

Independent of whether the texts provided here are the most appropriate for a given user, the ability to combine strategies is important, because it allows TAILOR to generate a greater variety of texts than would otherwise be possible. For example, for the pulse telephone which has a superordinate and three functionally important parts, TAILOR can generate eight different descriptions, even though the constituency schema is always chosen as the initial strategy (as the pulse telephone has no mechanism in its frame). These various descriptions are outlined below with the corresponding user model:

- The user model is nil: the process trace is chosen for the telephone and each of the parts.
- The user model indicates local expertise about the telephone: the process trace is chosen for the pulse dialer, the only part that differs from the telephone.
- The user model indicates local expertise about the receiver: the process trace is chosen for both the transmitter and pulse dialer.
- The user model indicates local expertise about the transmitter: the process trace is chosen for both the receiver and pulse dialer.
- The user model indicates local expertise about the pulse dialer: the process trace is chosen for the telephone.
- The user model indicates local expertise about the telephone and the pulse dialer: the constituency schema is chosen for the whole description.
- The user model indicates local expertise about the receiver and the pulse dialer: the process trace is chosen for the transmitter.
- The user model indicates local expertise about the transmitter and the pulse dialer: the process trace is chosen for the receiver.

Note that this number takes into account the fact that the heuristics chosen in TAILOR limit the number of combinations allowed. Furthermore, knowledge about basic concepts plays no role for this particular object. If it did, there would be yet more combinations.

The ability to combine strategies allows a generation system to have distinct strategies clearly defined without losing flexibility. Discourse strategies help a system to construct a text with a given structure, and it is desirable to have such strategies to constrain the generation process. Given the strategies and no mechanism for combining them, a system is limited in the kinds of texts it can generate by the number of strategies. If combining strategies can be achieved, the variety of texts that can be generated is greatly increased.

Because, in TAILOR, each of the two strategies is represented by an ATN, they can be easily combined. The arcs of the ATN indicate how to go from one node to another. Jumping from a node in one network to a node in another network is as easy as going from one node to another in the same network. As a result, the control strategy needed for switching strategies is readily available with the ATN formalism and should be taken advantage of.

To combine the strategies, the potential decision points and entry points have to be decided upon. ''Jump <entry-point>'' arcs between the two strategies can be added at the decision points. Tests on the arcs can control the combinations. These tests can incorporate a number of factors, including the information contained in the knowledge base, the discourse so far, and a user model.

In the TAILOR descriptions shown so far, the decision points in the strategies and the tests combining the strategies were set to allow switching only when a new object was introduced and needed to be described. The entry points into the strategies were always at the beginning, because switching was done to describe a new object. This does not have to be the case in general. For example, consider the text in Figure 6-11 describing a telephone.

In this text, both structural and functional information are used to describe only the telephone (not its subparts). The constituency schema was chosen at first, and TAILOR jumped to the process trace after providing the parts of the telephone, to include a process trace for the telephone as well. TAILOR does not return to the constituency schema after the process trace is given, and the process trace is entered after the introductory statement, to start the process trace immediately. Similarly, in Figure 6-12, a strategy switch occurs for the same object at an entry point other than the beginning. These two texts were generated by modifying the ATN used by TAILOR to show the feasibility of mixing strategies in ways other than the ones already defined. To produce these texts, only the tests at the decision points in the strategy were changed.

Telephone

This description starts with the constituency schema and switches to the process trace, for the telephone itself, after providing the subparts of the telephone.

TAILOR-87 output:

```
 The   telephone   is   a   device   that   transmits    Identification
soundwaves.  The telephone has a housing that
has   various   shapes   and   various   colors,   a     Constituency
transmitter   that   changes   soundwaves   into
current,   a   curly-shaped   cord,   a   line,   a
receiver to change current into soundwaves and
a dialing_mechanism.  When a human speaks into
the  transmitter  of  the  telephone,  a  varying       Switch to process
current is produced.  Then, the varying current         trace for the
flows through the line into the receiver of the         telephone itself
telephone.    This   causes   soundwaves   to   be
reproduced.
```

Figure 6-11: Combining the strategies

Combining strategies is by no means limited to two strategies as in TAILOR. A generation system having a number of discourse strategies could combine them in a variety of ways. Once the entry points, decision points and tests are chosen, jump arcs between the various subnets can be added. This strategies yields more flexibility and power in the kinds of texts a system can generate. (In designing the tests, however, one would have to make sure that coherence is not hindered when the system switches from one strategy to another one.)

The TEXT system was also able to combine several strategies with a recursion mechanism [McKeown 85]. McKeown pointed out that the predicates could be

Telephone

This description starts with the process trace and switches to the constituency schema.

TAILOR-87 output:

```
The telephone is a device that transmits         Process trace
soundwaves. When a human speaks into the
transmitter of the telephone, a varying current
is produced. Then, the varying current flows
through the line into the receiver of the
telephone. This causes soundwaves to be
reproduced. The transmitter is a microphone      Switch to constituency
with a small disc-shaped metal thin diaphragm.   Attributive information
The line is a wire. The receiver is a            about the subparts
loudspeaker with a small disc-shaped metal thin
diaphragm.
```

Figure 6-12: Combining the strategies, using an entry point other than the beginning

expanded into schemas. As a result, instead of simply matching a predicate against the knowledge base, the corresponding schema could be traversed recursively. This was not fully implemented in TEXT, however, and the test on recursion was based only on the predicate type and the amount of information contained in the knowledge base. TAILOR is also able to use the strategies recursively, in a similar way to TEXT, except that the test for recursion also involves the user model. This was not a major concern in this work, however.

Expanding on a predicate using recursion is only one of the ways to combine strategies. Combining them as explained above allows mixing strategies of different types in a variety of ways, since entry points are not necessarily limited to the beginning of the strategy and control does not have to return to the first strategy. This can only add flexibility to a generation system.

6.6 Conclusions

This chapter showed how TAILOR can automatically combine the two strategies presented earlier to provide descriptions to users with intermediate levels of expertise. By representing explicitly the user's domain knowledge in terms of parameters, TAILOR does not require an *a priori* set of stereotypes but can provide a wide variety of descriptions for a whole range of users between the extremes of naive and expert.

I showed how it is easy to combine discourse strategies when they are represented using the same formalism, an augmented transition network. Combining strategies gives a generation system more flexibility and allows it to widen the range of texts it can generate.

7. TAILOR system implementation

7.1 Introduction

I have implemented the discourse strategies presented in previous chapters in TAILOR, a program that generates descriptions tailored to a user's level of expertise. The discourse strategies guide the program to choose the appropriate information from the knowledge base, which is in the RESEARCHER's format. TAILOR looks at the information contained in the user model to decide on the strategy to employ. The strategy implementation as well as other components of TAILOR are discussed in this chapter.

TAILOR is implemented in Portable Standard Lisp (PSL) and runs both on an HP 9836 Workstation and an IBM 4381 under VM/CMS. Because this work was being conducted as the same time as the research for RESEARCHER's parser, the knowledge base for TAILOR was built by hand. However, it is faithful to the representation that would be built by RESEARCHER. It contains information about a number complex devices such as telephones, radio transmitters and amplifiers.[22]

7.2 System overview

TAILOR is the generation component of a question-answering system for RESEARCHER, as shown in Figure 7-1. Requests for descriptions are given in the following form:

Describe X

---

[22] The information contained in the knowledge base reflects that contained in the texts read from the encyclopedias and other sources. Simplifications that have been made in the texts are thus also made in the knowledge base. This work is more concerned with retrieving information from a knowledge base than with representing physical devices in the most accurate way.

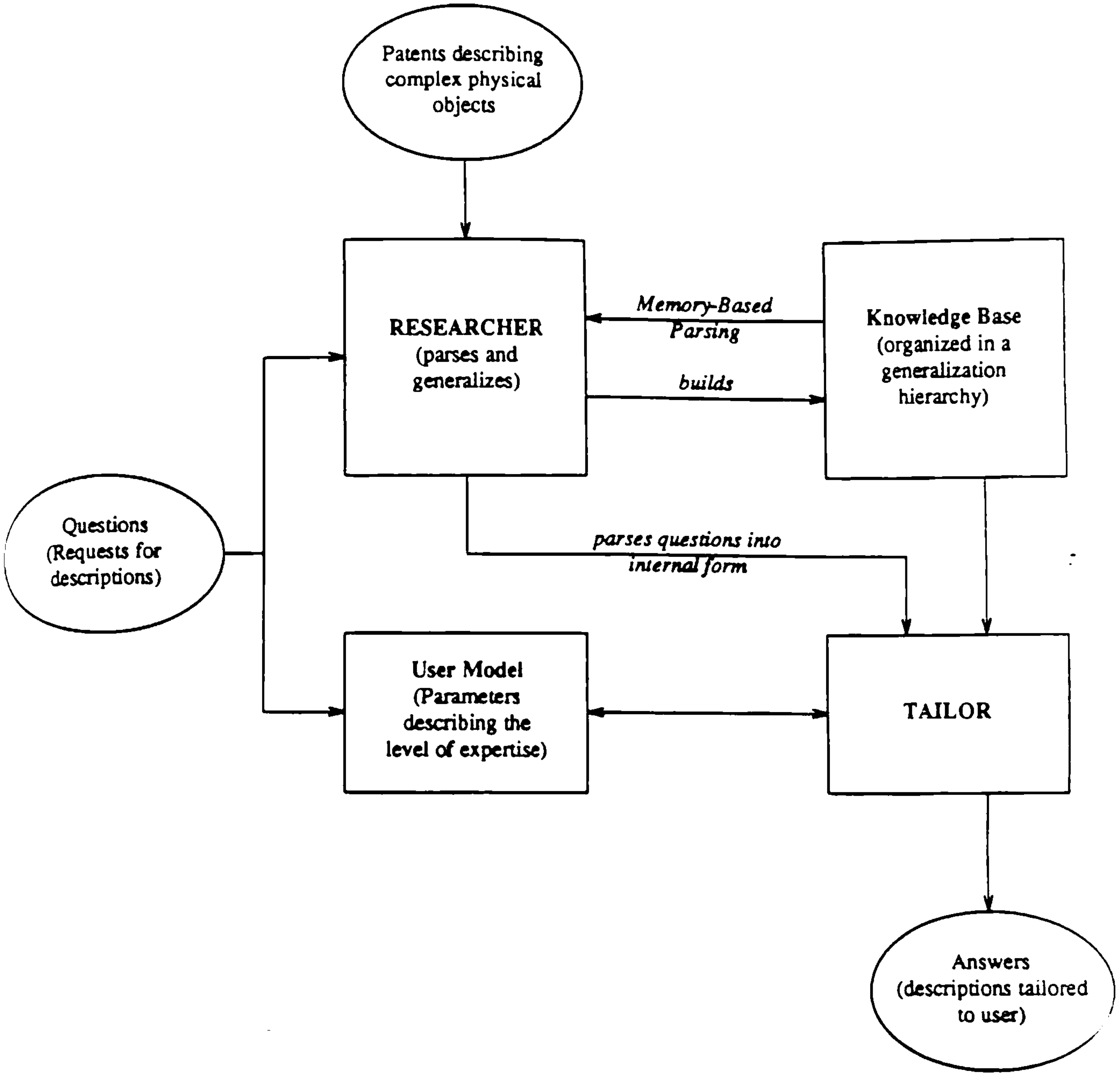

Patents describing complex physical objects
RESEARCHER (parses and generalizes)
Memory-Based Parsing
builds
Knowledge Base (organized in a generalization hierarchy)
Questions (Requests for descriptions)
parses questions into internal form
User Model (Parameters describing the level of expertise)
TAILOR
Answers (descriptions tailored to user)

Figure 7-1: RESEARCHER and the TAILOR System

where X is an object contained in the knowledge base. In the ideal system, RESEARCHER would parse questions using the same parser as the one it employs for parsing patent abstracts, and would then hand a representation of the question to the question-answering system. TAILOR's task is to decide how to answer the question.

7.3 System overview

A block diagram of TAILOR is shown in Figure 7-2. TAILOR is divided into three components:

1- The *textual component* determines the content and organization of the description to be generated. This is the main emphasis of the work. The textual component examines the knowledge base and chooses appropriate facts based on the level of expertise of the user and a discourse strategy. The output of this component is a conceptual representation of the content of the description. The textual component will be described in section 7.6.

2- The *dictionary interface* takes the conceptual representation produced by the textual component and chooses the syntactic structure of each proposition. It also assigns lexical items for the various concepts contained in the description. This process will be described in section 7.7. The output of the interface is a deep structure representation of the sentences to be generated. Because the emphasis of this work has been on the textual component, the complexity and subtleties of lexical choice have not been studied in depth.

3- The *surface generator* takes the output of the interface and constructs English sentences. The surface generator used by TAILOR is based on the one used by the TEXT system [McKeown 85]. The generator unifies the input with a functional grammar [Kay 79] to produce English sentences. The original functional grammar

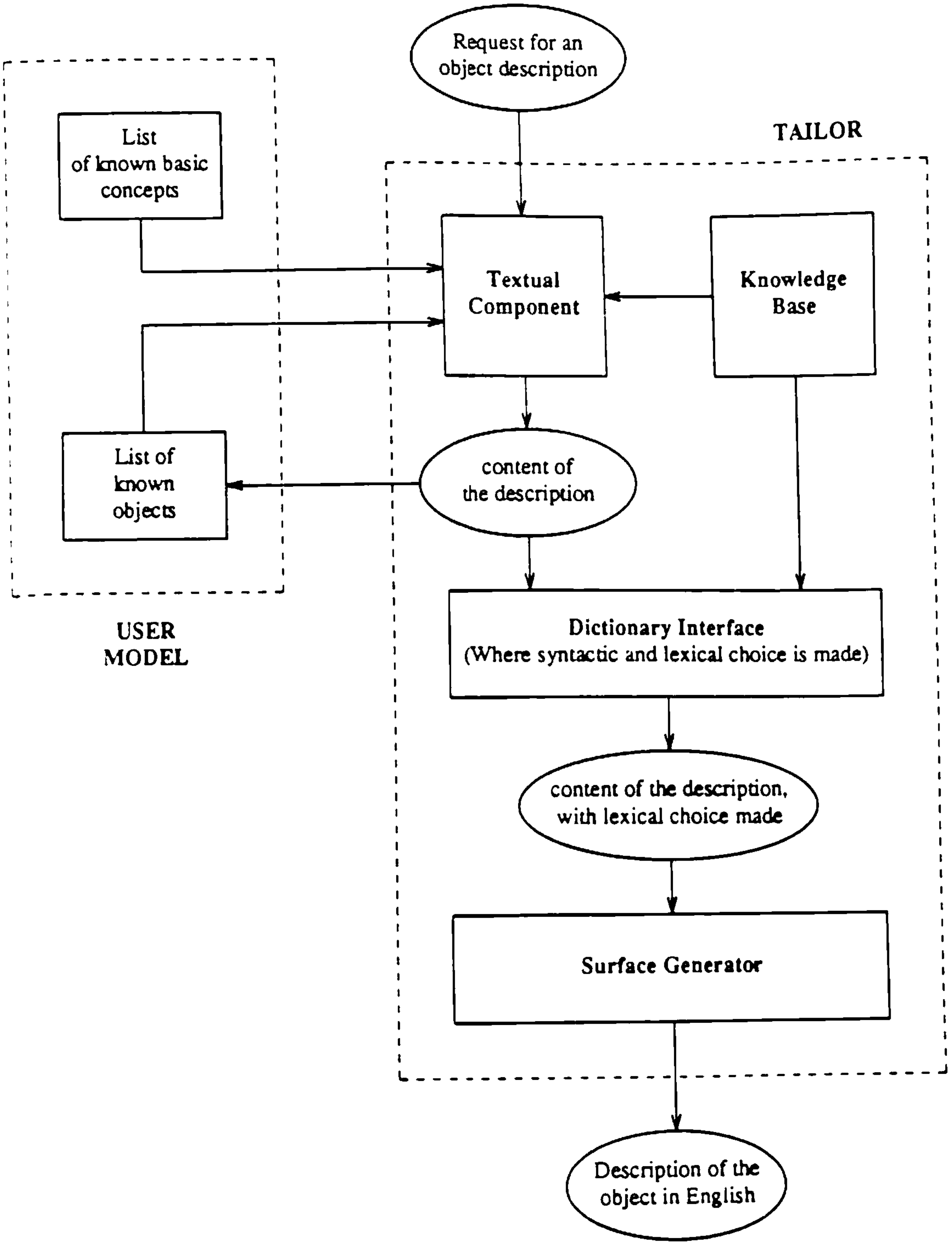

Request for an object description
TAILOR
List of known basic concepts
Textual Component
Knowledge Base
List of known objects
content of the description
USER MODEL
Dictionary Interface (Where syntactic and lexical choice is made)
content of the description, with lexical choice made
Surface Generator
Description of the object in English

Figure 7-2: The TAILOR System

has been extended, and the performance of the system improved. The functional grammar and the improvements to the original system will be discussed in section 7.8.

The remainder of this chapter describes the implementation of the components of TAILOR. Before describing each component of the system, the content and structure of the knowledge base will be presented in Section 7.4, as both affect the kinds of descriptions that can be generated by TAILOR. Furthermore, since the user model plays an important role in TAILOR, it will be discussed in Section 7.5, before the components of the system. The user model in TAILOR contains parameters representing the user's assumed domain knowledge.

7.4 The knowledge base and its representation

Without an inference engine, any generation system is constrained by the knowledge base it uses, as it cannot include in a text information that is not explicitly contained in the knowledge base. Because the emphasis of this work was on generation, not on knowledge representation, and since knowledge representation issues had already been addressed as part of the RESEARCHER project in [Wasserman and Lebowitz 83; Wasserman 85], I decided to use the knowledge representation as previously developed. Nevertheless, as is often the case, the development of the generation system resulted in a few modifications of the knowledge base representation, mainly in the representation of process information.

The knowledge base contains detailed descriptions of complex devices organized in generalization hierarchies. The knowledge base contains about 120 object frames and 150 frames of other types. Simplified diagrams of parts of the knowledge base are shown in Figure 7-3 and 7-4. Figure 7-3 shows a subset of the parts hierarchies, while Figure 7-4 presents a subset of the generalization hierarchies.

The knowledge base include three kinds of information about the objects:

- *structural information* indicating both what the components of an object are and how objects are spatially related to each other.
- *attributive information*, that is, properties associated with the objects, such as color and shapes.
- *functional information* showing how objects achieve their function.

These three kinds of information restrict the kinds of descriptions TAILOR can generate. For example, TAILOR cannot provide a description that includes an historical development of a device, or a cost/performance evaluation, since that information is not included in the knowledge base.

The knowledge base representation employed here is a framed-based representation, where the basic frames represent entities. Entities include the physical devices that are being represented, such as "disc-drive" and "magnet" and more abstract concepts such as "current" and "side." Other types of frames are used to represent events and their relationships (such as causal relationships). Each frame type is reviewed in turn. (See [Wasserman and Lebowitz 83; Wasserman 85] for more details about the representation.)

The basic structure of an entity is shown in Figure 7-5. The *type* slot of the frame indicates whether the object can be decomposed in terms of other parts or is a single indivisible structure. For example, a "disc-drive" is composite, as it comprises several other parts, while a "diaphragm" is unitary.

If the object is unitary, a shape descriptor that includes properties about the object is included in the *structure* slot. If the object is composite, the frame includes a list of the object subparts (a list of pointers to the subparts) in the *components* slot, and the

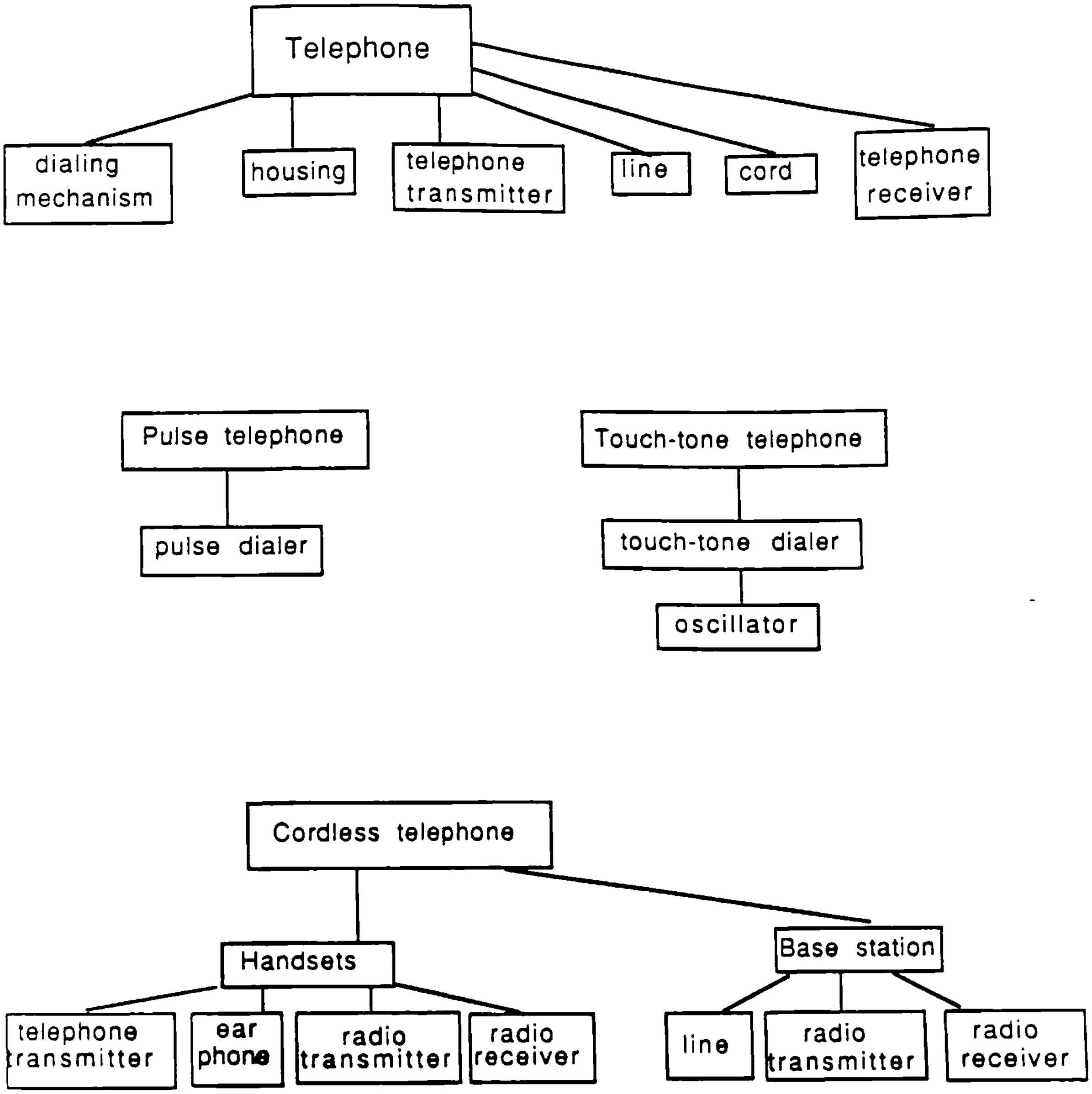


Figure 7-3: The knowledge base used in TAILOR; parts hierarchies

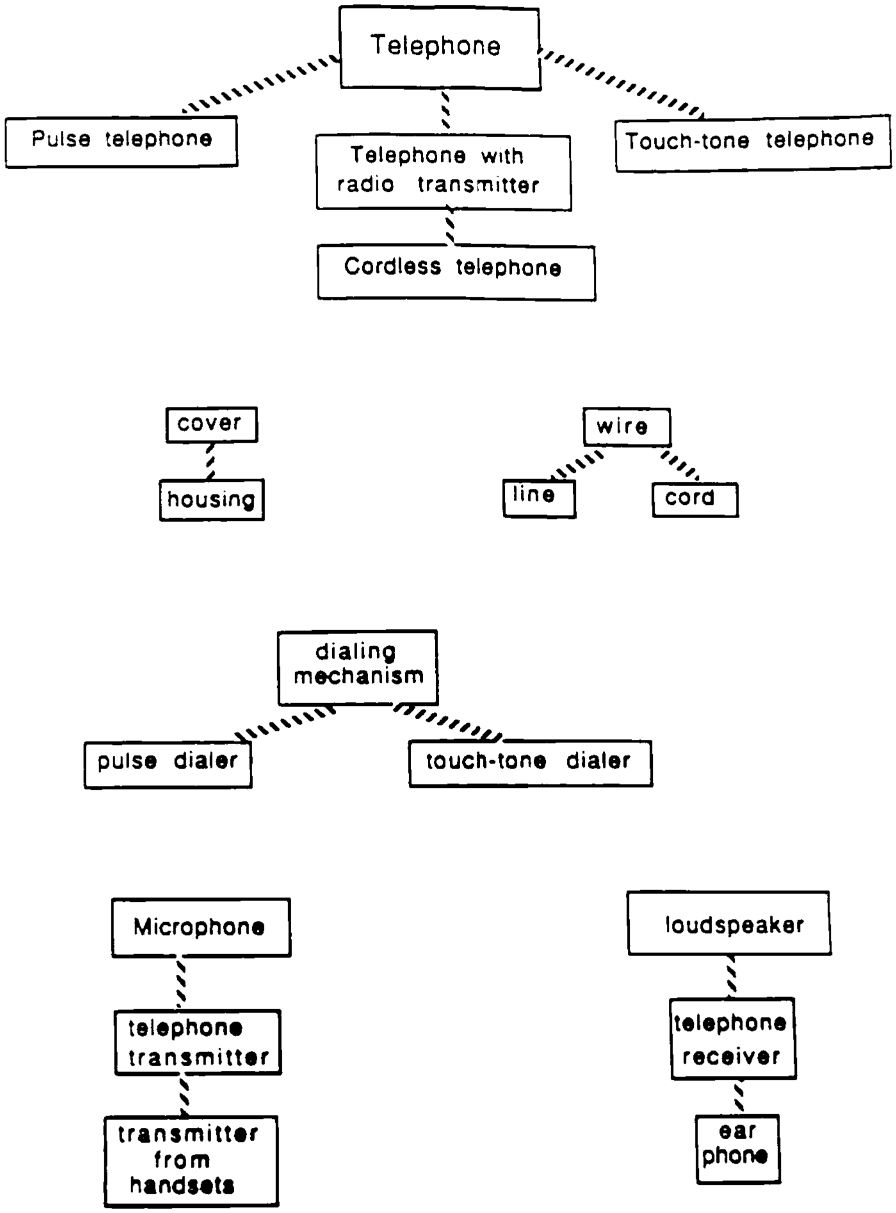


Figure 7-4: The knowledge base used in TAILOR; generalization hierarchies

Name: name of object
Type: ``unitary'' or ``composite''
Structure: a shape-descriptor if unitary;
a list of relation records if composite.
Properties: a list of properties and their respective values.
Components: a list of the object parts (pointers to the frames)
Purpose: a relation that represents the object's function in terms of an *input* and an *output*, representing the start and goal state for the process description.
Events-in: a list of events the object participates in.

Figure 7-5: An object frame

structure slot includes a list of relation records that indicate how the parts are spatially related to each other. Each relation record represents a static physical relation between two objects and is also represented by a frame. A compositional primitive based scheme has been developed to support the wide variety of physical relations that exist. A complete description of the representation scheme for the physical relations can be found in [Wasserman and Lebowitz 83].

The *properties* slot of the frame contains features of the device, such as its color or material. The *purpose* slot of a device frame is used to represent the *function* of the object. This function is expressed as a relation between two entities or possibly two other relations [Baker and Danyluk 86]. As an example, the microphone's function is to transform ``soundwaves'' into a ``varying current,'' as shown in Figure 7-6. The microphone purpose slot thus contains the *relation frame* corresponding to the event *the soundwaves are transformed into current.* In the figure, this relation frame is represented with its unique identifier, *&P-REL7.* In all the figures included in this chapter, the English noun-phrases in brackets are provided only for clarity and do not appear in the representation.

The decision to explicitly represent a device's function as a relation between an input and an output was made partially because of the necessity to be able to retrieve the start and goal states of a device's mechanism to be able to generate a process explanation. With this representation, the goal and start state of the main sequence of events can be easily obtained. As a relation frame, in turn, contains information about causal relationships it participates in, the causal links of an object's mechanism can be retrieved through the object's purpose slot. This will be explained in more detail in section 7.6.2.

Name: `MICROPHONE`
Purpose: `(&P-REL7 [P-TRANSFORMS SOUNDWAVES -> CURRENT])`

Figure 7-6: Representation of a microphone's function

The last slot in an object frame is the *events-in* slot. This slot indicates which actions, or events, the object participates in. For example, if the spindle of the disc-drive rotates, the event corresponding to this rotation would be included in the events-in slot of the spindle. The original representation scheme developed for RESEARCHER had concentrated on representing static spatial relations, such as *on-top-of* or *inside-of*. Because of the importance of dynamic relations (such as *rotation*) to describe the process information, the original scheme was extended to allow for dynamic relations, or events. These events were classified into several categories, and a set of primitive features similar to those used for spatial static relations is used to describe events in each category.

Relationships among functional events are represented by links between the event frames (or relation frames), called *event-links.* These links represent *control, temporal,* or *correspondence* relations, as discussed in Chapter 5. Control relations include: cause-effect, enablement, control, limiting, preventing, interrupting and terminating. Temporal relations indicate whether an event happens before, after, or at the same time as an another event. Correspondence relations are used when an event is proportional or equivalent to another one.

An event link is represented as a frame which contains a slot for the link type (i.e., control, temporal, or correspondence), and slots that indicate which events are related by the link. As a convention, these events are considered to be the *subject* and the *object* of the link. For example, that control link corresponding to ''<state X> causes <state Y>'' would have <state X> as its subject and <state Y> as its object. Both <state X> and <state Y> would also have a pointer back to the event-link. Thus each event frame also contains a slot that indicates the event-links in which this particular event participates. An example of this representation is shown in Figure 7-7. The top frame represents a causal link between two events. Its subject and object slots contain a pointer to the appropriate relation record. These relation records are shown underneath. Each has a slot indicating that they take a part in the causal link.

7.4.1 The generalization hierarchies

Entities in the knowledge base are organized in generalization hierarchies, thus defining prototypes: when two objects are similar, RESEARCHER extracts their identical features to form a generalization prototype [Wasserman 85]. This allows for a more compact representation, as redundant information is stored only once. Two additional slots in an object's frame indicate whether the object is an instance of a more generic class of objects (*variant-of* slot) and whether the object itself is a generalization (*variants* slot). (These slots were not shown in Figure 7-5.)

```
                  Link between two events:

                  NAME          = M-CAUSES [CONTROL]
                  SUBJECT       = &REL4
                  OBJECT        = &REL22
                  REC-TYPE      = EVENT-LINK
                  ID            = &MR2


 Subject Event: &REL4                 Object Event: &REL22

CLASS          = PURPOSE              CLASS          = PURPOSE
SUBJECT        = &MEM8 (DIAPHRAGM)    SUBJECT        = &MEM19 (CURRENT)
REL-FRAME      = P-VIBRATES           REL-FRAME      = P-VARIES
EVENT-LINKS-IN = (&MR2)               EVENT-LINKS-IN = (&MR2)
SUBSTEPS       = (&REL6, &REL5)       SUBSTEPS       = (&REL6, &REL5)
                 (moves forward, backward)             (increases, decreases)
```

Figure 7-7: Representation of events and links between events

As an example, Figure 7-8 shows the representation of the ''microphone,'' which has a variant, namely the ''telephone-transmitter,'' and which is a variant-of a ''device.'' Because the microphone has subparts, its type is ''composite,'' and its structure slot includes relation records. &RELn are the unique identifiers of the relation records. For example, &REL1 in the figure indicates that ''the diaphragm is clamped.'' &REL1 is an instance of the relation frame that corresponds to the relation ''clamp.'' Similarly, &P-REL7 is a relation record for the dynamic relation ''transform.'' It is used here to indicate the function of the microphone. &MEMn are the unique identifiers for object frames. For instance, &MEM36 is the identifier of the ''transmitter'' frame.

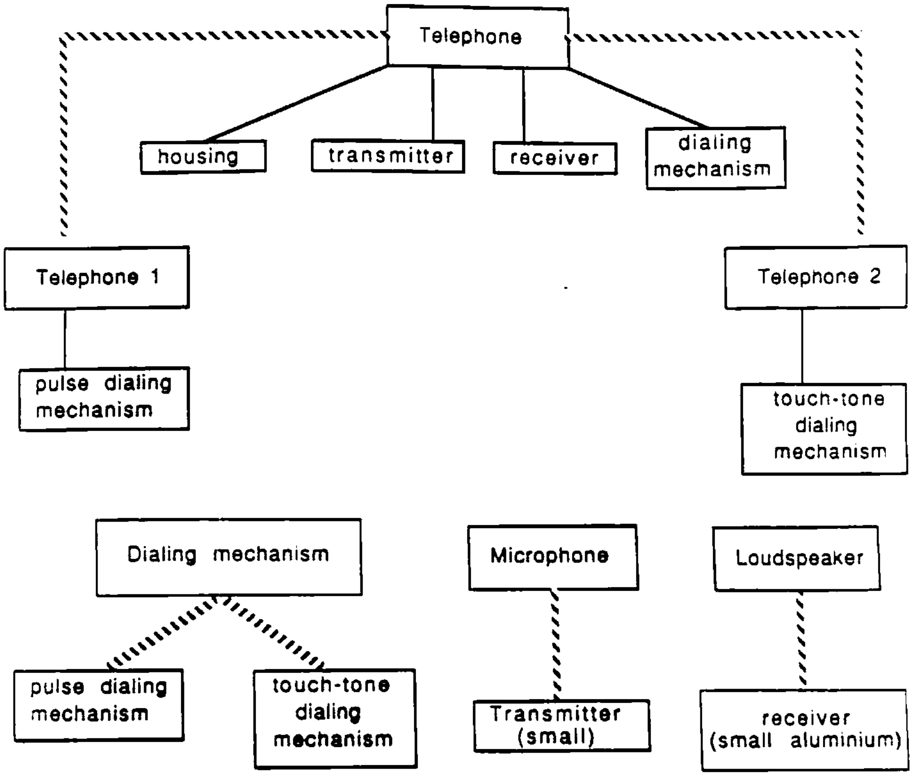
```
Name:        MICROPHONE
Type:        COMPOSITE
Structure:   (&REL1 [DIAPHRAGM CLAMPED]
Components:  (&MEM2 [RESONANT-SYSTEM], &MEM2 [DIAPHRAGM])
Purpose:     (&P-REL7 [P-TRANSFORMS SOUNDWAVES -> CURRENT])
Variants:    (&MEM36 [TELEPHONE-TRANSMITTER])
Variant-of:  (DEVICE)
```

Figure 7-8: Representation of a microphone

Each different object can be part of a generalization hierarchy. As a result, the knowledge base contains numerous generalization trees. This is illustrated in Figure 7-9, where the main object, the telephone, is a member of one generalization tree while its subparts are members of different one. In this figure, the solid lines represent the parts hierarchies and the broken ones the generalization hierarchies.

Telephone 1 has four parts: a housing, a cord and a wire, a transmitter, a receiver, and a pulse dialing mechanism. Telephone 2 is very similar, except that its dialing mechanism is a touch-tone dialing mechanism. These two telephones are represented as instances of the prototypical frame *telephone*. This frame has four parts: a housing, a telephone transmitter, and telephone receiver, and a generalization of the dialing mechanism. Furthermore, the transmitter and the receiver were already members of some generalization tree. Issues of creating a generalization hierarchy have been described at length in [Wasserman 85].

Each object frame contains information about how it differs from its parent. For



Telephone:
Variants: telephone 1, telephone 2
Components: housing, transmitter, receiver, dialing mech.

Telephone 1:
Variant-of: telephone
Components: pulse dialing mechanism

Telephone 2:
Variant-of: telephone
Components: touch-tone dialing mechanism

Dialing-Mechanism:
Variants: pulse dialing mech., touch-tone mech.

Pulse dialing mechanism:
Variant-of: dialing mechanism

Touch-tone dialing mechanism:
Variant-of: dialing mechanism

Transmitter:
Variant-of: microphone
Specialization: (small)

Receiver:
Variant-of: loudspeaker
Specialization: (small, aluminium)

Figure 7-9: Several generalization trees

example, the telephone transmitter is a microphone with a small diaphragm. The frame corresponding to the microphone contains the small diaphragm as a component, and also indicates that this part is to be substituted to the microphone's diaphragm whenever necessary. Similarly, when an object has a purpose different from its parent's, it is indicated in the object frame.

7.4.2 Limitations of the knowledge base

As pointed out at the beginning of this section, a knowledge base restricts the kinds of texts that could be generated. Two limitations are important to mention here.

There is no mechanism to explicitly represent functional comparisons and analogies. For example, it is not possible given the knowledge base and without an inference mechanism to determine the fact that a loudspeaker is a microphone in reverse. As a result, given a user model indicating expertise about microphone, TAILOR will not be able to describe the loudspeaker in terms of the microphone. Instead, TAILOR would consider the user a naive user with respect to loudspeaker, and produce the same text as if the user had no local expertise about microphones.

The other limitation is the inability to represent general laws of physics, such as *gravity* or *Newton's laws*. Therefore, TAILOR cannot describe the function of an object by simply mentioning the general law by which it can be explained. Adding the ability to represent such concepts would allow for the generation of more types of descriptions and would also add another factor to the tailoring, as it would be possible to explain a device mechanism either by explaining the sequence of events that take place (as is done in TAILOR) or by explaining the general mechanism involved.

7.5 The user model

Already presented in Chapter 3, the user model is characterized in Figure 7-10. It contains two parameters indicating whether the user has two types of knowledge.

Local expertise about specific objects:	*Pointers into the knowledge base*
Knowledge about basic concepts:	*A list of concepts*

Figure 7-10: A characterization of the User Model in TAILOR

The *local expertise* parameter indicates about which specific objects the user is knowledgeable. This parameter is represented by pointers into the knowledge base to avoid duplicating a portion of the knowledge base in the user model as done in [Hoeppner *et al.* 84]. Knowledge about basic concepts is indicated by list of concepts. By using this explicit representation, it is possible to set the parameters to any value for users inside the knowledge spectrum. Examples of user models are shown in Figure 7-11.

A *naive* user is one whose user model is empty, as shown in (c) in Figure 7-11. While the stereotype *novice* can be retained as a shorthand for users falling at that extreme of the continuum, the corresponding user model can still be represented explicitly. The user model for an expert user is given in (d) in the figure, where *expert* is used instead of a huge list of objects.

Notice that the user model is coarse grained, in that it contains a list of objects the user knows and whether he or she understands the basic underlying concepts. A more

(a) The user has local expertise about microphone and understands two of the basic concepts:

Local expertise: *&mem1 [microphone]*
Basic concepts: *(magnetism; voltage)*

(b) The user has local expertise about telephones and radio transmitter and understands one of the basic concepts:

Local expertise: *&mem46 [telephones],*
&mem85 [radio-transmitter]
Basic concepts: *(voltage)*

(c) The user is a naive user:

Local expertise: *nil*
Basic concepts: *nil*

(d) The user is an expert user:

Local expertise: *expert*
Basic concepts: *all*

Figure 7-11: More examples of user models in TAILOR

detailed model might include exactly which facts the user knows about objects, and how much the user understands the basic concepts. This representation was chosen because I feel that a more detailed user model would be much harder to obtain.

7.6 The textual component

TAILOR's textual component decides on both the content and the organization of the description to be generated. Upon receiving a request for a description, TAILOR decides whether to include chiefly structural information, using the constituency schema, or functional information, using the process strategy, or a mixture of both. This decision is not based on the question type since both strategies can be used to provide an answer to a request for a description. It is instead based on the content of the knowledge base and the user's expertise level. The discourse strategies presented in earlier chapters guide the generation process by selecting information from the knowledge base. The strategies also impose order on the chosen information, thus providing the text organization.

7.6.1 Initially selecting a strategy

The first step in generating a description is to select the discourse strategy that will determine the overall content and organization of the text. TAILOR can initially choose one of two discourse strategies to provide a description. This initial choice is based on the content of the knowledge base and the user model. The decision algorithm was presented in Chapter 6 and is only summarized here, with a few supplementary implementation details.

This algorithm is shown in Figure 7-12. In the first step, TAILOR examines the knowledge base to check whether the object to be described has a mechanism associated with it, in which case TAILOR has the choice of describing the device using either strategy. This is done by checking the value of the *purpose* slot (which indicates the function) in the device's frame. If the object frame indicates a function, the decision is based on the user model as described in the previous chapter and shown in the figure.

```
1) Is there a mechanism associated with the object to be described?
      i.e., Check the purpose slot

      No:  Use the constituency schema
      Yes: 2) Is the object to be described (or its superordinate) in the user model?
             (i.e., does the user have local expertise about this object
             or its superordinate?)

             Yes: Use constituency schema
             No:  3) Collect all the functional parts of the object.
                  If the user has local expertise about most of these parts,
                       use the constituency schema
                       Else use process trace.
```

Figure 7-12: The decision algorithm

7.6.2 Finding the main path

The main path is used to generate a process explanation. In order to find the main path, it is important to recognize that links between events play different roles (i.e., are of different types), as, when tracing the knowledge base, the program needs to choose among several such links. One way to achieve this is to assign rankings. Then, based on this ranking, a choice can be made. There is thus a need for an importance ranking indicating for each link type its relative importance in order to produce a process explanation. The ranking used in TAILOR is shown in Figure 7-13. This scale was selected because, based on texts, it appears that control links are the most important links to mention when providing a process explanation. If no control link can be found between two events, a temporal one is the next best choice.

I chose to represent all the rankings as parameters, as opposed to embedding the importance factors in a procedure, to be able to easily change the scales. In a domain with different scales, the parameter can be changed to reflect these differences in link importance.

<control links; temporal links; correspondence links>

Figure 7-13: Importance scale used to find the main path

Within each link type, there is also an importance factor, in order to decide among several links of the same type. In TAILOR's knowledge base, control relations are divided into two groups:

- the *positive control relations*, which include (from highest to lowest ranking) cause-effect, enablement and control.
- the *negative control relations*, which include limiting, preventing, interrupting and terminating.

Temporal relations are also ranked: first the precede relation, then the relation which represents same-time-as. Figure 7-14 summarizes the different link types and their respective importance rankings.

To find the main path, the program must first find the start and goal states of the main path. These are indicated in the function (purpose) slot of the object, as indicated in Section 7.4. The main path is found by searching through the links between events, going from the goal state to the start state. Starting from the goal state, the program looks at all the links linking other states to the one currently under consideration, performing an ordered depth-first search. All of these links are included in the events-in slot of the frame being considered. When there are several links to choose from, TAILOR picks one based on their importance ranking, and now considers the other event as its current state. In this manner, the program goes

---

Control relations: <Positive control relations; Negative control relations>

Positive control relations: <cause-effect enablement control>
Negative control relations: <limiting preventing interrupting terminating>

Temporal relations:

<precede same-time-as>

Correspondence relations:

equivalent-to/proportional (no ranking)

Figure 7-14: Links between events

---

through the links that connects various events, until it reaches the start state, backtracking if necessary.

To choose among different links, the program checks the importance ranking of the link types according to the importance scale given in Figure 7-13. The importance scale indicates which link types are most likely to lead to the start state. If several links are in the same importance category, the individual scales (also given previously) within that category are checked and a link is chosen. As an example, within the control category, a causal link is preferred over an enablement link.

An example of this procedure is shown in Figure 7-15, where the main path for the loudspeaker is searched for. The events that are being considered while searching for the main path are shown in italics, while the event-links are in bold face. The portion of the knowledge base corresponding to the loudspeaker is shown in Figure 7-16, with the various event-link types identified.

Purpose slot: *changes current into soundwaves*

The goal and start states are taken from the purpose slot:

Goal state: *Soundwaves-intensity varies*
Start state: *current varies*

Searching for the main path starting from the goal:

Soundwaves-intensity varies: participates in two event-links:
  diaphragm vibrates **causes** *soundwaves-intensity varies*
  soundwaves-intensity varies **corresponds to** *current varies*

cause is chosen over **corresponds to** based on the importance link;

The search continues with the event *diaphragm vibrates.*

diaphragm vibrates: participates in one event-link:
  field varies **causes** *diaphragm vibrates*

The search continues with the event *field varies.*

field varies: participates in one event-link:
  current varies **causes** *field varies.*

The search continues with the event *current varies.* This is the start, the search ends. The main path is:

  current varies **causes** *field varies.*
  field varies **causes** *diaphragm vibrates.*
  diaphragm vibrates **causes** *soundwaves-intensity varies.*

Figure 7-15: Finding the main path for the loudspeaker

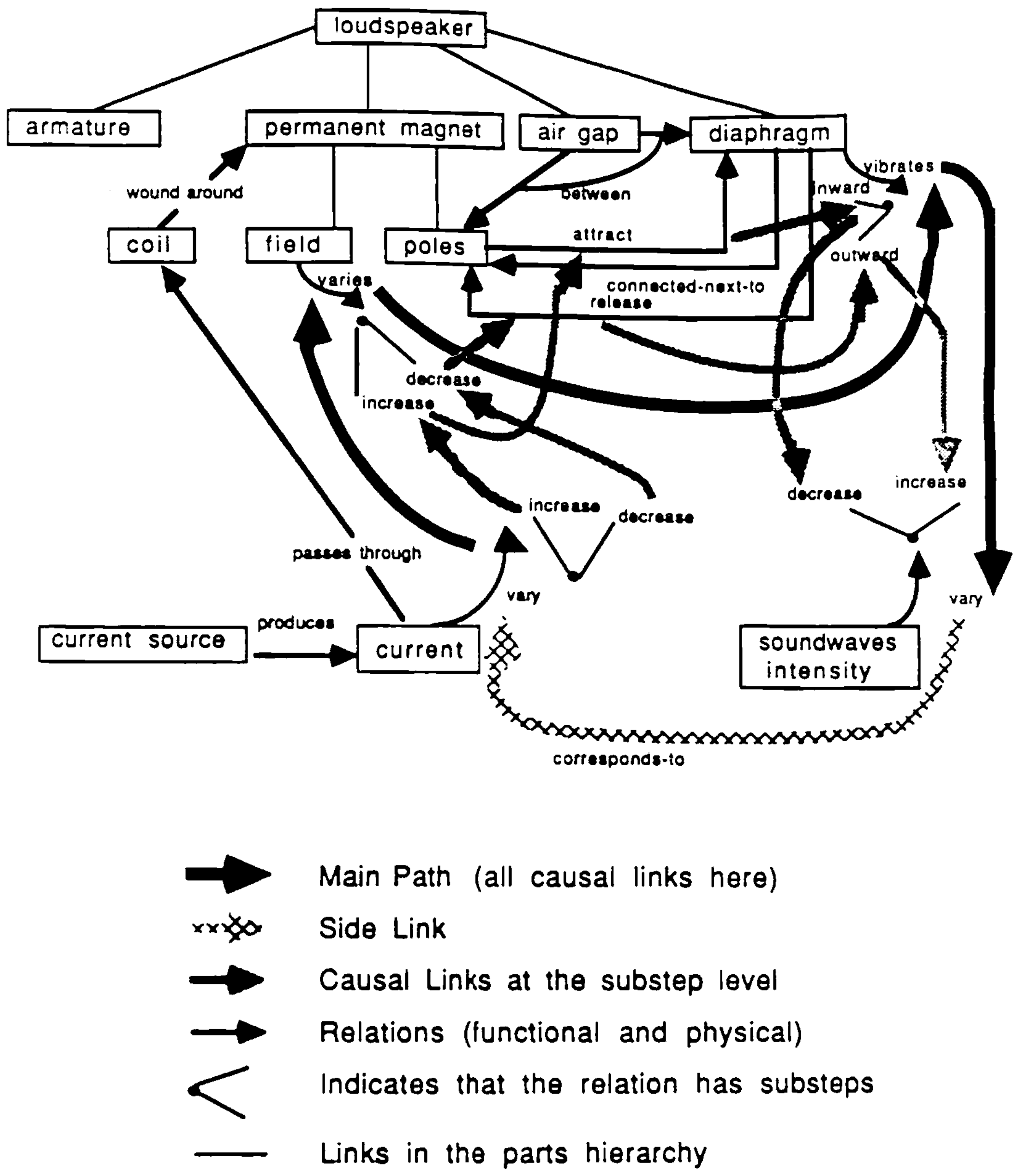


Figure 7-16: Knowledge base for the loudspeaker

I decided to search for the main path by starting from the goal state as I thought that this would involve less backtracking than if the program were to go from the start state, as the program would not try to trace through side effects. Furthermore, while the goal state is always known (i.e., since the goal state represents the output of the device), the start state is not always clear: there might be several states which could be considered the start state. So by starting from the goal state and moving towards the start state, the program can stop when the desired start state is reached. This could depend on which state was chosen as start state.

Right now, TAILOR searches for the main path any time it is required to do so. Minor changes to the program would allow this main path to be stored in the object frame to avoid having to recompute it each time it is needed.

Note that finding the main path for the process trace is essentially equivalent to constructing the relevant knowledge pool, as done in [McKeown 82], since it delineates the part of the knowledge base used to generate a text.

7.6.2.1 Marking the side links

Side links are put aside while searching for the main path and marked afterwards. While traversing the most important event-links for which the event under consideration is the object, the program keeps all the other links on hold. After the main path is found, all the other links are examined and tagged as possible side links or side chains. For example, in Figure 7-15, the link corresponding to the relation ''*soundwaves-intensity varies* CORRESPONDS TO *current varies*'' was put on hold while searching for the main path. Once the main path had been found, TAILOR marked this link as a side link.

Once all the potential side links have been found, they are counted. If there are many, they will be grouped at the end of the process explanation instead of being mentioned as part of the explanation. (This corresponds to diagram (3) in Chapter 5.) A parameter indicates the value of *many*. If the number of side links is higher than the parameter, a flag is set. This flag is used to decide whether to include a side link or not while describing the main path.

7.6.3 Implementation of the Strategies

The constituency schema and the process trace strategies are implemented using augmented transition networks (ATN) [Woods 73]. The ATN implementation used in TAILOR is based on TEXT's implementation [McKeown 85]. It is described briefly in this section, and more details can be found in [McKeown 85].

The arcs joining the various nodes in the network include a test and specify what information is to be retrieved from the knowledge base, and which node to go to next. Registers are used to save information while traversing the graph.

Once a strategy has been chosen, its corresponding ATN is traversed. Information is retrieved from the knowledge base as the arcs of the ATN are chosen. For the constituency schema, the ATN arcs represent the rhetorical predicates of the schema. So traversing the ATN corresponds to filling the schema, matching the predicates against the knowledge base. For the process trace, the arcs are not rhetorical predicates but directives on how to traverse the event-links in the knowledge base.

Each time an arc is taken, a proposition is obtained, that corresponds roughly to a sentence in the text to be generated. A proposition is either an instantiation of a rhetorical predicate, or a value obtained from following a directive (from the process trace). When an arc is a rhetorical predicate, the *predicate argument* is the object the predicate is applied to, and the *predicate value* is the value retrieved from the knowledge base when the predicate is applied to an object.

A proposition contains values from the knowledge base, and is represented in a predicate-value form: the predicate is the name of the arc taken (i.e., either a rhetorical predicate or one of the directives). The value that was retrieved from the knowledge base when the arc was taken follows. Finally, some focus information is included. Focus information indicates which item is the focused item. It used both in deciding among several potential propositions and in choosing syntactic structure and lexical items to translate the proposition in English. Both aspects will be explained later. In case of a rhetorical predicate, the proposition also contains the entity the predicate is applied to, that is the *predicate argument.*

Examples of propositions are shown in Figures 7-17 and 7-18. While propositions use unique identifiers to refer to object frames, events and event-links, the corresponding English names are provided in both figures for clarity. Figure 7-17 presents propositions resulting from matching the constituency and identification predicates against the knowledge base. In both cases, the predicate argument is the *telephone.* The first element of the proposition is the name of the predicate. Its argument follows, that is, the object the predicate was applied to. The list following the predicate argument is the predicate value retrieved from the knowledge base for this object. Finally, the last element of the proposition is its focus.

Figure 7-18 shows a proposition resulting from following an event-link. This is indicated by the atom *process* as the first element of the proposition. The link that was retrieved follows, including both subject and object events. Finally, as in the previous examples, the proposition also includes its focus.

When the fact is retrieved from the knowledge base and included in the description, it is marked so that it will not be repeated later.

Propositions corresponding to matching the *constituency* and *identification* predicate against the knowledge base.

The predicate argument is the object the predicate is applied to.
The predicate value is the information retrieved from the knowledge base when the predicate is applied to an object.
The default *focus* for the proposition is the object the predicate is applied to (its argument).

```
 predicate     argument      predicate value                       focus

((Constituency  telephone  (transmitter, housing, line, receiver)  telephone)
```

English translation: *The telephone has a transmitter a housing, a line and a receiver.*

```
 predicate          argument   predicate value          focus

((Identification  telephone    (device))                telephone)
```

English translation: *The telephone is a device.*

Figure 7-17: Examples of propositions obtained from traversing an arc of the constituency schema

7.6.3.1 ATN Arc types

The arcs of the ATN in TAILOR can be of several types. The arc types used for the strategy implementation are the same as those used for the schema implementation in TEXT. They are:

- *Fill <predicate or directive>*: This arc retrieves information from the knowledge base. In TEXT, this arc was used to instantiate a predicate by matching it against the knowledge base. In TAILOR, it is used to either instantiate a predicate or to give a directive on how to trace the knowledge base. The result of this arc is a proposition that is included in the description to be generated.
- *Jump <state>*: Jump to a specified state (without fetching anything from the knowledge base). When considering this arc, the ATN driver

Proposition corresponding to one of the directives (*next causal link*) for following the event-links in the knowledge base. The default focus for the proposition is the event-link.

```
                    ((Process
Subject Event:       <person speaks-into transmitter>
Link:               &mr0 [CAUSES]
Object Event:       <soundwaves hit diaphragm>)
Focus:              &mr0)
```

English translation: *A person speaking into the transmitter causes soundwaves to hit the diaphragm.*

Figure 7-18: Example of a proposition obtained from traversing an arc of the process trace

simulates the jump and thus computes the result of traversing its successors (i.e., the arcs from the state indicated in the jump arc).

- *Subr <subroutine-state>*: This arc specifies to start a subnetwork (or subroutine). Subnets are included only to simplify the graph. They could be included in the main graph. The subr arc would then be simply a jump to a state. A detailed description of how subroutines (and recursive calls) are handled can be found in [McKeown 85] and will not be included here.
- *Subr-end*: indicates the end of the subnet. The ATN driver returns to the state following the state that called the subnet in the main graph.
- *Push*: recursive call to a net. Before traversing this net, all the registers are saved, and new values may be given to the registers.
- *Pop*: indicates the end of a recursive call. The values of the registers are restored.

The arcs can have pre-actions, a test and post-actions. Pre-actions are performed before an arc is taken. They can reset registers (in case of a recursive call for example) or retrieve the possible foci to be used by the functions that fetch information from the knowledge base. Tests are performed in order to decide whether an arc is appropriate or not. Tests are arbitrary LISP functions. They often test registers or values contained in the predicate argument frame. Post-actions are executed once an arc has been chosen. For a fill arc, post-actions typically add the new proposition to the message constructed so far, mark the item retrieved from the knowledge base to avoid repetition, and update registers. Most arcs include a post-action that indicates which state to go next.

7.6.3.2 Traversing the graph

The control structure of the ATN driver can be summarized as follows:

1. Retrieve all arcs emanating from a state.
2. Compute the test for each arc; save the arcs with successful tests.
3. For each arc saved:
    a. Perform its pre-actions;
    b. Match the arc against the knowledge base;

   This results in a pool of possible propositions.
4. Choose one proposition (i.e., one arc) among this pool. This step will be discussed after presenting what information is retrieved from the knowledge base for each type of fill arc.
5. Perform its post-actions (e.g. mark the item just retrieved from the knowledge base). One of the post-actions is a jump to the next state.
6. Go back to (1), starting with the next state.

Because of the availability of facts in the knowledge base and the fact that the ATN contains several options, I did not encounter the need to have the system backtrack after it has chosen an arc because it reached a blocked state. Backtracking could be done if necessary as the whole text is constructed before being generated. (Note that it would also be possible to generate as propositions are retrieved from the knowledge base, although this would make backtracking much harder.) Backtracking would require keeping a record of what has been used in a register instead of marking the used item in the knowledge base as is done now.

7.6.4 Stepping through the Constituency Schema

The arcs of the ATN for the constituency schema correspond to the predicates of the schema. The rhetorical predicates define the type of information to be retrieved from the knowledge base. The ATN corresponding to the constituency schema is given in Figure 7-19. (This figure was already given in Chapter 5 and is repeated here for clarity.)

The first predicate indicated in the schema, identification, is optional. It is thus possible to go from state *CONST/* to *CONST/Intro* either by taking the arc labelled *identification* or with simply a jump to *CONST/Intro*, in which case the identification predicate is skipped and the *constituency* predicate taken immediately. The *constituency* predicate dictates to present the subparts of an object. This is possible when the object has parts, that is when the object is *composite*. To allow for *unitary* objects, the schema was slightly changed by adding another alternative, the arc labelled *attributive*. This arc is taken only for unitary objects, that is objects that do not have any subparts. This is indicated in the arc test. This test looks at the object frame to determine its type and decides which arc (attributive or constituency) is appropriate. If the *constituency* arc is taken, a register that indicates the subparts is set. This register will be used to supply more information about each part in turn.

After taking either of these arcs, the ATN driver then considers the next state, *CONST/Const*. At that point, one of three predicates can be taken for each subpart included in the parts register: *depth-identification*, *constituency* or *depth-attributive*. When there are no more parts in this register (it gets updated each time an arc is taken), a jump arc or the arc labelled *attributive* is chosen. The schema ends and the constructed text is returned.

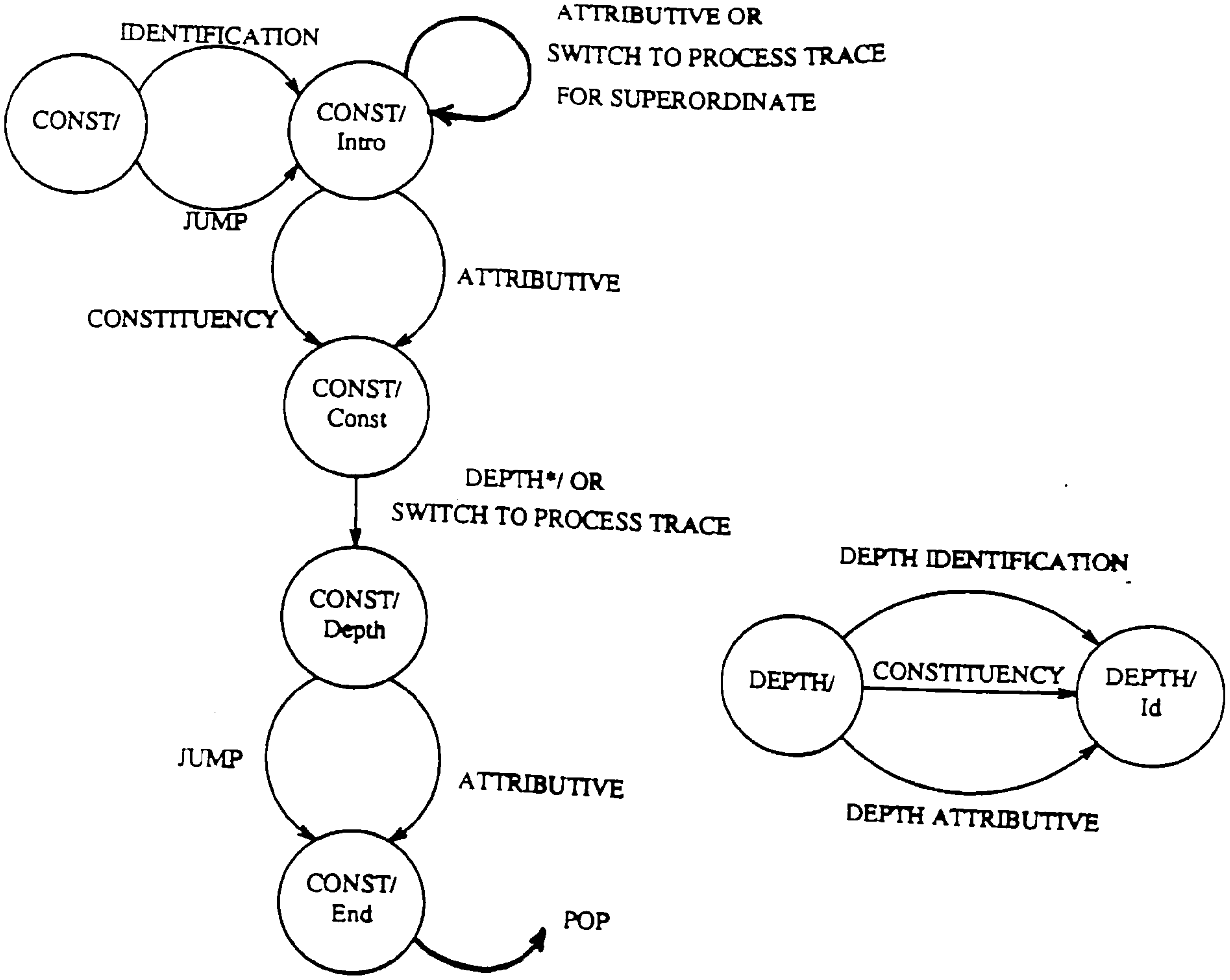


Figure 7-19: Constituency Schema and its ATN

7.6.4.1 Predicate Semantics

Predicate semantics are implemented using functions that actually fetch the appropriate information. It is possible for a predicate to match several facts in the knowledge base. This is the case, for example, for the *attributive* predicate which provides properties or structural attributes about an object. In that case, all possible matches are retrieved, each forming a different proposition that gets added to the pool of potential propositions. After all the potential propositions have been formed, directives are applied to choose the most appropriate one.

As mentioned in Chapter 4 , propositions corresponding to several predicates were sometimes merged into one sentence at the surface level. This can be done using a sophisticated interface capable of combining two or more propositions. Not having such an interface, TAILOR simulates this output by allowing the semantics of some predicates to retrieve more information than strictly required by the linguistic predicates. For example, the constituency predicate dictates to retrieve the parts of an object. In TAILOR's implementation of the function corresponding to this predicate, properties associated with the parts are can also retrieved. Figure 7-20 illustrates this phenomenon.

In the first sentence of (1) in Figure 7-20, the function corresponding to the constituency predicate simply retrieves the parts of the object. In the second sentence, the depth-attributive predicate is applied to the parts, and we obtain attributive information about the diaphragm. In (2), parts are immediately retrieved with their properties. The same information as in (1) is conveyed, though in a more compact form.

Global flags determine whether TAILOR is allowed to collapse propositions. It is also possible to turn them off and obtain single sentence production for each

---

1. Apply the constituency predicate to *the microphone*. Then, apply depth-attributive to the parts:

The microphone has a diaphragm and a system.
The diaphragm is disc-shaped and aluminium.
The system is doubly-resonant.

2. Apply the constituency predicate to *the microphone*, with properties of parts allowed:

The microphone has a disc-shaped aluminium diaphragm and a doubly-resonant system.

Figure 7-20: Including more information than strictly required by the predicates

---

proposition. Figure 7-21 gives examples of the semantics of each predicate, with the different options available.

The identification predicate retrieves the superordinate of an object in the generalization hierarchy. With the appropriate flags on it is possible to obtain at the same time the purpose of the object, as illustrated in 1.2 in Figure 7-21. The constituency predicate retrieves the subparts on an object (see 2.1 in the Figure), and it is possible to include properties (as in 2.2) and purposes (as in 2.3). The attributive predicate matches attributes of the object. This includes properties such as material and shape (as in 3.1), and structural relations relating the current object with previously mentioned objects, (as in 3.2). Note that structural relations involve several objects. It is thus possible for a relation to match the attributive predicate with several different arguments (all the objects that take a part in that structural relations). A text produced from stepping through the constituency schema is shown in Figure 7-22.

Identification:

1.1 (IDENTIFICATION &MEM1 (device))
The transmitter is a microphone.

Purpose of the object allowed:
1.2 (IDENTIFICATION &MEM1 (device) (USED-FOR (&REL27 &MEM17 &MEM19)))
The microphone is a device that changes soundwaves into current.

Constituency:

2.1 (CONSTITUENCY &MEM1 ((&MEM4) (&MEM11)))
The microphone has a diaphragm and a system.

Including subparts properties:
```
2.2 (CONSTITUENCY &MEM1 ((&MEM4  (PROPERTY (SHAPE DISC)
                                  (MATERIAL ALUMINIUM)))
                 (&MEM11 (PROPERTY (TYPE DOUBLY-RESONANT)))))
```
The microphone has a disc-shaped aluminium diaphragm and a doubly-resonant system.

Including subparts purposes when available:
```
2.3 (CONSTITUENCY &MEM1 ((&MEM4)
                 (&MEM11 (USED-FOR (&REL8 &MEM11 &mem21)))))
```
The microphone has a diaphragm and a system to broaden the response.

Attributive:

Retrieving properties:
3.1 (ATTRIBUTIVE &MEM4 (PROPERTY (SHAPE DISC) (MATERIAL ALUMINIUM)))
The diaphragm is disc-shaped and aluminium.

Retrieving structural relations:
3.2 (ATTRIBUTIVE &MEM4 (RELATION (&REL7 &MEM4 &MEM34)))
The diaphragm is mounted on the poles of the magnet.

Figure 7-21: Predicate semantics

Loudspeaker:

```
The loudspeaker changes current into soundwaves.  It
has a large thin dome-shaped paper diaphragm, a ring-
shaped permendur armature, a coil, a ring-shaped
permanent magnet and a gap.  The diaphragm is mounted on
the poles of the magnet.  The gap contains air.  The gap
is between the poles and the diaphragm.  The coil is
mounted on the magnet.
```

Figure 7-22: Stepping through the Constituency Schema

7.6.5 The ATN corresponding to the Process Trace

In Chapter 5, I presented the process strategy in detail. In particular, I described all the different side link structures that might occur and specified how they should be treated. For each of these structures, there is an arc in the ATN. The process trace as explained in chapter 5 only referred to producing a causal explanation of the device's mechanism. An introductory statement is actually also included in the strategy, in order to not immediately start the description with a causal relationship. This is represented in the ATN with the first arc, labelled *identification*. The other arcs of the ATN for the process trace dictate how to traverse the event-links contained in the knowledge base in order to produce a coherent process description. The first step in this process explanation is to find the main path. This step was explained earlier in this chapter. Given the main path, a list of event-links, the process trace mainly traces each link at a time. The ATN corresponding to the process trace was shown in Figure 5-15 in Chapter 5. It is repeated in shown in Figure 7-23. Its arcs are:

- *Identification*: This is identical to the identification predicate in the constituency schema. Here, the device is identified with its function.
- *Long-side-chain1?*: This corresponds to the side link structure shown in diagram 1-b in Chapter 5. There is a long side link that is not initiated

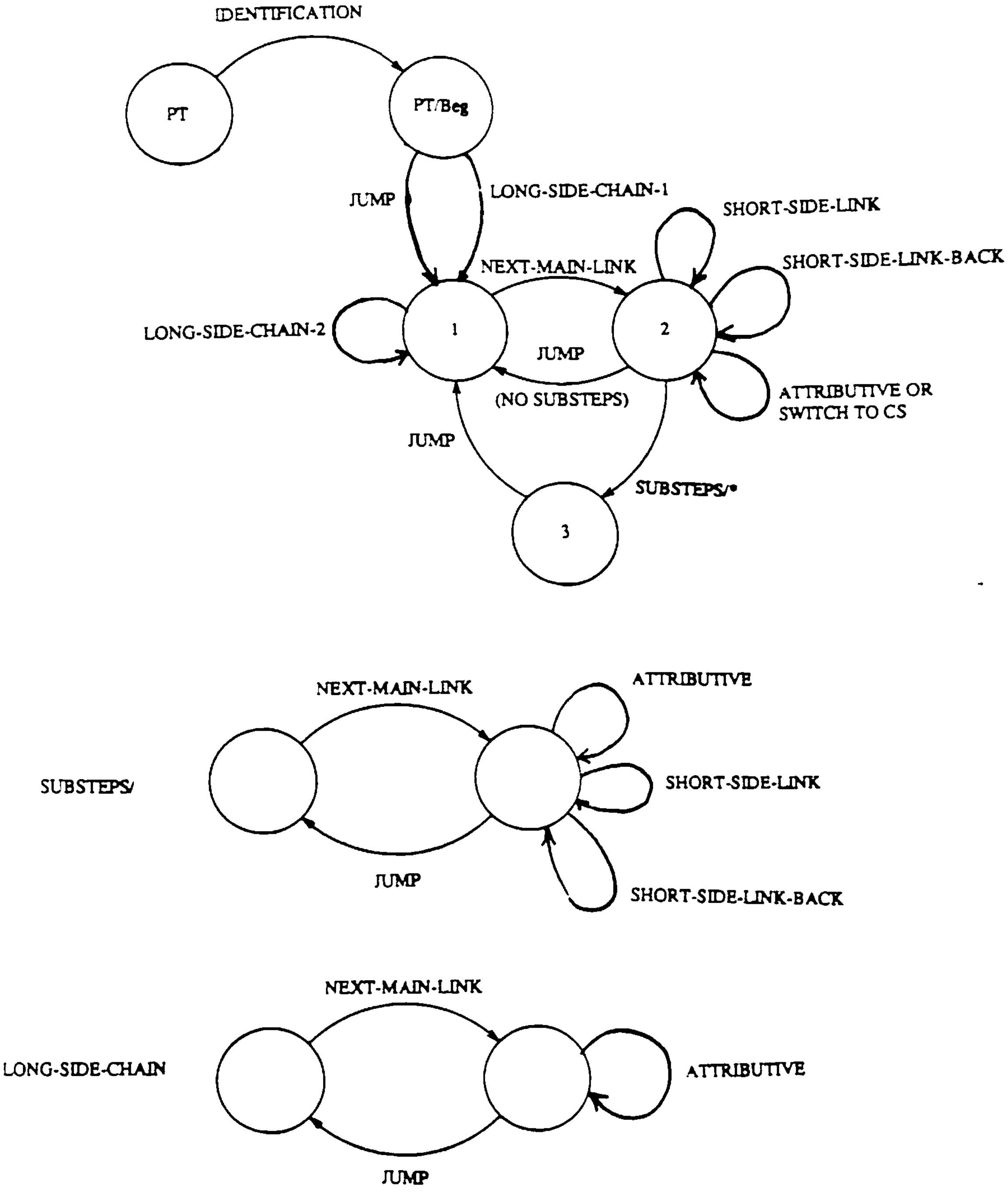


Figure 7-23: ATN corresponding to the Process Trace

from the main path but is reattached to the main path at a later point. This link needs to be included if it is an enabling (or causal) condition for an event on the main path. The link is traversed before starting the main path.

- *Next-main-link*: The main path consists of several event-links. This directive indicates to take the next link on the main path. When the arc is taken, the link is taken off the list that contains the events on the main path. Each time a link is taken, all the parts that are mentioned in the events are collected, so that attributive information might be given about each of them.
- *Long-side-chain2?*: This corresponds to the side link structure shown in diagram 1-a in Chapter 5. There is a long side link that comes off the main path at event (1) and gets reattached to the main path at a later point. The link is mentioned when event (1) has been introduced.
- *Short-side-link?* and *Short-side-link-back?*: This corresponds to diagram 2-a and 2-b of chapter 5, where a short side link is included as part of the process explanation. A test on the arcs makes sure that the register that indicates that there are too many short side links to include them as part of the process explanation is not set.
- *Attributive*: This arc is similar to the attributive predicate of the constituency schema. However, this predicate is given one constraint, that of retrieving only properties about the object, not structural relations. This was decided as it seems that it produces better descriptions for a naive user. The predicate is applied to all the parts that were just mentioned as part of the process explanation.
- *Substeps?*: This arc checks whether the event just mentioned is decomposable. If it is and the substeps do not involve basic concepts the user does not understand, the substeps are followed. The test actually also checks the length of the text generated so far so avoid producing very long texts.

A text generated by traversing the ATN of the process trace is shown in Figure 7-24.

7.6.6 Choosing an arc

Given a pool of possible propositions, one must be eventually chosen. This decision is partially based on the focus information contained in the proposition. Focus information is used to keep a text as coherent as possible by making sure propositions are related to each other in some ways. TAILOR uses the focus

Oscillator

An oscillator is a device that produces a varying current, when a battery produces a current.	identification
Because the battery produces the current, the transistor turns on.	next-main-link
That the transistor turns on is caused also by the capacitor discharging through the resistor.	short-side-link -back
The transistor turning on causes the capacitor to charge.	next-main-link
This causes the transistor to turn off.	next-main-link
Because the transistor turns off, the capacitor discharges through the resistor.	next-main-link
The resistor has low resistance.	Attributive
The capacitor discharging through the resistor causes the varying current to be produced.	next-main-link

Figure 7-24: Description using the Process Trace

guidelines defined by McKeown in the TEXT system. In TEXT, McKeown adapted for generation the focus rules Sidner had identified for use in discourse analysis. These focus guidelines, which are described in detail in [McKeown 85], dictate to choose, in order:

1. a proposition whose focus is among items (or concepts) that were just introduced in the previous proposition
2. a proposition with the same focus as the previous proposition
3. a proposition whose focus was the focus of a past proposition (an item previously discussed).

Focus information must be maintained in registers for these decisions to be made. The registers are updated after each proposition is added to the text to be generated.

In TAILOR, focus information alone, however, rarely constrains the decision process totally, and a choice is made based on other factors. One factor used in TAILOR is lexical choice, in particular in the case of the attributive predicate. As mentioned previously, it is possible for a relation to match the attributive predicate with several different arguments, that is with the focus on several different items. To decide when it might be appropriate to include this specific fact from the knowledge base, the textual component calls the dictionary interface to check whether the system has a lexical item for this relation that would place the focus on the required focused item. If no such lexical item can be found, the relation is not chosen for the item.

As an example, consider the relation *the air gap is between the poles and the diaphragm*. This relation matches the attributive predicate when the predicate is applied to any of the three items involved in the relation: the gap, the poles and the diaphragm. It is only when the relation is applied to the gap, however, that the dictionary interface returns an English word for the relation, that places the focus on the predicate argument. For the two other items, no lexical choice can be found with the required focus. As a result, this relation is chosen only when the predicate is applied to the gap. This is the only interaction between the textual component and the surface generator in TAILOR, where the surface structure and lexical choice affects the organization of the text.

Another factor involved in the decision process in TAILOR is making sure that a relation is mentioned only after all its constituents have already been introduced. For example, before including an analogical link between two relations, it is preferable to have first introduced the two events that are related by the analogical link. This is illustrated in Figure 7-25.

The analogical link *the soundwaves varies like the current* is included without checking whether both events have already been introduced

```
The current varies, like the intensity varies. The
variation of the current causes the field of the magnet
to vary. This causes the diaphragm to vibrate. The
vibration of the diaphragm causes the intensity of the
soundwaves to vary.
```

A more appropriate text would be:

```
The variation of the current causes the field of the
magnet to vary. This causes the diaphragm to vibrate.
The vibration of the diaphragm causes the intensity of
the soundwaves to vary. The intensity varies, like the
current varies.
```

Figure 7-25: When to include a relation

7.7 The Interface

TAILOR's interface takes as input the content of the description from the textual component and assigns lexical items to the various concepts contained in the description. The output of the interface is a functional description of the sentence to be generated. This functional description can be fed into a surface generator that will unify the input with a functional grammar to produce English. The output of the interface must be in the same formalism as that of the functional grammar. Figure 7-26 shows an example of an input to the interface and the output that is be produced.

The interface uses a lexicon to retrieve the lexical item(s) associated with entities, relations, or event-links. This lexicon is the same as the one used by RESEARCHER in parsing texts. Syntactic information and constraints were added to the original

---

Interface Input:

```
((IDENTIFICATION &MEM1 (DEVICE)) &MEM1)
```

Interface Output:

```
((cat s) (prot ((n === telephone) (article === def)))
       (verb ((v === be)))
       (goal ((n === device) (article === indef))))
```

English that would be generated after unification with the grammar:

The telephone is a device.

Figure 7-26: Interface input and output

---

lexicon. The syntactic information mainly indicates the part of speech of the item, as this is needed to construct the sentence. Constraints are indicated by a test. This test is an arbitrary function that is used to either test the appropriateness of the lexical item (possibly based on focus information) or to return syntactic constraints (such as "this verb must be used with the preposition *to*.)" The interface uses focus information and previously mentioned items to decide on articles. Focus information is included in the output of the interface, as it will also be used in the surface generator to decide on the verb voice (passive or active).

The interface is highly modular and regular. Frames of the same types are always translated in the same way, and the structure of the sentence to be generated is usually dictated and constrained by the predicate of the proposition.

For example, in figure 7-26, the proposition resulted from applying the *identification* predicate to the object frame *&mem1*. Because of the identification predicate, the verb *to be* was chosen by the interface. Once the main verb of the sentence was selected, the verb roles are filled by the other elements of the proposition. In the figure, as *&mem1* is the predicate argument, its corresponding English word, the *telephone* becomes the subject, or protagonist, of the verb. *Device*, the predicate value (i.e., the fact retrieved from the knowledge base when *identification* was applied to the telephone), becomes the object, or goal. The determiners are chosen depending on whether a particular item has already been mentioned or whether the item is the object of the description. After having decided on the translation for the predicate of the proposition, the interface calls itself recursively to assign lexical items to the predicate argument and value, and fills the appropriate roles.

The predicate of the proposition constrains the syntactic structure of the sentences as indicated in Figure 7-27. The identification predicate is always translated into the verb *to be*. The predicate argument becomes the subject of the verb, while the predicate value becomes the object. This is illustrated in Figure 7-28.

The constituency predicate is always translated into the verb *have*. The predicate argument is the subject of the sentence as for the identification predicate. The predicate value, here the constituents, form the object of the verb.

The attributive predicate is translated into the verb *to be* when the predicate value is one or more properties about the object. These properties are translated into adjectives. If the predicate value is a structural relation, (i.e., a relation record), the interface is called recursively with the relation record to form a sentence. A relation record is always translated into a sentence, where the relation dictates the verb. A

Identification predicate: Verb = be
Protagonist = item the predicate was applied to
Goal = information retrieved from the knowledge base
(Superordinate of the object)

Constituency predicate: Verb = have
Protagonist = item the predicate was applied to
Goal = information retrieved from the knowledge base
(constituents of the object)

Attributive predicate:

1) Information retrieved from the knowledge base are properties

Verb = be
Protagonist = item the predicate was applied to
Goal = adjectives (for the properties retrieved)
from the knowledge base

2) Information retrieved from the knowledge base is a structural relation

Call the interface to translate the structural relation,
i.e., the relation record.

Relation record: Verb = English word corresponding to the relation frame
Protagonist and Goal: depend on the syntactic information
indicated in the lexicon for the lexical item chosen.

Process link: a complex sentence constructed around the translation
of the link and that of the two events related by the link.

Figure 7-27: Translation of the various propositions

```
Proposition: ((identification &mem43 (&mem20))  &mem43)

Identification predicate ===>  verb = be
                 subject  &mem43 [telephone transmitter]
                 object   &mem20 [microphone]

Structure formed:
                ((cat s) (verb ((v === be)))
                         (prot ((n=== transmitter)
                                (article === def)))
                         (goal ((n === microphone)
                                (article === indef))))
```

Corresponding English: The transmitter is a microphone

Figure 7-28: Constructing a sentence from the identification predicate

lexical item is chosen for a relation, usually depending on the focus of the proposition. For example, in the relation <*the gap contains air*>, if the focus of the proposition is on the *gap*, the verb *contains* will be chosen. However, if the focus is on *air*, the verb *to be* is chosen together with the preposition *inside of.* The chosen lexical item and the focus of the proposition thus dictates how to construct the sentence and fill the other constituents. The translation for the attributive predicate is illustrated in Figure 7-29.

All relations (whether structural or functional) are systematically translated into *simple* sentences.

Finally, most process links (event-links) are translated into complex sentences. As a process link relates two events, each of which is a functional relation, each event is translated at first into a simple sentence of the kinds shown above. A complex

---

Proposition:

```
     ((attributive &meml3 (properties (size small)))  &meml3)
```

```
Identification predicate ===>  verb = be
                  subject:  &meml3 [diaphragm]
                  adjectival phrase:   small
```

Structure formed:

```
                ((cat s)  (verb ((v === be)))
                          (prot ((n=== diaphragm)
                                 (article === def)))
                          (ap ((adj === small))))
```

Corresponding English: The diaphragm is small

Proposition:

```
     ((attributive [gap] (&rel43 [contain] [gap] [air])) [gap])

        &rel43 ==> verb = contains
```

Structure formed:

```
                ((cat s)  (verb ((v === contain)))
                          (prot ((n=== gap) (article === def)))
                          (goal ((n === air))))
```

Corresponding English: The gap contains air

Proposition:

```
    ((attributive [air] (&rel43 [contain] [gap] [air])) [air])
        &rel43 ==> verb = be, with the preposition ``inside of''
```

Structure formed:

```
                ((cat s)  (verb ((v === be)
                                 (pp ((prep === inside-of)
                                      (n === gap)
                                      (article === def)))))
                          (prot ((n=== air))))
```

Corresponding English: The air is inside of the gap

Figure 7-29: Constructing a sentence from the attributive predicate

---

sentence is then formed by joining the two sentences based on the event link in a variety of ways. For example, there are a number of ways a cause-effect relation can be expressed:

- Using the explicit verb *cause*, as in "the current varying causes the soundwaves to vary."
- Using a subordinate, introduced by the subordinator *because*, as in "because the current varies the soundwaves vary." This sentence can also be expressed by reversing the order of the main clause and subordinate to get "the soundwaves vary because the current varies."
- Using a subordinate, introduced by the subordinator *when*, as in "when the current varies, the soundwaves vary." As in the previous case, the order of the two clauses can be reversed.
- Using the noun form of the verb if possible, as in "the variation of the current causes the soundwave to vary."

The interface is able to combine two simple sentences in all ways shown above. Because of extensions performed on the grammar, the interface only needs to specify the type of complex sentence desired, and the actual complex sentence is constructed by the grammar. As an example, to use a subordinate introduced with the subordinator *because*, the interface need only to form, from the two simple sentences, the structure shown in Figure 7-30. "Embed" is used in the protagonist and goal to indicate that the constituent is a sentence. The two features "surface" and "order" indicate the structure of the desired output. To translate an event-link, the interface translates the two events separately as simple relation records and joins them together by adding the desired features. If no features are added, the grammar defaults to the following construction: "the soundwaves varying causes the current to vary".

A subordinate clause introduced with the subordinator "when" is used to indicate the beginning of a substep chain. When several forms are available, the interface keeps track of the last form used and employs each form in turn for variety. To add

```
Proposition to translate:

<the soundwaves vary> causes <the current vary>

1) each event is translated into a simple sentence:

      ((cat s)   (verb ((v === vary)))
                 (prot ((n === soundwave) (number plur)
                        (article === def))))

      ((cat s)   (verb ((v === vary)))
                 (prot ((n === current) (article === def))))

2) construct the complex sentence by specifying to the grammar
the construction desired: a subordinate, introduced with the
subordinator because.

   ((cat s)
      (surface because) (order front)
      (verb ((v === cause)))
      (prot
        ((embed ((cat s)   (verb ((v === vary)))
                           (prot ((n === soundwave) (number plur)
                                  (article === def))))))
      (goal
        ((embed
                   ((cat s)   (verb ((v === vary)))
                              (prot ((n === current)
                                     (article === def)))))))
```

Figure 7-30: Combining simple sentences into a complex sentence

to the quality of the texts generated, the interface also makes note of the last event that was mentioned. This allows it use the pronoun "this" when it is possible, as illustrated in Figure 7-31.

Finally, objects are translated into the appropriate nouns. Sometimes, two different objects have the same English translation. To be able to unambiguously refer to them, the interface keeps a list of all the objects that are mentioned in the text. If two objects have the same English translation, the interface tries to disambiguate them, by

---

Suppose we had the two event-links:

`The soundwave intensity increasing` **causes** `the diaphragm to spring forward. The diaphragm springing forward` **causes** `the granules to be compressed.`

By keeping track of the last mentioned event, the interface can replace the subject of an event link by the pronoun *this*:

`The soundwave intensity increasing` **causes** `the diaphragm to spring forward. This` **causes** `the granules to be compressed.`

Figure 7-31: Using the pronoun *this* in a process explanation

---

including their respective properties. For example, the dialing mechanism has two gears, one small and one large, connected to each other. The translation "the gear is connected to the gear" is clearly ambiguous. By adding the properties associated with each object, "the small gear is connected to the large gear" is obtained, and each item is identified unambiguously. While simple, this disambiguation technique has been successful in TAILOR. This technique would not be adequate as a general disambiguation technique as it would break down if the objects had the same properties, in which case another way to differentiate the two objects would be required.

7.8 The surface generator

The surface generator takes the output of the interface and produces English. The surface generator used in TAILOR is based on that of TEXT and uses a functional unification grammar (FUG) as defined by [Kay 79]. The input is unified with the grammar to produce English.

The functional grammar was chosen because of its availability and its "clean" formalism. The functional grammar is represented in a declarative form. Because of the separation of the functional grammar and the unifier (the program that unifies the grammar and the input to produce English), it possible to change or augment the grammar without having to change any of the unifier's code. Moreover, using the functional grammar, various constraints can be directly encoded in the grammar, thus simplifying the interface that constructs the input to the grammar. For example, one constraint can state that the verb voice is dependent on the focus of the sentence. These global constraints can be represented separately from syntactic rules, so that they need to be stated only once; this is not the case in other formalisms (see [McKeown and Paris 87]). Finally, as the input is unified in the grammar, many syntactic details can be included in the grammar and the input simplified. For example, the input need not specify the number of the verb.

I have greatly improved the performance of the original unification program, and, although unification is non-deterministic, the program now generates sentences at a speed similar to that of a deterministic generation system. In this section, I first present briefly the unification process,[23] and then discuss TAILOR's grammar and unifier.

7.8.1 The functional grammar and the unification process

The functional grammar is called a functional description (FD). It is made of attribute-value pairs. Each attribute-value pair is itself a functional description and can also be formed of other FD's. The whole grammar contains subgrammars for each possible syntactic category, such as *sentence*, *noun phrase or np* and *verb group*.

---

[23] Details about the unification process can be found in [McKeown 85; Paris and Kwee 85]

There are two steps in generating an English sentence: *unification* with the grammar, and *linearization* of the resulting structure. The input to the unification process is a deep structure of the sentence to be generated, produced by the interface. It typically does not contain all the syntactic information necessary to generate the sentence. This input is unified with the grammar and enriched with all necessary syntactic information, including information about the order in which constituents should appear in the sentence. The linearizer then takes this enriched input and produces a flat list, the English sentence. Morphology and punctuation is also performed in the linearizer.

The unification process unifies each functional description in the input with the grammar. To unify a functional description, the unifier considers each attribute-value pair in the grammar and unifies its value part with that of the corresponding attribute in the input if present. If an attribute occurs in the grammar but not in the input, the input is enriched with the attribute-value pair from the grammar. (That is, the resulting structure is the *union* of the input and the grammar.) The grammar is thus used to enrich the input with all the syntactic information necessary to produce a sentence.

7.8.2 TAILOR's grammar

TAILOR's grammar grew out of one used in TEXT and was extended to support more syntactic constructions and to include more constraints on choices, thus simplifying the interface. The reader is referred to [McKeown 85] and [Paris and Kwee 85] for a full description of the grammar's features. The main changes to the grammar are described in this section.

1) The verb voice is now chosen based on focus information, and, when neither voice nor focus is indicated in the sentence, the voice is defaulted in the appropriate way (i.e., passive is chosen if there is no protagonist in the sentence).

2) Copular verbs have been added, as this construction was needed to express the attributive predicate. In a sentence with a copular verb, there is no goal, but an adjectival phrase. This is used in the translation of the attributive predicate, when the predicate value include properties.

3) More complex constructions have been added [Paris and Kwee 85; Kwee 87]. These include subordinate sentences and embedded clauses; that is, a sentences embedded in a constituent. Both are needed to express relationships among events, as shown in Figure 7-32.

Event link to be expressed:

<the soundwaves vary> causes <the current varies>

Using a subordinate clause:
Because the soundwaves vary, the current varies.

Using an embedded clause:
The soundwaves varying causes the current to vary.

Figure 7-32: Embedded clauses and their use in TAILOR

4) The grammar is able to construct a complex sentence from two simple ones, given explicit information on the structure to be used. This is important, as, in TAILOR, there is a great need for complex sentences to express process information. Explicitly constructing complex sentences in the interface is a tedious and complicated process. By having the grammar construct the complex sentence from simple sentences, the interface is greatly simplified. I showed in Figure 7-30 the

input the interface needs to construct. All the various complex sentences mentioned previously can be constructed in the grammar in a similar manner. A default construction is also provided, in case no specification occur in the input.

5) Besides constructing a complex sentence, the grammar is also able to use the noun form of a verb (whenever possible), resulting in a smoother text and greater variety of constructions. For example, instead of constructing a sentence from ''the diaphragm vibrates'' as one of the constituent of a complex sentence, the grammar is able to form the noun phrase ''the vibration of the diaphragm.'' This is done in a similar way as combining simple sentences: the feature (trans verb-noun) needs to be added in the input. The grammar then checks its lexicon to see if it has a noun form for the verb. If it does not, the result is the default construction.

TAILOR's grammar now contains many new constructions, and a wide variety of complex sentences. Furthermore, more decisions now occur in the grammar. Because of the complexity of the interface, it is desirable to include as many decisions as possible in the grammar instead of in the interface.

TAILOR's grammar is now fairly complete at the sentence level, but no connected discourse is accounted for: the grammar can only process one sentence at a time. If several sentences need to be joined together, this has to be done in the interface. Further extensions could include allowing the grammar to automatically generate connected discourse from several propositions.

7.8.3 TAILOR's unifier implementation

It has been argued that FUG is an inherently inefficient process due to its non-determinism [Ritchie 86; Rubinoff 86]. Some of these arguments, [Rubinoff 86] in particular, were based on the observation that the TEXT implementation of a FUG took up to 20 minutes of real time to produce a paragraph. This observation led researchers to believe that FUG was too slow for a practical generation system. It should be noted, however, that looking at real time is not necessarily indicative as, on time sharing machines such as the VAX/780 on which the TEXT system was running, real time is heavily load dependent. Using the numbers indicated in [McKeown 82], we computed that TEXT's FUG really processed one sentence in about 2 minutes (CPU time) [McKeown and Paris 87].

While this processing speed is still rather slow, I improved the system's efficiency, and TAILOR's implementation of FUG generates a sentence in about 2 seconds on the average on the IBM 4381, which is quite acceptable for a practical generation system. Furthermore, this compares favorably with other generation systems using deterministic methods. For example, [Rubinoff 86] cites that his adaptation of MUMBLE on the Symbolics 3600 generates a sentence in 5 seconds. TAILOR's unifier is now 3.5 times faster that TEXT's unifier when run on the same machine [McKeown and Paris 87], which is a better speed up than the one obtained by replacing the unification grammar by a deterministic generator (such has been done for the TEXT system, where the FUG has been replaced by MUMBLE. See [McKeown and Paris 87] for a comparison of the efficiency of two systems.) Furthermore, this speed up in the new unifier was obtained by doing only minor changes in the unification algorithm. These changes will be presented here. Considerably more speed up might to occur with further modifications. Efficiency of the functional grammar is thus not as much of a problem as was previously thought.

The first change involved immediately selecting the correct category for unification from the grammar whenever possible. For example, when the input specifies *cat np*, this category is immediately selected for unification with the input. In the old unifier, all the different categories would be tried first (as they represent different alternatives). Although unification with these categories would result in failure in a short amount of time, it would needlessly add to the process time. Immediately retrieving the correct category saves a number of recursive calls and thus improves the program performance.

Unification with the lexicon was changed to use the same technique. The correct lexical item is directly retrieved from the grammar for unification, rather than unifying with each entry in the lexicon successively, as was previously done.

Another change involved the generation of only one sentence for a given input. Although the grammar is often capable of generating more than one possible sentence for its input,[24] in practice, only one output sentence is desired. In the old version of the unifier, all possible output sentences were generated and one was selected. Since only one is needed, the program was changed to only generate the first successful result unless specifically directed otherwise.

Other minor changes were made to avoid recursive calls that would result in failure. Finally, other changes were made to the unifier, as a few problems arose when the unifier was used with the expanded grammar to produce rather complex sentences. All these changes are described in detail in [Paris and Kwee 85].

7.9 Issues pertaining to domain dependency

In this chapter, I presented the implementation of the TAILOR system. The strategies described here, their corresponding ATN and the ATN driver can all be used in different systems, although the tests might need to be changed to reflect the change in domain and knowledge base. The functions that actually fetch the information from the knowledge base are specific to the TAILOR system and its

---

[24] Often the surface sentences generated are the same, but the syntactic structure built in producing the sentence differs.

knowledge base and would have to be adapted in another domain. Similarly, the interface is dependent on the knowledge base in that each frame type requires a different syntactic construction. On the other hand, the surface generator, including the grammar and the unifier, is entirely domain independent and can be moved without any changes (except to the grammar lexicon).

Each of the three components of TAILOR is modular and interaction among them is limited to a few places. As a result, each could be transported separately and applied to another system. For example, the interface and the grammar could be replaced by the equivalent for another language, and descriptions in that language could be generated. Similarly, the output of the textual component could be passed to a graphics generator interface instead of a natural language generator. Alternatively, the textual component could be augmented to address more question types or include more strategies. In that case, the interface might have to be augmented to reflect the addition of more predicates or directives, but much of it would remain. The functional grammar could also be still used. Note, however, that more interaction among the components might be necessary to produce smoother texts.

7.10 TAILOR as a question answering system

As the generation component of a question answering system TAILOR at this point has two main limitations:

1) As discussed in Chapter 1, I am only concerned here in generating descriptions. Thus TAILOR only handles requests for descriptions, in the form shown above. A full question answering system would have the capability to answer any question that might be asked, although descriptions are a good starting point.

2) TAILOR is not an interactive system and there is very little feedback from previous discourse. In TAILOR, the user issues a request for a description and the

description is generated. The object that was just described is added into the user model, as TAILOR assumes the user now has local expertise about it. TAILOR does not keep a record of all the questions asked by the user and the answers provided. As a result, TAILOR cannot detect that a question is asked several times.

8. Conclusions

In this thesis, I have shown the feasibility of incorporating a model of a user's knowledge about the domain in a generation system. In this Chapter, I summarize the main points of this work. Then, a discussion of the feasibility of this approach is presented, as well as directions for future work.

8.1 Main points of this thesis

The user's level of expertise affects the kind of information to include in a text

Chapter 4 presented detailed analysis of various texts describing complex devices. The texts studied were aimed at users at the extremes of the knowledge spectrum, from naive to expert. The analysis revealed that the user's level of expertise affected the kind of information as opposed to only the amount of detail. This result is important as it demonstrates that the level of detail is not the only parameter to vary when tailoring an answer to a user's level of domain knowledge, as was previously assumed by designers of generation systems.

The process trace is a procedural strategy

From the text analysis, I identified two strategies that were used in naturally occurring texts to describe objects. One of these strategies, the constituency schema, had been previously posited by McKeown [85]. The other strategy, the process trace, is a new type of strategy which consists of directions on how to traverse the knowledge base instead of being formed of rhetorical predicates, like the constituency schema. This strategy was termed a *procedural* strategy, as it follows the knowledge base very closely. In contrast, the constituency schema, which I called a *declarative* strategy, is not dependent on the structure of the underlying knowledge base but rather imposes a structure on it. Identifying a new strategy for generation is useful as it can be added to the tools available to a generation system, and, therefore, gives a system more flexibility.

It is feasible to incorporate a model of the user's domain knowledge in a generation system

The two discourse strategies identified in texts were implemented in TAILOR, a system that generates descriptions of complex devices tailored to a user's level of expertise. The user model guides TAILOR in choosing an appropriate discourse strategy at every point in the generation process. TAILOR demonstrates that a generation system can make use of the user's domain knowledge to help choose appropriate facts from the knowledge base.

The strategies can be combined systematically to generate a wide variety of texts, tailored to a whole range of users

I showed how TAILOR could automatically combine the strategies in a systematic way to produce texts tailored to a whole range of users. The ability to combine the strategies gives a generation system greater flexibility than otherwise allowed. That is, instead of being able to generate only two different texts about the same object, a system can now generate a whole range of texts about the same object by combining the strategies in different ways. Furthermore, because of the explicit representation of its user model, TAILOR can tailor descriptions to users falling anywhere along the knowledge spectrum without requiring an *a priori* set of user types.

8.2 Feasibility and extensibility of this approach

In this work, I make the assumption that it is desirable for a system to tailor its answer to its users. This is true only provided that this tailoring does not hinder the system's performance or increase its complexity significantly. I argue that tailoring a description to a user's level of expertise using the method described in this thesis will not add much to the cost of generation and yet provides better answers. Whether a generation system tailors its answers to users or not, it needs a control structure that will guide the decision process both in determining facts to take from the knowledge base and in organizing them, lest the resulting text be incoherent. A discourse strategy is one way to guide its decision process that has been used successfully in previous generation systems. The two discourse strategies used by TAILOR are of comparable complexities to those used in other systems and, in fact, one is identical. Other systems employing discourse strategies must decide which strategy to use to produce a text. The decision is usually based on the question type and the structure and content of the knowledge base. In TAILOR, the user model plays a role, but this does not add any cost to the decision process. Furthermore, to combine the strategies does not add cost either, especially because of the formalism used to represent the strategies. By representing both strategies with an ATN, the control structure necessary is combine the strategies is readily available. It is not more costly to jump from one node in one network to another node in the same network than to go from one network to another.

TAILOR's user model at this point is coarse grained, in that it contains a list of objects that the user knows and whether he or she understands the basic underlying concepts. A more detailed model might include exactly which facts the user knows about objects. I have shown that a system benefits from a user model that indicates a user's knowledge about the domain, even when the model is not a detailed one.

While I feel that a detailed user model would be much harder to obtain, it would be interesting to see whether such a model would allow a system to provide more appropriate answers.

While this work was done only with respect to generating descriptions of complex devices, I think this approach will be useful in any information seeking environment to which users with different background and knowledge levels have access. Such environment could be a large knowledge base of facts (such as an encyclopedia), a help system, an expert system which needs to communicate with specialists, students and knowledge engineers, or simply with different types of users, such as the expert system for educational diagnosis proposed by [Cohen and Jones 87], and tutoring systems. Providing different information in an answer might also be done in explaining the behavior of an expert system which is used both as a teaching tool and as a problem solving engine.

The approach presented in this work would be readily applicable to any domain containing both functional and structural information. In some cases, however, the reverse of what is being done in TAILOR might be necessary: structural information might be more useful for naive and functional information for expert users. Consider for example the domain of *organizations*, where there are both constituents and functions. Providing the structure of the organization (e.g., "there is a president, three vice-presidents and five managers") might be sufficient for a relatively naive user, while information on who reports to whom (i.e., functionality) might be more appropriate for a person knowledgable about the organization.

For domains containing different kinds of information, there would be a need to identify which type of information is most appropriate for knowledgeable users and for more naive users. Tailoring to users with different knowledge about the domain by presenting different kinds of information will still appropriate.

8.3 Directions for future research

As TAILOR only handles requests for descriptions, an enhancement could be to extend this work to other types of questions in order to form a full question answering system. A more challenging task would be to study the effect of a user's level of expertise on the surface generation. In this work, I have been mainly concerned about the effect of a user's domain knowledge on the content of a description. It is likely that the choice of vocabulary and even syntactic structure could also be varied.

One extension would involve looking at other domains. I believe that varying the kind of information as opposed to only the level of detail will also be important in other domains. Determining what kinds of information are appropriate for which type of users in different domains needs to be identified.

Another extension to TAILOR is to use the system in conjunction with a graphics system. TAILOR determines the content of a description based on a user model. This description is by no means restricted to be expressed in English. Giving a description using a drawing might be more appropriate for some users than using words. It would be interesting to study how a system would decide on which media to choose to express the answer and how the two modes might highlight each other to provide a more expressive answer.

Determining the level of expertise remains an important problem. In Section 3.2, I have presented some factors that will help in inferring the user's level of domain knowledge. Implementing the ideas outlined in that section and studying the interactions among the various factors would be of great interest. Related to that issue, the task of updating the user model in a more sophisticated way than done by TAILOR is also a necessary and significant problem. This involves realizing when the user model was inferred incorrectly, repairing the error, and updating as the question answering session continues. Finally, it would be interesting to study how a system might realize that its strategies are not effective and how it might change its behavior. Imagine for example that users systematically ask a question twice. This would probably be a good clue as to the inappropriateness of the answer given, either in their content or in their phrasing. Determining this fact and being able act on it is likelv to be a very hard problem.

Finally, most generation systems that take a model of the user into consideration in constructing an answer are only concerned with one characteristic of the user. Similarly, TAILOR is only considering the user's domain knowledge. This work can be extended to use this parameter in conjunction with other user's characteristics, such as the goals, plans and beliefs. A system would probably improve its answering abilities by employing a user model comprising various aspects of the user. The interaction among these factors is probably very hard to study however.

8.4 Conclusions

In this work, I have demonstrated that the user's domain knowledge can be used as a factor in tailoring an answer. In particular, I have shown how the description of a complex physical object might be tailored to a user's level of expertise. I presented different kinds of knowledge users can have, explaining how a system can take them into consideration in order to generate a description. From my studies of texts, I have found two distinct discourse strategies that are used in describing complex devices. I postulated that the level of expertise of the user affects the *kind* of information given as opposed to just the *amount* of detail provided. Even though I conducted this study in the domain of complex physical objects, I believe this result can extend to other domains. I thus proposed that a user model containing information about the user's domain knowledge can be used in a question answering system to guide the decision process. I presented the two distinct descriptive strategies that can be used in a question answering program and showed how they can be mixed to include the appropriate information from the knowledge base, based on the information contained in user model.

Finally I presented TAILOR, a program that generates descriptions tailored to users with various levels of expertise. TAILOR employs one of the two discourse strategies described to generate a text for a novice or an expert. TAILOR is also able to automatically mix the strategies to provide device descriptions tailored to users whose domain knowledge fall anywhere along the knowledge spectrum. By representing explicitly the user's domain knowledge in terms of parameters, TAILOR does not require an *a priori* set of stereotypes but can provide wide variety of descriptions for a whole range of users.

Appendix A. Examples of texts studied

A.1 Texts from high school text books, junior encyclopedias and manual for novices

A Telegraph Sounder [Blackwood *et al.* 51]

By pressing the key, you close the electric circuit. Then, the electromagnet pulls down the iron plate, mounted on the pivoted bar.

An Electric Bell [Blackwood *et al.* 51]

The current magnetizes the U-magnet, which pulls over the iron plate and makes the clapper strike the bell. Then the circuit is cut at the contact breaker, the magnet releases the iron plate, and a spring pulls it back.

A Simple Telephone [Blackwood *et al.* 51]

By speaking into the mouthpiece you make the transmitter diaphragm vibrate and thus vary the pressure on the carbon grains. This varies their resistance, so that the current varies. The strength of the electromagnet varies, pulling the receiver diaphragm to and fro. It reproduces the sound.

A Model Commutator [Blackwood *et al.* 51]

As it rotates, each half ring is first negative, then positive. The current through the wire loop reverses twice for each revolution.

Incandescent Lamps [Coleman 11]

In an incandescent lamp, the current passes through a slender filament or wire which, owing to its high resistance, is heated white hot. The ends of the filament are attached to short platinum wires, which pass through the glass and connect with the metal casing, *d*, and plug, *g*, at the base of the lamp.

Thermometers [Reichlis and Lemon 52]

In a metallic thermometer - or dial type - heat expansion causes a coiled compound bar to twist. The twisting bar turns a pointer to indicate the correct temperature on a dial, which is calibrated (that is properly marked off).

Cathode Ray Tube [Verwiebe *et al.* 62]

This tube is highly evacuated. When a cathode ray (stream of electrons) leaves the cathode (negative electrode) and strikes the metal plate *A*, most of the electrons are stopped. Only those electrons passing through the slit will fall on plate *B*. Plate *B* is coated with a fluorescent material, and as the electrons strike plate *B*, a fluorescent light appears. When a magnet is brought near the tube, the electron beam is deflected just as a wire carrying a current near a magnet would be deflected. Both the beam and the wire are deflected in the direction indicated by the right-hand rule given for motors.

The Electron Microscope [Baker *et al.* 57]

Electrons are deflected by either electric or magnetic fields, and a stream of them can be focused by a suitable field of either kind just as light is focused by a lens.

Galvanometer [Britannica-Junior 63]

A galvanometer is an instrument for measuring an amount of current. If two magnets are brought together, they try to arrange themselves with their magnetic field lined up in the same direction. All galvanometers make use of this principle. In one type of galvanometer a coil of wire hangs between the poles of one or more permanent horseshoe magnets. The electromagnetic field is set up by a flow of electricity. The coil rotates (turns) to line up with the field of the permanent magnet. The stronger the current, the more the coil will turn. A mirror fastened to the wires reflects a small beam of light from a lamp onto a dial scale as the coil is twisted. Some galvanometers may read directly from a pointer on the scale. Another group of galvanometers uses a fixed coil of wire with a permanent magnet turned inside the coil. A pointer attached to the magnet shows the amount of rotation, in other words, the strength of the current.

Automobile Engine [Weissler 73]

An automobile engine produces power by burning a mixture of gasoline and air in a small space called a combustion chamber. When the mixture burns, it expands and pushes out in every direction. The combustion chamber is often located just above a cylinder, into which is installed a closely fitting plug called a piston. The piston is capable of being moved up and down in the cylinder. When the piston is lowered in a running engine, it creates a vacuum in the cylinder and draws in a mixture of fuel and air. The piston is then pushed up to the top of the cylinder, compressing the air-fuel charge, A spark ignites the mixture, which expands and pushed the piston downward.

A.2 Texts from adult encyclopedias and the manual for experts

General Description of the Engine [Chevrolet 78]

Starting at the front of the engine, cylinders in the left bank are numbered 1-3-5 and cylinders in the right bank are numbered 2-4-6.

The crankshaft, nodular cast-iron, is supported in the crankcase by four bearings. The number two bearing is the end thrust bearing.

The crankshaft is counterbalanced by weights cast integral with the crankshaft. Additional counterbalancing is obtained from a flex plate and harmonic balancer.

The tin plated alloy pistons have full skirts and are cam ground. Two transverse slots in the oil ring grooves extend through the piston wall and permit drain back of oil collected by the oil ring.

The camshaft is supported in the crankcase by five steel-backed babbitt-lined bearings. It is driven from the crankshafts by sprockets and chain.

The cylinder heads are cast-iron and incorporate integral valve stem guides and rocker arm shaft pedestral. Right and left cylinder heads are identical and interchangeable, although in service, it is good practice to reinstall the cylinder heads on the side from which they are removed.

The intake utilizes a low restriction, dual intake manifold. It is bolted to the inner edges of both cylinder heads so that it connects with all inlet ports. Since the intake manifold is cast-iron, as is the carburetor throttle body, the manifold incorporates a special exhaust heat passage to warm the throttle body...

Galvanometer [Collier's 62]

The commonest type of galvanometer is the moving-coil galvanometer. A permanent magnet creates a magnetic field in the air gap. The coil is supported by the suspension wire in such a manner that the coil can rotate in the air gap without touching any part of the magnet.

Galvanometer [Collier's 62]

A galvanometer of the D'Arsonval type is used to measure the magnitude of electric current. It consists of a needle attached to a coil suspended between the poles of a horseshoe magnet.

Telescope [Britannica 64] - Summary

Instrument for viewing or detecting distant and particularly extra-terrestrial objects, including optical telescopes, which combine lenses or mirrors or both to form a visible image, and radio telescopes which pick up and analyze radio-frequency electromagnet radiations.

Electric Motors [Britannica 64][25]

Motors must consists of two parts - a rotor and a stator. The rotor is usually round and is the moving portion. It contains conductors to establish and shape magnetic fields that will interact with other magnetic fields produced by the stator. The rotor also may have a contacting device (slip rings or commutator) to connect it electrically with the external circuit, together with a supporting shaft and bearings, an integral blower for cooling, etc. The stator consists of similar magnetic materials and electrical conductors to establish and shape the magnetic fields, as well as the frame to support the whole machine. The manner in which the conductors are arranged (and in which the magnetic fields are controlled) determines the type of motor and its characteristics.

Vacuum-tube Voltmeter [Collier's 62]

The vacuum-tube voltmeter employs an electron tube as a rectifier. In one form of vacuum-tube voltmeter, the two halves of a duplex diode rectifier tube are used, one half rectifying during one half of the cycle of the voltage being measured and the other rectifying during the other half of the cycle.

---

[25] There was also another section explaining *how* motors work.

Appendix B. Input/Output examples from TAILOR

User Model:

Local Expertise? *amplifier*

Basic Concepts? *nil*

>> *Describe radio-transmitter*

The radio-transmitter has an oscillator that produces a varying current when a battery produces another current, an amplifier that produces a strong varying audio-frequency signal from a varying audio-frequency signal, a mixer, and an amplifier to produce a strong radio-frequency signal from a radio-frequency signal. An amplifier has a vacuum-tube to produce a strong current from a power source and a metal grid. The oscillator has a high-resistance resistor, a battery, a capacitor, a low-resistance resistor and a transistor.

More? [oscillator even if it may involve basic concepts you don't understand?] *yes*

An oscillator is a device that produces a varying current, when a battery produces a current. Because the battery produces a current, the transistor of the oscillator turns on. That the transistor turns on is caused also by the capacitor of the oscillator discharging through the resistor of the oscillator. The transistor turning on causes the capacitor to charge. This causes the transistor to turn off. Because the transistor turns off, the capacitor discharges through the resistor. The resistor has low resistance. The capacitor discharging through the resistor causes current to be produced.

New User Model: (amplifier, radio-transmitter, oscillator)

>> *Describe microphone*

A microphone is a device that changes soundwaves into current. Because a person speaks into the microphone, soundwaves hit the diaphragm of the microphone. This causes the diaphragm to vibrate. When the intensity of the soundwaves increases, the diaphragm springs forward. This causes the granules of the button to be compressed. The compression of the granules causes the resistance of the granules to decrease. This causes the current to increase. Then, when the intensity decreases, the diaphragm springs backward. This causes the granules to be decompressed. The decompression of the granules causes the resistance to increase. This causes the current to decrease. The vibration of the diaphragm causes the current to vary. The current varies, like the intensity varies.

New User Model: (amplifier, radio-transmitter, oscillator, microphone)

>> *Describe telephone*

A telephone is a device that transmits soundwaves. Because a person speaks into the transmitter of the telephone, a current is produced. The transmitter is a microphone with a small diaphragm. The transmitter has a diaphragm and a system to broaden the response. The system has a cavity and a button. Then, the current flows through the line into a receiver. The current flowing through the line into the receiver causes soundwaves to be reproduced.

More? [Receiver] *yes*

A receiver is a loudspeaker with a small aluminium thin disc-shaped diaphragm. A loudspeaker is a device that changes current into soundwaves. The variation of a current causes the field of the magnet to vary. The magnet is ring-shaped and permanent. When the current increases, the field increases. Because the field increases, the poles of the magnet attract the diaphragm of the loudspeaker. This causes the diaphragm to spring forward. Then, when the current decreases, the field decreases. This causes the poles to release the diaphragm. Because the poles release the diaphragm, the diaphragm springs backward. The variation of the field causes the diaphragm to vibrate. When the diaphragm springs backward, the intensity of the soundwaves increases. Then, when the diaphragm springs forward the intensity decreases. The intensity varies, like the current varies.

New User Model: (amplifier, radio-transmitter, oscillator, microphone, telephone, receiver)

>> *Describe pulse-telephone*

A pulse-telephone is a telephone with a pulse-dialer. The pulse-telephone has a pulse-dialer that produces current pulses, when someone dials, a housing that has various shapes and various colors, a receiver to change current into soundwaves, a curly-shaped cord, a line and a transmitter that changes soundwaves into current. The pulse-dialer is a dialing-mechanism. Because a person dials, the dial-assembly of the pulse-dialer turns clockwise. This causes the spring of the pulse dialer to be compressed. The spring is circular. The compression of the spring enables the spring to be decompressed. The person dialing enables the person to release the dial of the dial-assembly. This causes the spring to be decompressed. The decompression of the spring causes the dial-assembly to turn counterclockwise. This causes the gear of the pulse-dialer to turn. The gear is small. The dial-assembly turns counterclockwise, proportionally to the way the gear turns. Because the gear turns, the protrusion of the gear hits the lever of the switch.

This causes the lever to close the switch. Because the lever closes the switch, current-pulses are produced. The receiver is a loudspeaker with a small metal thin disc-shaper diaphragm. A dialing-mechanism is connected to a wall by the line. The housing is connected to the dialing-mechanism by the cord. The housing contains the transmitter and it contains the receiver. The transmitter is a microphone with a small diaphragm.

New User Model: (amplifier, radio-transmitter, oscillator, microphone, telephone, receiver, pulse-dialer, pulse telephone)

References

[Allen and Perrault 80]
Allen, J. F. and Perrault, C. R.
Analyzing Intention in Utterances.
Artificial Intelligence, 15(1): pages 143 - 178, 1980.

[Anderson *et al.* 77]
Anderson, R.C., Spiro, R.J., and Anderson, M.C.
Schemata as scaffolding for the representation of information in connected discourse.
Technical Report 24, Center for the Study of Reading, Urbana, Illinois, 1977.

[Baker *et al.* 57] Baker, D.L., Brownlee, R.B. and Fuller R.W.
Elements of Physics.
Allyn and Bacon, Inc., 1957.

[Baker and Danyluk 86]
Baker, M. and Danyluk, A. P.
Representing Physical Devices.
Technical Report, Columbia University Department of Computer Science, 1986.

[Blackwood *et al.* 51]
Blackwood, O.H., Herron, W.B. and Kelly, W.C.
The High School Physics.
Ginn and Company, 1951.

[Britannica 64]
The New Encyclopedia Britannica .
1964
Encyclopedia Britannica Inc.

[Britannica-Junior 63]
Britannica Junior Encyclopedia .
1963
Encyclopedia Britannica Inc.; William Benton Publisher.

[Brown and Burton 78]
Brown, J. S. and Burton, R. R.
Diagnostics Models for Procedural Bugs in Basic Mathematical Skills.
Cognitive Science, 2 (2): pages 155 - 192, 1978.

[Carberry 83] Carberry, S.
Tracking User Goals in an Information-Seeking Environment.
In *Proceedings of AAAI-83*. American Association of Artificial Intelligence, 1983.

[Carberry 87] Carberry, S.
Plan Recognition and its Use in Understanding Dialogue.
User Models in Dialog Systems.
Springer Verlag, Symbolic Computation Series, Berlin Heidelberg New York Tokyo, 1987.

[Chemical 78]
Encyclopedia of Chemical Techology.
1978
Kirk and Othmer, Eds; John Wiley & Sons, New York, NY .

[Chevrolet 78] General Motors Corporation.
Chevrolet Service Manual.
1978

[Chi *et al* 81] Chi, M.T.H., Glaser, R. and Rees, E.
Expertise in Problem Solving.
Advances in the Psychology of Human Intelligence.
Lawrence Erlbaum, Hillsdale, 1981.

[Chin 86] Chin, D. N.
User Modelling in UC, the UNIX Consultant.
In *CHI'86 Proceedings.* Computer Human Interactions, 1986.

[Cohen and Jones 87]
Cohen, R. and Jones, M.
Incorporating User Models into Expert Systems for Educational Diagnosis.
User Models in Dialog Systems.
Springer Verlag, Symbolic Computation Series, Berlin Heidelberg New York Tokyo, 1987.

[Coleman 11] Coleman, S.E.
A Text-Book of Physics.
D.C. Heath & Company, Boston, New York, Chicago, 1911.

[Collier's 62]
Collier's Encyclopedia.
1962
The Crowell-Collier Publishing Company 1962; William Halsey editorial director.

[Cullingford *et al* 82]
Cullingford, R. E., Krueger, M. W., Selfridge, M. and Bienkowski, M. A.
Automated Construction of Classifications: Conceptual Clustering Versus Numerical Taxonomy.
IEEE Transactions on Systems, Man and Cybernetics, 12(2):168 - 181, 1982.

[Davison 84] Davison, A.
Readibility Formulas and Comprehension.
Comprehension Instruction; Perspectives and Suggestions.
Longman, New York and London, 1984.

[Egan and Gomez 82]
Egan, D. E. and Gomez, L. M.
Characteristics of People who can Learn to use Computer Text Editors: Hints for Future Text Editor Design and Training.
In *Proceedings of the ASIS Annual Meeting.* 1982.

[Egan and Gomez 83]
.
Assaying, Isolating and Accomodating Individual Differences in Learning a Complex Skill.
Individual Differences In Cognition.
Academic Press, New York, 1983.

[Encyclopedia of Science 82]
The New Encyclopedia of Science .
1982
Raintree Publishers.

[Fikes and Nilsson 71]
Fikes, R. E. and Nilsson, H. J.
STRIP: A New Approach to the Application of Theorem Proving to Problem Solving.
Artificial Intelligence, 2: pages 189 - 205, 1971.

[Grice 75] Grice, H. P.
Logic and Conversation.
In P. Cole and J. L. Morgan (editor), *Syntax and Semantics.* Academic Press, New York, 1975.

[Grimes 75] Grimes, J. E.
The Thread of Discourse.
Mouton, The Hague, Paris, 1975.

[Hayes and Reddy 79]
Hayes, P. and Reddy, R.
Graceful Interaction in Man-Machine Communication.
In *Proceedings of IJCAI-79.* International Joint Conference on Artificial Intelligence, 1979.

[Hobbs 78] Hobbs, J.
Why is a Discourse Coherent?.
Technical Report 176, SRI International, 1978.

[Hobbs 80] Hobbs, J. and Evans, D.
Conversation as Planned Behavior.
Cognitive Science, 4(4):349 - 377, 1980.

[Hoeppner *et al.* 84]
Hoeppner, W., Morik, K. and Marburger, H.
Talking it Over: The Natural Dialog System HAM-ANS..
Technical Report ANS-26, Research Unit for Information Science and Artificial Intelligence, University of Hamburg, 1984.

[Hovy 85]
Hovy, E.H.
Integrating Text Planning and Production in Generation.
In *Proceedings of IJCAI-85*. International Joint Conferences on Artificial Intelligence, Stanford, California, 1985.

[Hovy 87]
Hovy, E.H.
Some Pragmatics Decision Criteria in Generation.
In G. Kempen (editor), *Natural Language Generation: Recent Advances in Artificial Intelligence, Psychology, and Linguistics*. Kluwer Academic Publishers, Boston/Dordrecht, 1987.
Paper presented at the Third International Workshop on Natural Language Generation, August 1986, Nijmegen, The Netherlands.

[Jameson and Wahlster 82]
Jameson, A. and Wahlster, W.
User Modelling in Anaphora Generation: Ellipsis and Definite Description.
In *Proceedings of ECAI-82*, pages 222 - 227. ECAI, Orsay, France, 1982.

[Kaplan 79]
Kaplan, S. J.
Cooperative Responses from a Portable Natural Language Database Query System.
PhD thesis, University of Pennsylvania, Philadelphia, Pa, 1979.

[Kaplan 82]
Kaplan, S. J.
Cooperative Responses from a Portable Natural Language Query System.
Artificial Intelligence, 2(19)1982.

[Kay 79]
Kay, M.
Functional Grammar.
In *Proceedings of the 5th meeting of the Berkeley Linguistics Society*. Berkeley Linguistics Society, 1979.

[Kukich 85]
Kukich, K.
Explanation Structures in XSEL.
In *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Chicago, Illinois, 1985.

[Kwee 87] Kwee, TjoeLiong.
Natural Language Generation. One Individual Implementer's Experience.
In *Proceedings of the First European Workshop on Language Generation.* Language Generation Workshop, Royaumont, France, January, 1987.

[Lancaster and Kolodner 87]
Lancaster, J. S. and Kolodner, J. L.
Problem Solving in a Natural Task as a Function of Experience.
Technical Report, School of Information and Computer Science, Georgia Institute of Technology, Atlanta, Ga 30332, 1987.

[Lebowitz 83a] Lebowitz, M.
RESEARCHER: An Overview.
In *Proceedings of the Third National Conference on Artificial Intelligence.* American Association of Artificial Intelligence, Washington, DC, 1983.

[Lebowitz 83b] Lebowitz, M.
Generalization from Natural Language Text.
Cognitive Science, 7(1):1 - 40, 1983.

[Lebowitz 85] Lebowitz, M.
RESEARCHER: An experimental intelligent information system.
In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence,* pages 858 - 862. Los Angeles, 1985.

[Lehnert 77] Lehnert, W.
A Conceptual Theory of Question Answering.
In *Proceedings of IJCAI-77.* International Joint Conference of Artificial Intelligence, Cambridge, Massachusetts, 1977.

[Linde and Labov 75]
Linde, C. and Labov W.
Spatial Networks as a Site for the Study of Language and Thought.
Language, 57-4:924-939, 1975.

[Litman 86] Litman, D.
Linguistic Coherence: a Plan-based Alternative.
In *Proceedings of the 24th Annual Meeting of the ACL.* Association of Computational Linguistics, New York City, New York, June, 1986.

[Mann 84a] Mann, W.C.
Discourse Structures for Text Generation.
In *Proceedings of COLING '84.* Association for Computational Linguistics, Stanford, Ca., July, 1984.

[Mann 84b] Mann, W.C.
Discourse Structure for Text Generation.
Technical Report ISI/RR-84-127, Information Sciences Institute, February, 1984.
4676 Admiralty Way, Marina del Rey, California 90292-6695.

[Mays 80a] Mays, E.
Correcting Misconceptions about Data Base Structure.
In *Proceedings 3-CSCSI.* Canadian Society of Computational Studies of Intelligence, Victoria, B. C., May, 1980.

[Mays 80b] Mays, E.
Failures in Natural Language Systems: Applications to Data Base Query Systems.
In *Proceedings of the 1980 Conference on Artificial Intelligence.* American Association of Artificial Intelligence, Stanford, Ca, 1980.

[McCoy 86] McCoy, K. F.
The ROMPER System: Responding to Object-Related Misconceptions Using Perspective.
In *Proceedings of the 24th Annual Meeting of the ACL.* Association of Computational Linguistics, New York City, New York, June, 1986.

[McCoy 87] McCoy, K.F.
Reasoning on a Dynamically Highlighted User Model to Respond to Misconceptions.
Computational Linguistics Journal, Special Issue on User Modelling, 1987.

[McKeown 82] McKeown, K. R.
Generating Natural Language Text in Response to Questions About Database Structure.
PhD thesis, University of Pennsylvania, May, 1982.
Also a Technical report, No MS-CIS-82-05, University of Pennsylvania, 1982.

[McKeown 85] McKeown, K.R.
Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text.
Cambridge University Press, Cambridge, England, 1985.

[McKeown *et al.* 85]
McKeown, K. R., Wish, M. and Matthews, K.
Tailoring Explanations for the User.
In *Proceedings of IJCAI-85.* International Joint Conferences on Artificial Intelligence, 1985.

[McKeown and Paris 87]
McKeown, K. R. and Paris, C. L.
Functional Unification Grammar Revisited.
In *Proceedings of the 25th Annual Meeting of the ACL.* Association of Computational Linguistics, Palo Alto, California, 1987.

[Morik 85] Morik, K.
User Modelling, Dialog Structure, and Dialog Strategy in HAM-ANS.
In *Proceedings of EACL-85.* European Association of Computational Linguistics, Geneva, Switzerland, 1985.

[Morik 86] Morik, K.
Modeling the User's Wants.
Presented at the First International Workshop on User Modelling.
August, 1986

[Nessen 86] Nessen, E.
SCUM; User Modeling in the SINIX-Consultant.
Presented at the First International Workshop on User Modelling.
August, 1986

[New Book of Knowledge 67]
The New Book of Knowledge - The Children's Encyclopedia .
1967
Grolier Inc., New York.

[Paris 84] Paris, C. L.
Determining the Level of Expertise.
In *Proceedings of the First Annual Workshop on Theoretical Issues in Conceptual Information Processing.* Atlanta, Georgia, 1984.

[Paris and Kwee 85]
Paris, C. L. and Kwee, TjoeLiong.
Guide to the Unification Process and its Implementation; Progress Report on Extending the Grammar.
Technical Report, Computer Science Department, Columbia University, New York, NY 10027, 1985.

[Pollack 86] Pollack, M.
A Model of plan inference that distinguishes between the beliefs of actors and observers.
In *Proceedings of the 24th Annual Meeting of the ACL.* Association of Computational Linguistics, New York City, New York, June, 1986.

[Quilici 87] Quilici, A.
Detecting and Responding to Plan-Oriented Misconceptions.
User Models in Dialog Systems.
Springer Verlag, Symbolic Computation Series, Berlin Heidelberg New York Tokyo, 1987.

[Reichlis and Lemon 52]
Reichlis and Lemon.
Exploring Physics.
Harcourt, Brace and Company. Inc., 1952.

[Rich 79] Rich, E. A.
User Modeling Via Stereotypes.
Cognitive Science, 3:329 - 354, 1979.

[Ritchie 86] Ritchie, G.
The Computational Complexity of Sentence Derivation in Functional Unification Grammar.
In *Proceedings of COLING '86.* Association for Computational Linguistics, Bonn, West Germany, August, 1986.

[Rubinoff 86] Rubinoff, R.
Adapting MUMBLE: Experience with Natural Language Generation.
In *Proceedings of the Fifth Annual Conference on Artificial Intelligence.* American Association of Artificial Intelligence, 1986.

[Schank and Abelson 77]
Schank, R. C. and Abelson, R. P.
Scripts, Plans, Goals and Understanding.
Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1977.

[Shepherd 26] Shepherd, H. R.
The Fine Art of Writing.
Macmillan Co., New York, N.Y., 1926.

[Sidner 85] Sidner, C. L.
Plan Parsing for Intended Response Recognition in Discourse.
Computational Intelligence, 1985.

[Sidner and Israel 81]
Sidner, C. and Israel D.
Recognizing Intended Meaning and Speakers' Plans.
In *Proceedings of IJCAI-81.* International Joint Conference on Artificial Intelligence, August, 1981.

[Sleeman 82] Sleeman, D.
Inferring (Mal) Rules From Pupil's Protocols.
In *Proceedings of ECAI-82.* European Conference (?) of Artificial Intelligence, Orsay, France, 1982.

[Sleeman 85] Sleeman, D.
UMFE: A User Modelling Front-End Subsystem.
International Journal of Man-machine Studies, 23: pages 71 - 88, 1985.

[Stevens *et al.* 79] Stevens, A. , Collins, A. Goldin, S. E.
Misconceptions in Student's Understanding.
International Journal of Man-machine Studies, 11: pages 145 - 156, 1979.

[Tennant 78] Tennant H.
The Evaluation of Natural Language Question Answerers.
Technical Report, University of Illinois at Urbana/Champaign, 1978.
Ph.D. Proposal, Department of Computer Science, Advanced Automation Group, Coordinated Science Laboratory.

[The Way Things Work 72]
The Way Things Work.
1972
Simon and Schuster, New York, N.Y.

[van Beek 87] van Beek, P.
A Model for Generating Better Explanations.
In *Proceedings of the 25th Annual Meeting of the ACL.* Association of Computational Linguistics, Palo Alto, California, 1987.

[Verwiebe *et al.* 62]
Verwiebe, F.L., Van Hooft, G.R. and Suchy R.R.
Physics; A Basic Science.
D. van Nostrand Company, Inc., Princeton, New Jersey, 1962.

[Wallis and Shortliffe 82]
Wallis, J.W. and Shortliffe, E.H.
Explanatory Power for Medical Expert Systems: Studies in the Representation of Causal Relationships for Clinical Consultation.
Technical Report STAN-CS-82-923, Stanford University, 1982.
Heuristics programming Project. Department of Medecine and Computer Science.

[Wasserman 85] Wasserman, K.
Unifying Representation and Generalization: Understanding Hierarchically Structured Objects.
PhD thesis, Columbia University Department of Computer Science, 1985.

[Wasserman and Lebowitz 83]
Wasserman, K. and Lebowitz, M.
Representing Complex Physical Objects.
Cognition and Brain Theory, 6(3):333 - 352, 1983.

[Weiner 80] Weiner, J.
BLAH, a System that Explains its Reasoning.
Artificial Intelligence Journal, 15:19 - 48, 1980.

[Weissler 73] Weissler A. and Weissler P.
A woman's guide to fixing the car.
Walker and Company, New York, 1973.

[Wilensky *et al.* 84]
Wilensky, R., Arens, Y. and Chin, S. N.
Talking to UNIX in English: An Overview of UC.
Communications of the ACM, 27(6)June, 1984.

[Wilson and Anderson 86]
Wilson, P. T. and Anderson, R. C.
What They Don't Know Will Hurt Them: The Role of Prior Kowledge in Comprehension.
Reading Comprehension: From research to Practice.
Erlbaum, Hillsdale, New Jersey, 1986.

[Woods 73] Woods, W.
An Experimental Parsing System for Transition Network Grammars.
In Rustin, R. (editor), *Natural Language Processing.* Algorithmics Press, New York, 1973.