# *Technical Report*

UCAM-CL-TR-239
ISSN 1476-2986

Number 239

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

# Planning multisentential English text using communicative acts

Mark Thomas Maybury

December 1991

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

*http://www.cl.cam.ac.uk/*

This technical report is based on a dissertation submitted July 1991 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Wolfson College.

Technical reports published by the University of Cambridge Computer Laboratory are freely available via the Internet:

*http://www.cl.cam.ac.uk/techreports/*

ISSN 1476-2986

## Summary

The goal of this research is to develop explanation presentation mechanisms for knowledge based systems which enable them to define domain terminology and concepts, narrate events, elucidate plans, processes, or propositions and argue to support a claim or advocate action. This requires the development of devices which select, structure, order and then linguistically realize explanation content as coherent and cohesive English text.

With the goal of identifying generic explanation presentation strategies, a wide range of naturally occurring texts were analyzed with respect to their communicative structure, function, content and intended effects on the reader. This motivated an integrated theory of communicative acts which characterizes text at the level of rhetorical acts (e.g., describe, define, narrate), illocutionary acts (e.g., inform, request), and locutionary acts (e.g., ask, command). Taken as a whole, the identified communicative acts characterize the structure, content and intended effects of four types of text: description, narration, exposition, argument. These text types have distinct effects such as getting the reader to know about entities, to know about events, to understand plans, processes, or propositions, or to believe propositions or want to perform actions. In addition to identifying the communicative function and effect of text at multiple levels of abstraction, this dissertation details a tripartite theory of focus of attention (discourse focus, temporal focus, and spatial focus) which constrains the planning and linguistic realization of text.

To test the integrated theory of communicative acts and tripartite theory of focus of attention, a text generation system TEXPLAN (Textual EXplanation PLANner) was implemented that plans and linguistically realizes multisentential and multiparagraph explanations from knowledge based systems. The communicative acts identified during text analysis were formalized as over sixty compositional and (in some cases) recursive plan operators in the library of a hierarchical planner. Discourse, temporal, and spatial focus models were implemented to track and use attentional information to guide the organization and realization of text. Because the plan operators distinguish between the communicative function (e.g., argue for a proposition) and the expected effect (e.g., the reader believes the proposition) of communicative acts, the system is able to construct a discourse model of the structure and function of its textual responses as well as a user model of the expected effects of its responses on the reader's knowledge, beliefs, and desires. The system uses both the discourse model and user model to guide subsequent utterances. To test its generality, the system was interfaced to a variety of domain applications including a neuropsychological diagnosis system, a mission planning system, and a knowledge based mission simulator. The system produces descriptions, narrations, expositions, and arguments from these applications, thus exhibiting a broader range of rhetorical coverage than previous text generation systems.

## Preface

I have been fortunate to learn under the guidance of Dr. Karen Sparck Jones, whose wisdom and encouragement have been a continual source of insight and motivation. I thank Dr. Karen Sparck Jones and Dr. Roger Needham for their support of my transatlantic research arrangement, without which this dissertation would never have been possible. To the entire *Computer Speech and Language Processing* M Phil staff (Frank Fallside, Karen Sparck Jones, Steve Pulman, Steve Young, Mavis Barber, and others) for instilling a sound appreciation of the complexities of speech and language which served as a rich foundation for my research. I thank the Rotary Foundation International and the Air Force Institute of Technology (especially Lt. Col. Parsons) for funding my first year at Cambridge. I am grateful also to my Cambridge research colleagues and Wolfson College friends whose humor and kindness tempered my Cambridge studies.

I acknowledge the generous support of the Rome Air Development Center, in particular that of Col. Rudolph Zuberbuhler and Raymond Urtz. I have enjoyed working under the management of my good friend, Robert Ruberti. I thank the RADC researchers at the Knowledge Engineering Branch (especially the natural language group and the LACE team) for many enlightening technical discussions. I thank Ann Crescenzi of the RADC Technical Library who tracked down numerous, often obscure references. Finally, thanks to the many researchers who selflessly and patiently commented on dissertation drafts and discussed technical issues including Karen Sparck Jones, John Levine, David McDonald, Marie Meteer, Eduard Hovy, John Bateman, Doug Appelt, Dan Suthers, Cecile Paris, Johanna Moore, Bill Swartout, Bob Kass, Alison Cawsey, Ehud Reiter, and others.

To my parents and family, whose love encouraged me continually. Last, and certainly most important, I am deeply indebted to Michelle, whose indefatigable support, kindness, and love made this all possible. To her I dedicate this work.

## Statement of Originality

This dissertation is the result of my own work and is not the outcome of any work done in collaboration. I hereby state that this dissertation is not substantially the same as any I have submitted for a degree, diploma, or other qualification at any other university. Some application systems written by others were exploited for experimental purposes. This is clearly indicated in the dissertation. Furthermore, Chapter 8, in part, details research I performed for the M. Phil. degree in Computer Speech and Language Processing, a year which the Board of Graduate Studies has approved as counting toward the Ph. D. Other than parts of Chapter 8, no part of this dissertation has already been or is being concurrently submitted for any such degree, diploma or other qualification.

# Contents

# Figures

# Chapter 1

# INTRODUCTION

## 1.1 Problem and Aim

Computational systems that interact with humans often need to define their terminology, elucidate their behavior, or support their recommendations or conclusions. In general, they need to explain themselves. Explanations include descriptions of domain concepts and entities, narrations of events, expositions of plans, processes, or propositions, and finally, arguments which support a claim or advocate action. Enhancing the *representation* of explanations in knowledge-based systems has been the focus of intense research in artificial intelligence (Winograd, 1972; Clancey, 1983; Hasling et al., 1983; Swartout, 1977, 1981ab, 1983ab; Neches et al., 1985). In contrast, this dissertation focuses on the *presentation* of explanations, in particular the generation of multisentential natural language (as opposed to multi-media) explanations.

Natural language generation can be broadly divided into strategic and tactical stages (McKeown, 1982). The former concerns the selection, structure, and order of explanation content, termed *text planning*, and the latter entails the *linguistic realization* of that content as English. Knowledge based applications in a variety of generic tasks (e.g., diagnosis, simulation, or planning), even if they have rich representations of explanations, require mechanisms to plan and linguistically realize explanations in order to produce output that reflects Grice's (1975) maxims of quantity, quality, relation, and manner. Thus the practical aim of this work is to develop computational mechanisms which plan and linguistically realize textual explanations of domain application concepts, methods, plans, recommendations and conclusions.

As explanations are often presented via multisentential text, the above practical goal gives rise to a theoretical aim. Research in natural language generation has identified several computational linguistic and textual problems. These include:

1. How is text organized above the sentence?
2. What is the relationship of focus to content selection, ordering, and realization?
3. How does the structure and focus of text affect surface form?
4. What is the relation of communicative intentions to text structure and surface form?

5. What effects can texts be designed to have on an addressee?
6. How do these general issues in language generation affect explanation representation?

Therefore the theoretical aim of this dissertation is investigate the hypothesis that the generation of multiple utterances, as with planning single utterances (Appelt, 1985), is a plan-based activity that is based on communicative acts. This requires the analysis of text in search for underlying communicative acts that achieve distinct effects on the hearer's knowledge, beliefs, and desires. Thus this dissertation investigates the communicative structure and communicative function of a range of explanations for knowledge based systems. Related to this is the issue of how focus constrains generation (Sidner, 1979; McKeown, 1982) and so a second theoretical aim is to investigate how attentional constraints relate to text planning and linguistic realization.

## 1.2 Research Methodology

As with previous computational investigations of natural language generation (e.g., Weiner, 1980; McKeown, 1982; McCoy, 1985ab; Paris, 1987ab)[1], the starting point of the computational theory was the examination of human produced explanations in an attempt to identify underlying communicative strategies and constraints on explanation generation. During text analysis I attempted to identify the communicative elements of explanatory text, the communicative function those elements serve in the text (i.e., their effect on the hearer), and associated focal constraints (as in McKeown, 1982).[2] This began with analysis of text in terms of rhetorical predicates (Grimes, 1975; McKeown, 1985; Paris, 1987ab) as well as the locutionary and illocutionary function of utterances (Searle, 1969; Appelt, 1982). Finally, an attempt was made to identify communicative acts which characterize groups of illocutionary acts over segments of text or texts as a whole. These were termed *rhetorical acts* such as describe, compare, narrate and argue. The associations among individual communicative acts (e.g., subordination, ordering, and grouping) were considered as well as their function (i.e., intended effect) as individual acts and as larger collections of acts. Since it was important to examine a broad range of texts in a variety of domains to ensure a broad sampling of data, writing textbooks (e.g., Kane and Peters, 1986; Picket and Laster, 1988), rhetoric texts (e.g., Brown and Zoellner, 1968; Brooks and Hubbard, 1905) as well as general sources such as encyclopedias and advertisements were examined.

[1]Hovy (1988) and Moore (1989) have based their work on Rhetorical Structure Theory (Mann and Thompson, 1987) which is based on rhetorical analysis of naturally occurring texts.
[2]As text analysis is a subjective endeavor, this dissertation makes no claims of psychological adequacy, but simply indicates that the communicative strategies are motivated by what humans produce.

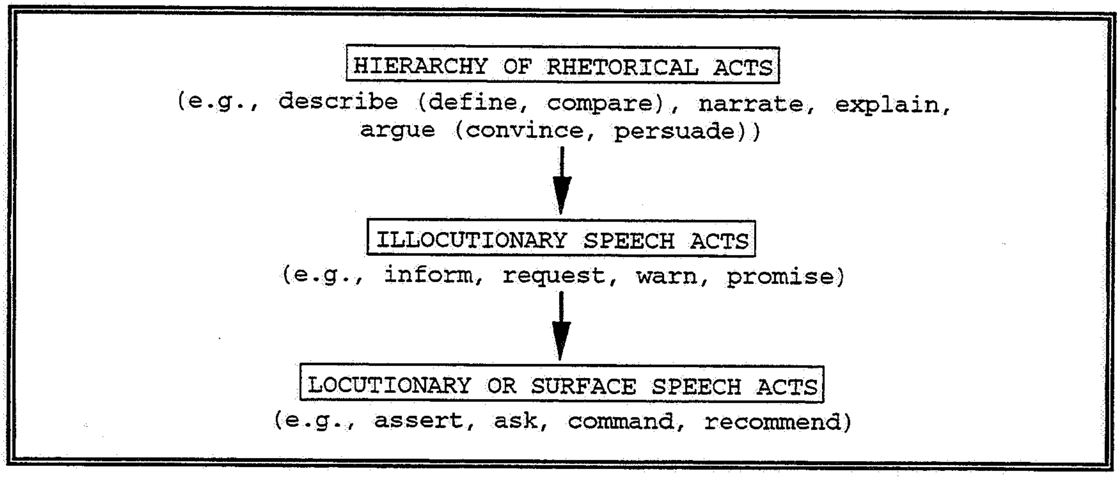



Figure 1.1 Integrated Theory of Communicative Acts

Motivated by the text analysis, this dissertation proposes an integrated theory of *communicative acts* shown in Figure 1.1 which characterizes a text in terms of *rhetorical acts*, *illocutionary acts* (Austin, 1962; Searle, 1969; Cohen, 1978; Allen, 1979), and *surface speech acts* (Appelt, 1985). Just as Grosz and Sidner (1986) argue that discourses have purposes and particular discourse segments have purposes, so too this dissertation argues that texts in and of themselves can be decomposed into individual communicative acts which are aimed at achieving particular effects on the addressee. The primary focus of this work is to define the top level communicative acts, i.e., a range of hierarchical rhetorical acts (e.g., describe, define, narrate) which characterize four major types of text: *description*, *narration*, *exposition*, and *argument* (these constitute the four principal chapters of this dissertation). In any particular piece of literature, however, many of these types of text may be employed, often intermixed. In addition, analysis of the kinds of text investigated in this dissertation (e.g., reports, directions) identified three distinct notions of attention which constrain the order and realization of content: discourse focus (Sidner, 1979, adapted for generation by McKeown, 1982), temporal focus (Webber, 1988a), and spatial focus. We distinguish between the local focus of attention of individual utterances and the topic or subject of multiple utterances. Once again, actual texts may utilize a number of additional reference points such as the speaker, the audience, the mode of communication (e.g., text, speech, smoke signals), the genre, etc. (see Lewis, 1972). Both communicative acts and focus models were formalized and tested computationally in the system, TEXPLAN (Textual EXplanation PLANner), which the next section overviews.

## 1.3 System Overview

Figure 1.2 illustrates a schematic overview of TEXPLAN, which can be divided into two basic processes: text planning and linguistic realization. The *text planner*, takes as input a discourse goal and selects, structures, and orders content from some underlying application. The *linguistic realizer* translates the resulting text plan onto English surface form. Text planning and linguistic realization can be serial or interleaved (as described in Chapter 8) and so the text planner can plan an entire text which is then realized or it can plan and realize a text utterance by utterance, allowing failure or success of utterance realization to guide text planning. Each communicative act (e.g., rhetorical, illocutionary, or locutionary) is formalized as a plan operator with preconditions, constraints, effects, and decomposition and appears in the plan library of a hierarchical planner (Sacerdoti, 1973, 1977) (the plan language is detailed in Chapter 4). TEXPLAN has over sixty domain-independent rhetorical plan operators (22 descriptive, 16 narrative, 10 expository, and 15 argumentative plan operators) detailed in Chapters 4-7. There are six illocutionary plan operators for the four illocutionary acts of inform, request, warn, and concede (these plans include `inform-by-assertion`, `request-by-asking`, `request-by-commanding`, `request-by-recommendation`, `warn-by-exclamation` and `concede-by-assertion`). Finally, there are five locutionary acts (`assert`, `ask`, `command`, `recommend`, and `exclaim`) which have associated propositional content and correspond to surface forms such as declarative, interrogative and imperative sentences.

Planning is initiated when the system, as speaker, or the user, as hearer, posts a *discourse goal* to a simple dialogue manager. A discourse goal is expressed in terms of some desired effect on the user's knowledge, beliefs, or desires. Given this discourse goal, the text planning component of the system searches the communicative plan library (including rhetorical, illocutionary, and locutionary plans) for high-level rhetorical acts which can accomplish the intended goal. These acts are then decomposed into other rhetorical acts and eventually into illocutionary acts which decompose into locutionary acts which have associated *rhetorical propositions.*

Rhetorical propositions are rhetorical predicates instantiated with information from the underlying domain knowledge base and are similar to those used by McKeown (1982) and Paris (1987ab). However the types of text produced by TEXPLAN (including description, narration, exposition, and argument) require a broader range of rhetorical predicates so these include not only predicates such as logical-definition, attribution, and cause but also predicates such as evidence, enablement, and motivation (the 21 predicate types are detailed in Chapter 8). Thus the text planner selects and orders *communicative plans* which structure propositional content, guided by the given discourse goal, the previous discourse context (i.e., previously uttered communicative plans and propositional content), global focus caches, and a model of the user, which is updated with the effects of the current text if it is successful. The result of text planning is a hierarchical text plan that includes a communicative plan decomposition (with failed options and untried alternatives) as well as an effect decomposition that captures the expected effect of that text on

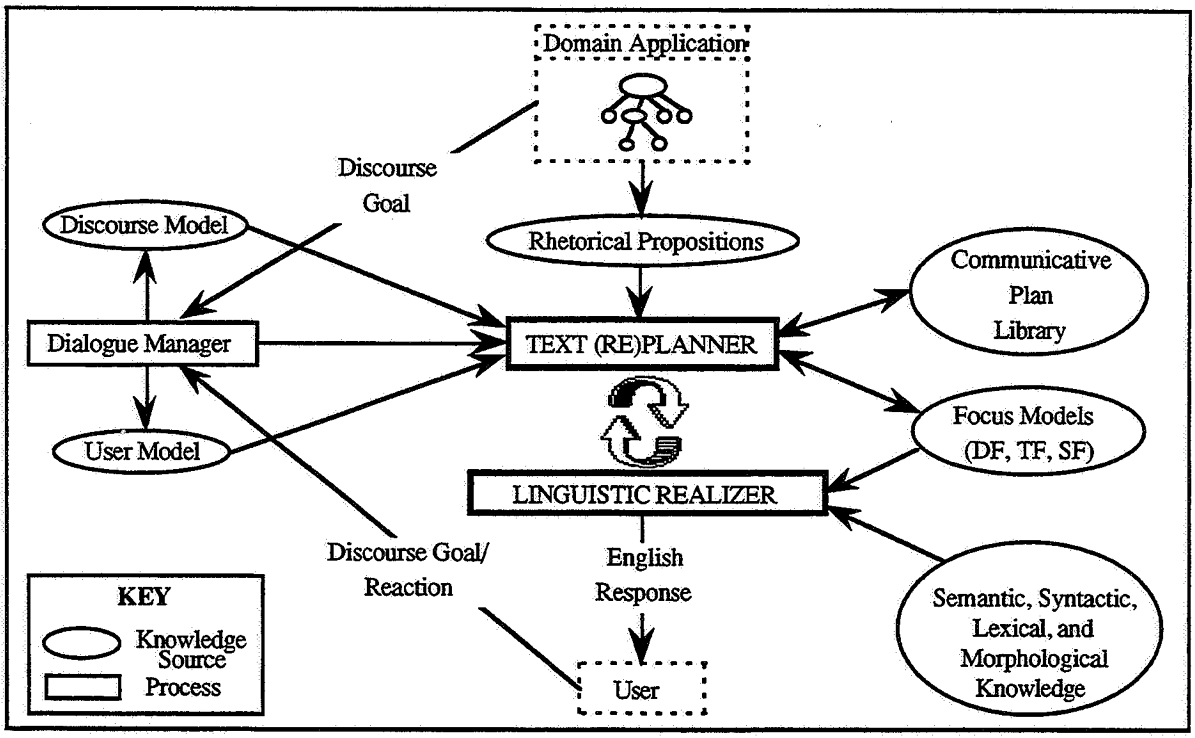



Figure 1.2 TEXPLAN System Overview
[The Domain Application is variable]

the knowledge, beliefs, and desires of the hearer. This text plan along with the discourse goal that motivated it are recorded in the discourse model, a stack of previous discourse goals and associated text plans.

After text planning, the communicative plan is realized as English text using linguistic knowledge as well as focus constraints (discourse, temporal, and spatial) as detailed in Chapter 8. After a text is realized, the user has an opportunity to accept the text or to indicate their reaction in a number of canned ways (e.g., understand, confused, disbelief, understand but not convinced) which signals to a simple dialogue manager to either update the user model with the expected effects of the previous text, replan a new text that achieves the previous discourse goal, post a new discourse goal to the text planner, or give up.

As Figure 1.2 suggests, TEXPLAN was tested in a variety of domains in systems that addressed a range of generic problem solving strategies such as the Knowledge Replanning System (KRS) (Dawson et al., 1987), the knowledge based simulator, Land Air Combat in ERIC (LACE) (Anken, 1989), a cartographic Map Display System (Hilton, 1987), and the NEUROPSYCHOLOGIST medical diagnosis system (Maybury and Weiss, 1987). The motivation for multi-application testing was not only to ensure domain-independence of the communicative plans but more importantly because most current applications either have limited knowledge bases that support only a restricted class of texts (e.g., description or narration) or the nature of their task requires only a subset of these text types. For example, the entity-relationship models underlying most database applications support description. The event representations underlying simulations enable narration. In contrast, most advisory systems can support deductive and inductive argument as well as description but there are usually no events to narrate. Exposition, in contrast to description, narration, and argument, requires knowledge of plans, processes, and propositions. The need for multiple, varied domains to test the generation of a range of text types is reflected in the sample responses illustrated in the next section.

## 1.4 Sample Text Output

To illustrate TEXPLAN's ability to generate description, narration, exposition and argument, this section exemplifies several texts from various application systems. The first type of text, description, is illustrated by the text below from the Knowledge Replanning System (KRS) (Dawson et al., 1987). The following multisentential description of a domain entity is given in response to a user's query "What is a target?" (simulated by posting the discourse goal "get the hearer to know about the class, `#<target>`" to TEXPLAN's discourse controller).

```
USER:     What is a target?
TEXPLAN:  Targets are entities.  They have a latitude/longitude, a cloud
cover, a cloud height, a visibility, and a weather condition.  There are
five targets: passages, facilities, electronic hardware, weapons, and
vehicles.  Weapons, for example, are targets such as anti-aircraft
missiles, surface-to-surface missiles, and enemy aircraft.
```

The proper realization of the text demands the tracking of *discourse focus*, used for example to guide pronominalization (e.g., the use of "they" in sentence two of the response to refer to "targets") or to guide grammatical structure (e.g., voice selection, there-insertion, or it-extraposition). Connectives such as "for example" increase the text's cohesiveness and are produced by the linguistic realization component to signal the underlying communicative structure of the text. While the above text is very similar to those produced by McKeown's (1982) constituency schema, the text plan underlying this text captures both the communicative structure and function of the text as well as its expected effect on knowledge of the user.

In contrast to the above description, the second type of text is event narration. For example, the following is a short 'biography' detailing what happened to a mission simulated by the knowledge based simulation system, LACE (Anken, 1989). As in the above description, the query, "What happened ...?", is simulated by sending the discourse controller the goal "get the hearer to know about the events concerning the mission `#<OCA101>`":

```
USER:     What happened with Offensive Counter Air Mission 101?
TEXPLAN: Offensive Counter Air Mission 101 was an air strike against
Delta airfield.  It began mission execution at 8:41::40 Tuesday December
2, 1987.  It received four aircraft from the 900TFW-F-4c.  Seven minutes
later it was flying its ingress route.  Then ten minutes later it bombed
its target.  It began flying its egress route.  Thirty-six minutes later
it ended its mission.  It generated its post-mission report.
```

The proper realization of this narration requires tracking not only discourse focus (e.g., for pronominalization), but also the event currently focused on in time (called the *temporal focus* following (Webber, 1988a)). Tracking temporal focus is necessary in order to determine appropriate verb tense and aspect as well as to realize temporal adverbials such as "seven minutes later". Chapter 5 details narration and the use of temporal focus.

In contrast to the above example of narration, a third generic type of text is exposition, exemplified in part by the text below from the Map Display System (Hilton, 1987), a knowledge based cartographic system which includes a route planner. The user's query is simulated by sending the discourse controller the intended effect of the text, get the user to know how to go from As to Merseburg.

```
USER:     How do I get from As to Merseburg?
TEXPLAN: From As take Route 21 Southeast five kilometers to the
intersection of Route 92 and Route 21.  At that intersection take Route
92 Northwest thirty-six kilometers to Oelsnitz.  From Oelsnitz continue
on Route 92 Northwest thirty-eight kilometers to the intersection of
Route 175 and Route 92.  From that intersection take Route 2 Northeast
twenty-eight kilometers to the intersection of Route 2 and Route 176.
From there take Route 176 Northwest thirty-nine kilometers to the
intersection of Route 91 and Route 176.  At that intersection take Route
91 Northeast sixteen kilometers to Merseburg.  Merseburg is a town
located at 51° latitude and 12° longitude one hundred thirty-two
kilometers Northwest of town As.
```

The proper realization of this example of exposition, termed locational instruction, requires tracking not only of discourse and temporal focus as in the above examples of description and narration, but also the current focus of attention in space, termed *spatial focus*. As detailed in Chapter 6, spatial focus guides the realization of spatial anaphora (e.g., "`here`" versus "`there`"), demonstratives (e.g., "`this`" versus "`that`"), and the realization of durative and directional adverbials (e.g., "`Northwest thirty-six kilometers`"). In contrast to locative instruction, Chapter 6 also details other forms of expository text such as operational instructions, which tell the user how to perform a task and process instructions, which indicate how something functions.

In addition to description, narration, and exposition, the final type of text addressed by this dissertation is argument (and persuasion). Argument texts can be functionally divided into those that attempt to convince the user to believe a proposition and those that attempt to persuade them to act. For example, the following text from the medical diagnosis system NEUROPSYCHOLOGIST (Maybury and

Weiss, 1987), attempts to convince the reader of a claim by providing evidence. The user's query is simulated by posting to the discourse controller the goal, get the user to believe that `Patient1` has Korsakoff's disorder.

```
USER:     Why have you concluded that Patient1 has Korsakoff's disorder?
TEXPLAN: Patient1 has Korsakoff's disorder with 75% probability.  An
apathetic demeanor indicates a 70% probability of Korsakoff's disorder.  A
poor memory and low IQ scores indicate a 80% probability of Korsakoff's
disorder.
```

The text supports the claim with evidence ordered from least certain to most certain. While the above argumentative text aims to convince the reader to believe a claim, others aim to evoke action. The text below, produced for the Knowledge Replanning System (Dawson et al., 1987), urges the user to act by showing the motivation for doing so (i.e., recovering from an error state) and elucidating the causes of the problem.

```
USER:     Why should I replace F-111e aircraft with F-4g aircraft in
Offensive Counter Air Mission 1002?
TEXPLAN:  You should replace F-111e aircraft with F-4g aircraft in
Offensive Counter Air Mission 1002.  A conflict between the aircraft and
the target in Offensive Counter Air Mission 1002 motivates replacing F-
111E aircraft with F-4g aircraft.  You inserted Ludwigslusts-Alpha in
the target slot and Ludwigslusts-Alpha was radiating which caused a
conflict between the aircraft and the target in Offensive Counter Air
Mission 1002.
```

In the above response, TEXPLAN first requests the user to perform an action, next shows what motivates the action, and finally indicates the multiple causes of the state which motivated the action. Argumentative texts, both those that support claims and those that promote action, are detailed in Chapter 7. A final chapter, 8, details the linguistic realization of these four types of text: description, narration, exposition, and argument.

## 1.5 Dissertation Scope

This dissertation concentrates on the use of communicative acts and focus constraints to present coherent and cohesive textual explanations. This work is not concerned with scientific explanation or explanation based learning (e.g., Schank, 1986) which investigates cognitive simulations that provide insight into memory search and the reorganization of knowledge structures. Furthermore, while the communicative plans detailed in this dissertation were motivated by analysis of human produced text, this research focuses on engineering rather than cognitive modelling.

This work does not address enhancing the content of explanations (e.g., Swartout, 1977, 1981ab, Clancey, 1983; Neches, 1985) nor does this research address the interpretation of language or classification of explanation questions (although the model presented in this dissertation does represent the intended effect of a system response, which could be associated with question classes.) Therefore, the dissertation assumes as a starting point a communicative goal that the underlying application or the user has posted to TEXPLAN's discourse controller as to what effect to attempt on the user's knowledge, beliefs, or desires (e.g., achieve the state that the hearer believes P). Furthermore, this work focuses on multisentential text and does not address question-answering as in database query (e.g., yes/no and wh-queries such as "How many employees earn more than their bosses?").

In addition, this research does not focus on the construction or maintenance of detailed models of the individual user (e.g., Wilensky et al., 1988; Kass and Finin, 1988; Carberry, 1988). It does, however, suggest how different text types can potentially effect the knowledge, beliefs, and desires of the user at all levels of the text (i.e., rhetorical, illocutionary, and locutionary).

This work makes no claims concerning modelling explanatory dialogues (Cawsey, 1989, 1990; Wolz, 1990), clarification subdialogues (Litman and Allen, 1987), follow-up questions (Moore, 1989), or interruptions. For testing purposes, however, a number of reaction classes were formulated in order to demonstrate alternative explanation strategies in response to similar queries. Reaction classes were also used to illustrate how context (i.e., the discourse and user model) was modified after uttering text and how this could be used to guide subsequent responses.

Finally, this work does not address tailoring responses rhetorically (Paris, 1987ab), stylistically (Hovy, 1987), or to perspective (McKeown et al., 1985) on the basis of models of the user. This research does not aim at developing miscommunication recovery mechanisms such as those which address user's false presuppositions (Kaplan, 1982), misconceptions about domain entities (McCoy, 1985ab), or ill-formed plans (Joshi, et al., 1984; Pollack, 1986; Quilici, 1988).

## 1.6 Novelty and Contribution

The principal contributions of this dissertation concern the integrated theory of communicative acts, a range of communicative plans which characterize several text types, and the association of different types of focus with different classes of text. While the principal focus of this work is on text planning, TEXPLAN includes a linguistic realization component which operates either serially or interleaved with the text planner and has a unique representation that includes a relational grammar that maps case semantics onto a phrase structure grammar.

Communicative acts have been investigated in the past, initially with respect to the illocutionary speech acts such as inform and request which underlie single utterances (Cohen, 1978; Allen, 1979). The notion of language as a planned behavior dates to Austin (1962), direct and indirect illocutionary acts to Searle (1969, 1975), and plan-based models of speech acts to (Bruce, 1975). Appelt (1985) investigated the generation of two types of speech acts: illocutionary speech acts (inform and request) and surface speech acts (assert, command, and ask) (as well as propositional acts and utterance acts, detailed in Chapter 3). Grosz and Sidner (1986) investigated the relationship between intentional structure and the discourse segmentation. In contrast, this dissertation argues for a more refined, tripartite representation of communicative acts: rhetorical acts (e.g., describe, define, narrate), illocutionary acts (e.g., inform, request, warn), and locutionary acts (e.g., assert, command) which are used to plan multisentential text.

In contrast to recent computational implementations which structure propositions (Hovy, 1988a) or illocutionary actions (Moore, 1989) using rhetorical relations based on Rhetorical Structure Theory (Mann and Thompson, 1987), TEXPLAN's plan operators (which represent rhetorical, illocutionary, or locutionary acts) construct both communicative action decompositions and effect decompositions for a range of text types including description, narration, exposition, and argument. A communicative plan decomposition represents the discourse model of the text (and embodies the text structure), whereas the effect decomposition represents the expected consequences of each action in the plan decomposition on the user's knowledge, beliefs or desires (i.e., what the text plan contributes to the user model once it is executed, that is linguistically realized). Section 1.4 illustrates the range of text types produced by TEXPLAN and suggests that text planners that aim to produce a broad range of explanations cannot be effectively tested in the context of only one application task (e.g., diagnosis, simulation or planning), because this restricts their rhetorical range to a subset of description, narration, exposition or argument.

Finally, this dissertation examines how different types of local focus can constrain text planning and realization. McKeown (1982) was the first to suggest using local focus (Sidner, 1979) and global discourse focus (Grosz, 1977) to constrain the selection, ordering, and realization of explanation content. Hovy and McCoy (1989) later explored how discourse Focus Trees (McCoy and Cheng, 1991) could constrain choice in planning systems. In contrast to this previous work, TEXPLAN represents three distinct types of focus (discourse, temporal, and spatial) which are associated with different text types (description, narration, and exposition) and can effect the order and realization of text content. However, as far as temporal focus, tense and aspect are concerned, this is an active area of research in philosophy (Dowty, 1986), linguistics (Tedeschi and Zaenen, 1981) and computational linguistics (Allen, 1988) and this dissertation makes no claims regarding novelty of its temporal, tense, or aspectual representations. It simply indicates that tense and aspectual information, like intentional or attentional constraints, should be used to guide the selection and realization of events and states.

This work is thus novel in several respects. First, it contributes an integrated theory of communicative acts: rhetorical, illocutionary, and locutionary. Second, it examines a broader range of generic text classes than past systems including description, narration, exposition, and argument. These texts are characterized both in terms of their communicative structure and function, in particular their effects on the hearer's knowledge, beliefs and desires. Finally, this work considers three distinct types of focus -- discourse, temporal, and spatial -- and how they constrain linguistic realization.

## 1.7 Dissertation Organization

The remainder of this dissertation is structured as follows. First Chapter 2 critically examines past work in explanation and natural language generation. Chapter 3 then considers past and recent plan-based approaches to explanation. Chapters 4 through 7 constitute the core of the dissertation and are organized around four generic types of text -- description, narration, exposition, and argument -- each of which have distinct effects on the hearer/reader. Chapter 4 focuses on *description*, whose purpose is to get the hearer/reader to know about some entity. In contrast, Chapter 5 examines *narration*, a text type which gets the hearer to know about event sequences. Chapter 6 examines *exposition*, a form of text which enables the hearer to execute plans, understand processes or understand propositions. Chapter 7 then considers a final form of text, *argument*, which is used to convince the hearer of a proposition or persuade them to act. Chapter 8 then details how TEXPLAN's linguistic realization component produces grammatical and cohesive English from the hierarchical text plans which are exemplified in Chapters 4 through 7. Finally, Chapter 9 summarizes the results, evaluates the research, and suggests areas for future research.

# Chapter 2

# EXPLANATION: HISTORY AND ISSUES

No way of thinking or doing, however ancient, can be trusted without proof.
Henry David Thoreau, *Walden*

## 2.1 Introduction

This chapter discusses past work in computer generated explanations and, in parallel, outlines key problems faced by systems that produce textual explanations. The discussion begins with a brief introduction of philosophical investigations that considered both the content and form of explanations. The problems examined by early philosophers surfaced again later as researchers began to build automated explanation facilities. Computational explanation research has focused on techniques aimed at better representing explanations in knowledge based systems as well as methods that provide more flexible and effective presentations of explanations. The latter includes techniques of planning and linguistically realizing explanations and tailoring them to individual users. The chapter concludes by summarizing past research in automated explanation and indicating some current directions which aim to better represent and present explanations.

## 2.2 "res" versus "verba"

The roots of modern explanation date to the epistemological investigations of early Greek and Roman philosophers. The complex relationship between the content of an explanation and its presentation was evident from the very beginning. Socrates distinguished the art of presenting ideas (rhetoric) from the search for truth (dialectic) and argued that the former was inferior to the latter because rhetoric was independent of truth and, moreover, could be used to achieve immoral ends. He attributed techniques such as definition and subdivision to dialectic. Accordingly, Socrates believed not that idea presentation but rather "wisdom is the beginning and end of eloquence." (Dixon, 1987, p. 13)

In contrast, Aristotle, a pupil of Plato, began his 330 B.C. treatise on the principles of rhetoric by stating "Rhetoric is a counterpart of Dialectic." Aristotle further argues that it is the moral duty of an advocate to present his argument in the most efficacious manner. The Roman author Cicero later praised Aristotle's efforts to unify "the scientific study of facts with practice in style" (De Oratore,[1] III, xxxv, p. 141). Cicero wrote:

> Socrates ... in his discussions separated the science of wise thinking from that of elegant speaking, though in reality they are closely linked together ... This is the source from which has sprung the undoubtedly absurd and unprofitable and reprehensible severance between the tongue and the brain, leading to our having one set of professors to teach us to think and another to teach us to speak (De Oratore, III, p. 60-1).

Consequently, Cicero argues that the perfect orator possesses "wisdom combined with eloquence" (De Oratore, III, p. 142).

Like these Greek and Roman philosophers, researchers investigating automated explanation are faced with the complex relationship between the content of an explanation (the "res") and its presentation (the "verba"). Researchers have investigated, on the one hand, the representation of explanations in knowledge based systems and, on the other hand, presentation techniques which achieve more perspicuous explanations. Figure 2.0 shows the various sources of knowledge (in ovals) and levels of processing involved in moving from some abstract representation of information to the structuring, ordering, and realization of that information as a textual explanation. Thus there are two major components of explanation: the *representation* of the knowledge necessary for explanation and the *presentation* of this to the user in linguistic form. Explanation presentation can be further grossly divided into a *strategic* stage, *text planning*, and a *tactical* phase, *linguistic realization*. Whereas text planning results in structured and ordered explanation content, the *message*, linguistic realization produces English text, i.e., *surface form*.

The oval marked "Application System" in Figure 2.0 signifies the complex system architecture and behavior of some knowledge based application (e.g., a medical diagnosis system, a chemical structure analyzer, or a resource allocation planner). The discourse model refers to some characterization of the intentions, foci of attention, and structure of the discourse (e.g., user queries, system responses, etc.). The rhetorical/speech act model consists of knowledge about the structure and function of communicative acts such as speech acts and rhetorical acts (defined in Chapter 4). Agent models encode the knowledge, beliefs, and desires of discourse participants. Because the specific levels to which the knowledge in the ovals applies varies from system to system, the ovals are simply placed to indicate their generic relevance. Some knowledge sources apply to multiple stages, for example models of attention in the discourse model affect content sequencing (e.g., focus shift rules), syntactic form (e.g., active versus passive voice), and lexicalization (e.g., pronominalization). Within each knowledge source it is useful to distinguish between the *intensional* and *extensional* aspect of the information. This is analogous to the distinction between

[1]"Concerning the Orator" composed three years before Cicero's death.

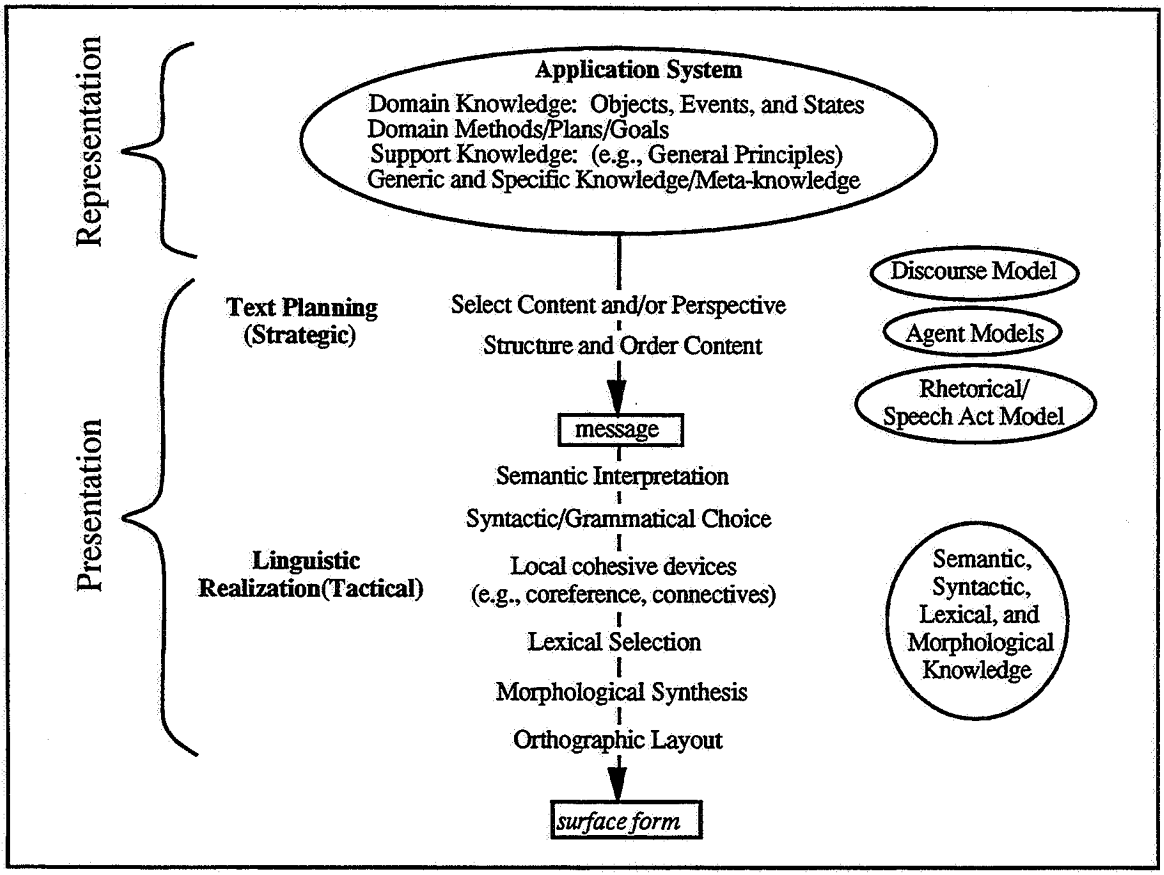



Figure 2.0 Explanation Framework

generic classes (e.g., town) and specific instances (e.g., "Rome, NY") as in object-oriented systems; general versus instantiated methods (or plans) as in planners; and generic versus specific sessions as in a medical consultation. In addition, for each knowledge source it is important to distinguish the structure and properties of the associated knowledge representation formalism from its content in a specific domain or case.

Just as Aristotle and Cicero noted the close interplay between "res" and "verba", it is clear that the modular organization and sequential processing of Figure 2.0 is an oversimplified characterization of explanation, which more likely involves intertwined knowledge sources and parallel processes. Nevertheless, it is useful as an expository framework. The remainder of this chapter first considers early attempts at generating explanations that to a large extent conflated many of the distinctions shown in Figure 2.0. It then discusses previous attempts to better represent explanations and finally critiques techniques for explanation presentation.

## 2.3 Early Explanation Systems/Techniques

### 2.3.1 Canned Text

Sophisticated computational systems that reason need to describe their terminology, explain their methods, and justify their behavior. Initial attempts to explain computer programs centered around single utterances in isolated context. At first messages were simply typed in by the programmer, associating strings of words with code that was executed. This provided canned text as good as the human could compose and was satisfactory in limited contexts (e.g., on-line help). In many situations, however, canned text proved insufficient. First, this approach lacks flexibility. It forces the programmer to anticipate every necessary message and context; it is feasible only in the most trivial of applications. Even more significant, if the underlying system is altered and the canned text remains unchanged, the actual performance of the system can be far from that which the system's messages suggest. Programmers tend to compensate for these potential inconsistencies by writing general and, oftentimes, misleading messages. Consider the following UNIX error message which results after a user tries to find out how to remove a file. U indicates the user, S signifies the system, and numbers indicate the temporal sequence of utterances in the discourse. The system simply outputs the canned message "command not found" along with the input item that triggered it.

```
U1: move my-file to my-subdirectory
S1: move:  Command not found.
U2: Can you tell me how to move a file?
S2: can:  Command not found.
```

In contrast to the above canned message, consider the following response from the UNIX Consultant (Wilensky et al., 1984, 1988):

```
U1: Can you tell me how to move a file?
S1: Use mv.
    For example, to move the file named foo to the file named fool, type
    'mv foo fool'.
```

After formal analysis of the query, this more cooperative response is generated using models of syntax, semantics, rhetoric, context, and domain concepts.

In addition to the weaknesses of inflexibility and potential inconsistency, however, typed-in text strings have no conceptual marking or organization. As a consequence, it is impossible to reason about them to provide more effective explanation, such as providing examples to make things concrete (e.g., the above UNIX Consultant dialogue), making analogies, summarizing content, or describing activities at multiple levels of abstraction.

### 2.3.2 Template Filling: SHRDLU

Terry Winograd (1972) achieved a significant improvement over canned text in his blocks world system SHRDLU using a number of templates ranging from purely canned text to abstract patterns that were realized using expressions for domain objects and events. In the simplest case, SHRDLU used a fixed response, for example saying "ok" when a command was carried out or "I understand" when a declarative sentence was analyzed. A slightly more sophisticated response (analogous to the UNIX example above) involved "filling in the blank" as when SHRDLU responded to the use of an unfamiliar word, *w*, by saying "sorry, I don't know the word *w*." And instead of simply filling the blank with the input phrase, SHRDLU could transform it:

> For example, if the user types something like "the three green pyramids", and the system cannot figure out what he is referring to, it types "I don't know which three green pyramids you mean." It has simply replaced "the" with "which" before filling the blank. The "I assume" mechanism does the opposite, replacing an indefinite determiner or quantifier with "the". If we talk about "some green pyramid" or "a green pyramid", then later refer to that pyramid as "it", the system can notify us of its interpretation of "it" by saying "by 'it' I assume you mean the green pyramid." (Winograd, 1972, p. 163-4)

Finally, SHRDLU could fill in the blanks of templates with referring expressions constructed from its internal model of objects and events in the blocks world (see example in next paragraph).

Patterned responses were triggered by the syntactic form of the question (e.g., yes/no, wh). In a simple case, HOW-MANY questions were answered by finding the relevant objects in the world model, counting them, and then printing the number followed by "of them". A more complex case involved responding to questions asking WHY an action was taken. A WHY question about a top level goal would produce the canned text "because you asked me to." At a lower level in the goal/subgoal tree, however, the system responded to a WHY question by indicating what goal the system was attempting to achieve. For example in one interaction, when asked "Why did you clear off that cube?" SHRDLU replied "to put it on the small red cube." To say this the program first retrieved the associated event, `(#PUTON OBJ1 OBJ2)`, from its history list. SHRDLU then retrieved the template associated with the event `#PUTON` (Winograd, p. 167):

```
(APPEND (VBFIX 'PUT) OBJ1 '(ON) OBJ2)
```

`VBFIX` was a program that produced verb morphology based on the type of question asked (e.g., it returns the "ing" form of the verb to answer HOW questions or the infinitive form to answer WHY questions). `OBJ1` and `OBJ2` are bound to English surface forms by a straightforward naming program based on features of the objects such as their size, shape, or color (Winograd, p. 166). The result is that the `PUTON` event is translated onto the surface form, "To put it on the small red cube." Notice how SHRDLU was able to produce fragmented or incomplete forms.

Associations between domain events and their linguistic expression were manipulated with procedures that map the knowledge representation onto more natural sounding English text. For instance there was a special check for the order of particles and objects to ensure that SHRDLU output "to pick up the small blue pyramid." and "to pick it up." rather than "to pick up it.". Similarly, the pronoun "it" was used if there was reference to an object in the user's query. One key drawback of this approach is the need to anticipate and define by hand templates for each domain action and to carefully control the heuristics that guide the mapping onto surface form. Another difficulty is that templates are repetitive and hence can bore, fatigue, and/or irritate the reader. Nevertheless, SHRDLU's range of templates was a significant improvement over purely canned text.

### 2.3.3 Code Translation: Digitalis Advisor

Instead of using templates, which mix program variables and canned text or "proto text" to represent the underlying program behavior, maximum consistency can arise from directly translating the actions a program executes during an individual run. The Digitalis Therapy Advisor (Swartout, 1977), a program to advise physicians on the appropriate administration of digitalis,[2] was written in OWL, an English-based programming language (Szolovits et al., 1977). For example, the code in Figure 2.1 (called an "OWL plan") tests if the patient is elderly, indicating increased sensitivity to digitalis (Swartout, 1977, p. 40). If the user asks how "How do you check sensitivity due to advanced age?" -- the user actually types in the LISP-form (describe-method [(check (sensitivity (due (to advanced-age))))]) -- the system translates the above code into the English shown in Figure 2.2.

```
[(CHECK (SENSITIVITY (DUE (TO (ADVANCED-AGE))))
 METHOD:
  (OR
     (IF-THEN (GREATER-THAN 70 (AGE PATIENT))
              (BECOME (FACTOR REDUCTION-ADVANCED-AGE 0.75)))
     (BECOME (FACTOR REDUCTION-ADVANCED-AGE 1.0)))]
```

Figure 2.1 OWL Plan

[2]A drug that slows and stabilizes the cardiac rhythm of patients experiencing arrhythmias and that strengthens the heartbeat of patients in heart failure.

```
TO CHECK SENSITIVITY DUE TO ADVANCED-AGE I DO THE FOLLOWING STEPS:
 1. I DO ONE OF THE FOLLOWING:
    1.1 IF THE AGE OF THE PATIENT IS GREATER THAN 70 THEN I SET
    THE FACTOR OF REDUCTION DUE TO ADVANCED-AGE TO 0.75.
    1.2 OTHERWISE I SET THE FACTOR OF REDUCTION DUE TO ADVANCED-
    AGE TO 1.0.
```

Figure 2.2 English version of OWL Plan

As the example shows, the generator maps the OWL plan almost directly onto English surface form. The process is "almost" direct because there are some simple routines for lexical and determiner selection. Near direct translation is possible because all OWL procedures and variables are named after concepts meaningful to the physician using the system. Furthermore, calls to OWL plans are organized to emulate human problem-solving behavior. This implicit ordering provides structure to the explanations (e.g., "CHECK SENSITIVITY DUE TO ADVANCED-AGE"). The production of explanations thus blurs the representation/presentation distinction since the underlying representation -- an OWL plan -- contains entities and entity ordering that are natural for the surface form. Or to put it the other way around, the OWL rules are an internal representation of an expert's stated knowledge.

The example above, however, reveals the linguistic and therefore presentational difficulties that arise from directly translating the underlying representation. Not only is the phraseology rigid, but failure to reason about reference (e.g., repeating the noun phrase "THE FACTOR OF REDUCTION DUE TO ADVANCED-AGE" instead of pronominalizing it) leads to wordy text. Furthermore, the structure of the presentation is confusing, especially the seeming contradiction between the first and second line: "I DO THE FOLLOWING STEPS" and "I DO ONE OF THE FOLLOWING" which arise from translating, respectively, METHOD and OR in the original OWL plan. These presentational inadequacies are exacerbated by the fact that the content does not indicate why advanced age requires a reduction in digitalis.

A slightly more complex example concerns the code used to check for increased digitalis sensitivity caused by increased serum calcium (Swartout, 1981, p. 16) shown in Figure 2.3. The system translates this OWL plan into the English explanation concerning hypercalcemia shown in Figure 2.4. We immediately notice the structural similarity between the first explanation about advanced age and the second explanation about calcium sensitivity. In addition, the second example reveals how extraneous information is included in an explanation simply because the explanation reproduces the underlying code. The record-keeping activities associated with the above procedure (steps 1.1.1 and 1.2) do not add to the intelligibility

```
[(CHECK (SENSITIVITY (DUE (TO (CALCIUM))))
 METHOD:
  (OR
     (IF-THEN
        (OR
           (GREATER-THAN 10 (QUANTA SERUM-CALCIUM))
           (IV-STATUS CALCIUM GIVEN))
        (BECOME (FACTOR REDUCTION-HYPERCALCEMIA 0.75)):1,
        (BECOME-ALSO
           (REASONS REDUCTION HYPERCALCEMIA)):)
     (AND:2
        (UNBECOME (REASONS REDUCTION HYPERCALCEMIA)):
        (BECOME (FACTOR REDUCTION-HYPERCALCEMIA 1.0)):2))]
```

Figure 2.3 OWL Plan for calcium sensitivity

```
TO CHECK SENSITIVITY DUE TO CALCIUM I DO THE FOLLOWING STEPS:

1. I DO ONE OF THE FOLLOWING:

   1.1 IF EITHER THE LEVEL OF SERUM CALCIUM IS GREATER THAN 10
   OR IV CALCIUM IS GIVEN THEN I DO THE FOLLOWING SUBSTEPS:

       1.1.1 I SET THE FACTOR OF REDUCTION DUE TO HYPERCALCEMIA
          TO 0.75.
       1.1.2 I ADD HYPERCALCEMIA TO THE REASONS OF REDUCTION.

   1.2 OTHERWISE, I REMOVE HYPERCALCEMIA FROM THE REASONS OF
   REDUCTION AND SET THE FACTOR OF REDUCTION DUE TO
   HYPERCALCEMIA TO 1.00.
```

Figure 2.4 English explanation of OWL plan for calcium sensitivity

of the explanation with regard to digitalis administration and should be left out. [Of course this complete translation would be a valuable tool for debugging or maintaining the system (Swartout, 1985).]

In addition to these descriptions of general methods, Swartout's program could describe individual cases by keeping a trace of the execution of the code. In the following case the system tells how it checked thyroid sensitivity. The input query "Why did you check sensitivity caused by thyroid function?" is typed in the functional notation:

```
(describe-event [(check (sensitivity (due (to thyroid-function))))])
```

Figure 2.5 illustrates the systems response (Swartout, 1977, pp. 28-29, 52-56).

```
DURING THE SESSION ON 9/21/76 AT 11:10, I CHECKED SENSITIVITY
DUE TO THYROID-FUNCTION BY EXECUTING THE FOLLOWING STEPS:

1. I ASKED THE USER THE STATUS OF MYXEDEMA.
   THE USER RESPONDED THAT THE STATUS OF MYXEDEMA WAS PRESENT.

2. SINCE THE STATUS OF MYXEDEMA WAS PRESENT I DID THE FOLLOWING:

   2.1 I ADDED MYXEDEMA TO THE PRESENT AND CORRECTABLE
   CONDITIONS.  THE PRESENT AND CORRECTABLE CONDITIONS THEN
BECAME MYXEDEMA.

   2.2 I REMOVED MYXEDEMA FROM THE DEGRADABLE CONDITIONS.
       THE DEGRADABLE CONDITIONS THEN BECAME HYPOKALEMIA,
   HYPOXEMIA, CARDIOMYOPATHIES-MI, AND POTENTIAL POTASSIUM
   LOSS DUE TO DIURETICS.

   2.3 I SET THE FACTOR OF REDUCTION DUE TO MYXEDEMA TO 0.67.
       THE FACTOR OF REDUCTION DUE TO MYXEDEMA WAS PREVIOUSLY
   UNDETERMINED.

   2.4 I ADDED MYXEDEMA TO THE REASONS OF REDUCTION.
       THE REASONS OF REDUCTION THEN BECAME MYXEDEMA.
```

Figure 2.5 English explanation of thyroid-function sensitivity

While impressive in content, this example underscores many of the linguistic problems mentioned above. And as Swartout himself later noted (Swartout, 1981), steps 2.1, 2.2, and 2.4 "are more likely to confuse a physician-user than enlighten him" because they refer more to implementation details than to domain or problem solving concepts. Equally, the above text is unclear about the specific purpose of all this activity except to refer to the "reasons of reduction." In fact, the system is going to reduce the dose of digitalis but this motivation may not be obvious to the user because he has no relevant context.

While the Digitalis Advisor failed to explicitly communicate its purpose in the above example, it was at times able to indicate the *intent* of its actions. For example, in the system interaction shown in Figure 2.6 it was able to tell why it asked the user a question (Swartout, 1977, p. 18):

The system produces the first three sentences in the above response by translating the goals in its goal stack into English. The final sentence is canned text associated with the concept that the question is asking about (serum potassium). This final output is significant because it presents important causal information that justifies the request for data. Later we discuss how this implicit knowledge is made explicit in Swartout's (1981) XPLAIN system.

```
S1: WHAT IS THE LEVEL OF SERUM POTASSIUM
    ENTER NUMERICAL VALUE ====>

U1: Why?

S2: MY TOP GOAL IS TO BEGIN THERAPY.  ONE STEP IN DOING THAT
    IS TO CHECK SENSITIVITIES. I AM NOW TRYING TO CHECK
    SENSITIVITY DUE TO POTASSIUM.  IF THE LEVEL OF SERUM
    POTASSIUM IS UNDER 3.70 IT WILL CAUSE THE BODY-STORES GOAL
    TO BE REDUCED SINCE A LOW POTASSIUM CONDITION WILL
    INCREASE DIGITALIS SENSITIVITY.
```

Figure 2.6 English explanation of intention

Digitalis Advisor was a significant improvement over previous canned text and the most basic of template approaches. However, while the "paraphrase the code" approach removed the danger of inconsistency, it revealed that direct code translation often resulted in rigid and confusing text. Winograd's (1972) SHRDLU had in part addressed the inconsistency problem by associating templates with different query forms (e.g., y/n versus wh questions) and filling blanks with translations of underlying domain objects and events (see section 2.3.2). But by translating code, XPLAIN was able to produce longer stretches of output. Swartout's research also emphasized the importance of selecting information pertinent to the type of user (e.g., physicians versus system developers) as well as the importance of indicating the intent of a system's actions.

### 2.3.4 Combining Rule Templates with Code Conversion: MYCIN

In contrast to the Digitalis Advisor, which had the advantage of the linguistic bias of the OWL programming language, the MYCIN expert system for diagnosis of bacterial infections (bacteremia or blood infections and meningitis) represented domain knowledge in 450 pattern-action rules which required much more substantial translation into English. This was done using rule templates and code conversion.

For example, Figure 2.7 shows the internal representation of Rule 050 which determines if the identity of the infecting organism is bacteriodes (Barr and Feigenbaum, 1981, p. 187). The premise of the rule consists of clauses which have the form: `<predicate function> <object> <attribute> <value>`. The vocabulary of the clauses consisted of 24 domain-independent predicate functions (e.g., SAME, KNOWN, DEFINITE) and a range of domain-specific attributes (e.g., IDENTITY, SITE), objects (e.g., ORGANISM, CULTURE), and associated values (e.g., E.COLI, BLOOD). In order to translate rules into English, MYCIN retrieved templates associated with each of its primitive functions (e.g., the AND, SAME, MEMBF, and CONCLUDE functions in Rule 050). The templates used the values of the parameters of the functions to fill in the blanks. This was analogous to Winograd's (1972) association of templates with each

```
PREMISE:    ( AND  (SAME CNTXT INFECT PRIMARY-BACTEREMIA)
                   (MEMBF CNTXT SITE STERILESITES)
                   (SAME CNTXT PORTAL GI) )

ACTION:            (CONCLUSION CNTXT IDENT BACTERIODES TALLY .7)
```

Figure 2.7 MYCIN Rule 050

```
IF    1) the infection is primary-bacteremia, and
      2) the site of the culture is one of the sterile sites, and
      3) the suspected portal of entry of the organism is the
         gastrointestinal tract,

THEN there is suggestive evidence (.7) that the identity of the
      organism is bacteriodes.
```

Figure 2.8 English version of Rule 050

event in the blocks world (see the PUTON example in section 2.3.2) except that Winograd reasoned linguistically about objects and events in the template in order to produce surface form (e.g., choosing determiners, adjectives, and verb endings). For example, consider the MEMBF function from the MYCIN rule in Figure 2.7 and its propositional structure:

```
   (MEMBF       CNTXT        SITE       STERILESITES)
      |           |            |              |
  Predicate     Object     Attribute        Value
```

The template associated with MEMBF translates the above predicate information into the sentence: "The site of the culture is one of the sterile sites." Using both these predicate templates and code conversion techniques, the rule in Figure 2.7 was translated into the English of Figure 2.8. As with the Digitalis Advisor, the structure of the English text blindly followed the underlying representation of Rule 050.

MYCIN diagnosed a disorder by chaining backwards in a depth-first, exhaustive search of the rule set, and attempting to achieve a goal by first finding all rules whose ACTION part matches the goal. It then evaluated each rule's PREMISE in turn, which sometimes led to the invocation of yet another rule that then became a subgoal. MYCIN captured this goal/subgoal information in a "history tree." After the consultation was complete, the user could simply type "WHY" to ascend the history tree and obtain explanations of higher-level goals, or type "HOW" to descend the tree and determine how subgoals were achieved. So if the user were to query "WHY" the conclusion of the above Rule 050 was inferred, the system would translate the premise of the underlying rule. The example shown in Figure 2.9 illustrates the why explanation facility of MYCIN.

```
S1: Does Mary live in a crowded environment?
U1: WHY
S2: [i.e., WHY is it important to determine whether PATIENT-1 does
    live in a crowded environment?]

    [3.0] This will aid in determining the organisms (other than
    those seen on cultures or smears) which might be causing
    the infection.  It has already been established that
    [3.1] the infection which requires therapy is meningitis, and
    [3.2] organisms were not seen on the stain of the pending csf
          culture (CULTURE-1), and
    [3.3] the type of the infection is bacterial

    Therefore, if

    [3.4] Mary does live in a crowed environment then there is
    weakly suggestive evidence (.2) that Neisseria-
    meningitidis is one of the organisms (other than those
    seen on cultures of smears) which might be causing the
    infection

Author: Yu
[Rule533]
[back to question 1]
```

Figure 2.9 MYCIN's "why" explanation facility

While MYCIN's explanations look impressive, there are several limitations worth noting. Even though the user's simple query "WHY" (or "HOW") has multiple interpretations (e.g., "why did you conclude this (i.e., what is the premise)," "why does the conclusion follow from the premise," "why are you asking this question now," "why is this question important," and so on), MYCIN's lack of linguistic knowledge forces it to interpret WHY or HOW in the most straightforward manner, limiting the kinds of interesting questions one might pose to the system. Furthermore, as Davis (1976) first pointed out, MYCIN does not have the knowledge to respond to these other interpretations. To compensate for this deficiency, MYCIN prints out its (standard) interpretation of the user's query using the template "Why is it important to determine <data>?" (Hasling et al., 1983, p. 5).

In addition to the HOW and WHY facilities, MYCIN allowed the user to ask a restricted set of specialized question types about both general domain and specific session information. For example, if the user asks (in a restricted query language) about the <value> of <parameter> in <context>, the system can select between two simple templates associated with that question type. If the system inferred the value, it uses the template:

```
I used <rule> to conclude that <parameter> of <context> is <value>.  This
gave a cumulative Certainty Factor of <certainty factor>.
The last question asked before the conclusion was made was <question
number>.
```

If the user supplied the value, however, it fills in the blanks of:

```
In answer to question <question number> you said that <parameter> of
<context> is <value>.
```

While this may allow for a greater range of input questions, the simple template filling approach to response generation used for these types of questions is inflexible and repetitious. Furthermore, since MYCIN has no representation of dialogue context, it is unable to relate its output to previous explanations in the dialogue.

In addition to these presentational deficiencies, MYCIN's explanations are also epistemologically lacking. For example, control knowledge is implicit. The ordering of the rules in the knowledge base and the ordering of the clauses in the premise of a rule is an implicit representation of strategic problem-solving knowledge. That is, some rules and clauses screen out others, thus guiding the search process to avoid needless processing or question asking. Other implicit control knowledge includes MYCIN's global deduce-then-ask strategy which avoids asking questions for which it can deduce an answer. Furthermore, different types of knowledge, such as causal and evidential knowledge, are intertwined in MYCIN's rules. And finally, there is often knowledge missing that justifies why the conclusion follows from the premise. As we will discuss in the next section, researchers recognized these limitations and began to search for ways to improve underlying representations, for example NEOMYCIN's explicit representation of control knowledge in metarules.

### 2.3.5 Lessons from MYCIN and the Digitalis Advisor

With the development of systems to perform expert problem solving it was initially believed that if the problem solving activities could be paraphrased then adequate explanations would result. Indeed, MYCIN and the Digitalis Advisor illustrated that (deep) templates based on the predicate structure of rules and code conversion keep the presentation consistent with the underlying program (as did Winograd's event templates). As we have seen, however, this strength is also a great weakness since communicative success is tied closely to the proper representation and organization of the code. The programmer must be careful to choose procedure and variable name translations that are meaningful to the end-user and, more importantly, to organize methods and knowledge in a manner that will effectively structure an explanation. Unfortunately, a common result of direct translation is inflexible and often rigid output. Perhaps the most significant contribution of MYCIN and the Digitalis Advisor was their revelation of the need for additional support knowledge to define the terms used in underlying statements (and hence concepts), to explicate the purpose behind actions, and to justify inferences by indicating their rationale.

## 2.4 Explanation Representation

To overcome some of these deficiencies in expert system explanation, researchers focused on more explicit and enhanced representations of knowledge and reasoning strategies. This section first discusses attempts to represent explanation at multiple levels of abstraction, then approaches to providing richer support knowledge (e.g., access to domain concepts and domain principles), and finally points out some unresolved issues and current research directions.

### 2.4.1 Explicit Representation of Control Strategy: NEOMYCIN

With the goal of applying MYCIN to automated tutoring, Clancey (1983) found the need to extract the problem-solving knowledge implicit in MYCIN's 400 rules and explicitly represent it in the form of domain-independent metarules (called “tasks”) that controlled the selection and execution of the domain-specific rules. The resulting system, NEOMYCIN, produced explanations in the same (deep) rule template manner as MYCIN, but since the domain-independent problem-solving knowledge in the metarules was explicitly separated from domain knowledge, the user could seek either an abstract explanation of the reasoning strategy or a concrete explanation of the current medical consultation.

Consider the diagnostic session shown in Figure 2.10 (Hasling et al., 1983, p. 10). The response, S5, gives the system’s reason for asking the user its question by referring to the concept’s diagnostic utility in the domain. The metarule that is driving this request for information about headaches is (Hasling et al., 1983, p. 11) shown in Figure 2.11. NEOMYCIN uses this metarule to produce the abstract explanation shown in Figure 2.12. The template for this more general explanation is shown in Figure 2.13 (Hasling et al., 1983, p.11).

```
S1: Please enter information about the patient.
    Name  Age  Sex Race
U1: MARY  42   F   LATINO
S2: Please describe the chief complaints:
U2: HEADACHE
    STIFF-NECK
    NAUSEA
S3: How many days has Mary's headache lasted?
U3: 10
S4: How severe is Mary's headache (on a scale of 0 to 4 with 0
    for very mild and 4 for very severe?)
U4: WHY
S5: [i.e., WHY is it important to determine the severity of Mary's
          headache?]

    [1.0] We are trying to characterize the data just obtained
    and develop the differential.

    Headache severity is a characterizing feature of headache.
```

Figure 2.10 NEOMYCIN Diagnostic Session

```
                         METARULE073

IF     there is a datum that can be requested that is a
       characterizing feature of the recent finding that is
       currently being considered,
THEN   find out about the datum.
```

Figure 2.11 NEOMYCIN Metarule 073

```
U5: ABSTRACT
S6: [i.e., WHY is it important to determine the severity of Mary's
          headache?]

    [1.0] We are trying to characterize the data just obtained
    and develop the differential.

    A datum that can be requested is a characterizing feature of
    the recent finding that is currently being considered.
```

Figure 2.12 NEOMYCIN Abstract Explanation

```
[i.e., WHY is it important to determine <data>?]

[1.0] <immediately preceding task>

<canned text that tells what is true about the domain
knowledge base or the problem-solving history that enables
the metarule that accomplishes this task to succeed>
```

Figure 2.13 NEOMYCIN Explanation Template

So while the user's prompting question was simple and the presentation strategy utilized the same primitive templates, the ability to flexibly choose between abstract and concrete knowledge levels was a significant improvement. Moreover, users could identify themselves as either system or domain experts which resulted in the selection of appropriate words or phrases during rule translation. For example, the translation of a causal link could use the phrase "is strongly associated with." But if the user was identified as a system expert, NEOMYCIN substitutes the word "triggers" (cf. Moore and Swartout, 1987). Hence, NEOMYCIN reflected the modest beginning of an expanding area of work in tailoring output (lexical/phrasal selection) to the user. Nevertheless, NEOMYCIN was still unable to provide justifications underlying inferences, or definitions of terminology. These inadequacies led Clancey (1983, 1986) later to argue that additional types of knowledge were required to explain rule based systems including the structure of the domain (e.g., subsumption relations among data, diagnoses, and therapies), problem solving strategies (i.e., the procedure for applying rules), and support knowledge (e.g., the causal model underlying rules).

### 2.4.2 Explicit Representation of Support Knowledge: XPLAIN

Despite NEOMYCIN's ability to explain at multiple levels of abstraction by exploiting explicit strategic knowledge, several classes of explanation were not represented in the improved architecture, including justification of behavior and definition of terminology. Part of the problem was that the implicit rationale in, for example, premises leading to conclusions, was only known by the programmer or domain expert at the time of system development. This led Swartout (1981) to design XPLAIN (see Figure 2.14) as an improvement of the Digitalis Advisor. XPLAIN automatically combines causal knowledge of the domain (called a domain model) together with general problem solving methods for the domain (called domain principles) to generate a "refinement structure." The resulting "refinement structure" contains links from rules to domain principles which allows justification for actions to be included in explanations. Thus though XPLAIN still relied on code conversion for its output English, the content of its explanations was superior to those from the Digitalis Advisor.

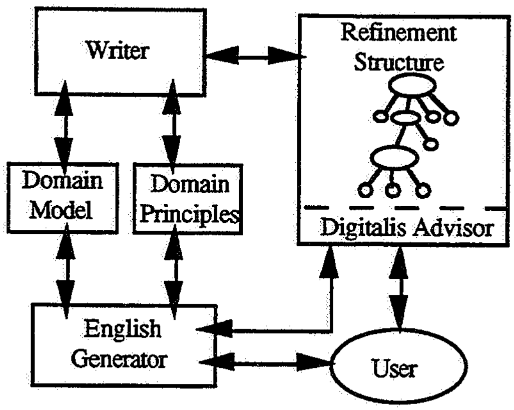



Figure 2.14 XPLAIN System Overview

To see the improvement, recall the explanation the original Digitalis Advisor produced by translating the most recent goals in the stack, shown in Figure 2.15. In contrast, the new version of Digitalis Advisor, XPLAIN, produces the explanation shown in Figure 2.16. Notice in the explanation in Figure 2.16 that not only is the system able to explain its behavior by translating the code, it is able to justify its behavior by indicating the causal relation of Serum-Ca and of Serum-K. Moreover, since the abstract method for dosage reduction is the same for serum calcium and serum potassium, the explanation translator is able to make an analogy to the first in the explication of the second. This improved explanation is achieved by paraphrasing the code using the following algorithm:

1. Describe the current goal.
2. Find the current domain principle in development history and use the domain rationale to describe how it matches the domain model.
3. Describe the prototype method of the domain principle.

The Digitalis Advisor's domain principles include a goal, a prototype method, and a domain rationale (see Figure 2.17). The domain rationale distinguishes XPLAIN from other refinement based systems (e.g., Sacerdoti, 1977) since it uses knowledge from the descriptive domain model (see Figure 2.18) to define terms at one level of refinement using terms in the level below (Neches, Swartout, Moore, 1985, p. 1339). To understand how the domain rationale indicates this interlevel correlation, consider the abstract description in Figure 2.17 of the domain principle that concerns anticipation of drug toxicity. The low-level description of this method is concerned with "findings." The domain rationale in Figure 2.18 indicates which findings indicate a dangerous deviation.

```
S1: What is the level of Serum Calcium?
    ENTER NUMERICAL VALUE ====>
U1: why?
S2: My top goal is to begin therapy.  One step in doing that is
    to check sensitivities.  I am now trying to check
    sensitivity to Calcium.
```

Figure 2.15 Digitalis Explanation

```
S1: Please enter the value of Serum-Ca:
U1: why?
S2: The system is anticipating Digitalis toxicity.  Increased
    Serum-Ca causes increased automaticity, which may cause a
    change to ventricular fibrillation.  Increased Digitalis
    also causes increased automaticity.  Thus, if the system
    observes increased Serum-Ca, it reduces the dose of
    digitalis due to increased Serum-Ca.
    Please enter the value of Serum-Ca:
U2: 3.7
S3: Please enter the value of Serum-K:
U3: why?
S4: The system is anticipating Digitalis toxicity.  Decreased
    Serum-K also causes increased automaticity.  Thus, (as with
    increased Serum-Ca) if the system observes decreased
    Serum-K, it reduces the dose of digitalis due to decreased
    Serum-K.
    Please enter the value of Serum-K:  9.0
```

Figure 2.16 XPLAIN Explanation

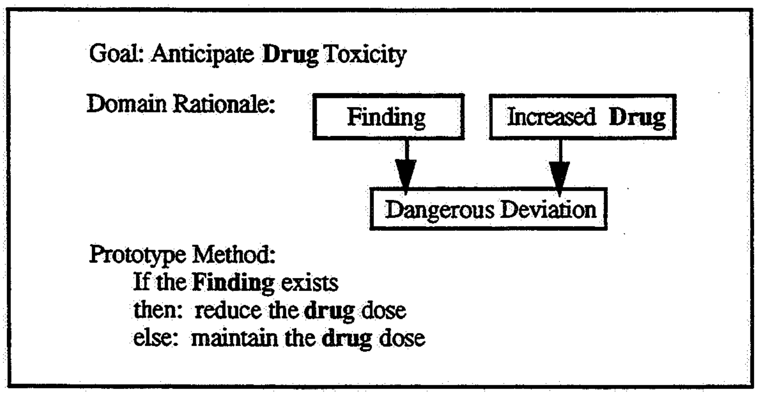



Figure 2.17 Domain Principle

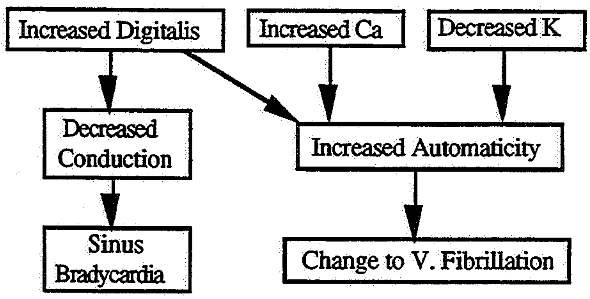



Figure 2.18 XPLAIN Domain Model

Just as NEOMYCIN represented strategic knowledge in metarules, XPLAIN represents generic methods so that it can reason using domain-independent problem-solving techniques, instantiated with domain-dependent knowledge. Equally, just as NEOMYCIN was able to vary its level of abstraction and to tailor some of its phrasal selection when producing output, so too XPLAIN was able to tailor content to particular users through its use of "viewpoints." These are markers in the knowledge base that indicated which steps in a prototype method were implementation details which were useful for a programmer but confusing to a physician. These steps were filtered out depending upon the type of user.

### 2.4.3 Additional Support Knowledge: Explainable Expert Systems

As a result of the explicit representation of strategic knowledge, NEOMYCIN and XPLAIN were able to provide abstract descriptions of their problem solving behavior, and XPLAIN was able to justify its actions. The Explainable Expert Systems (EES) project (Swartout, 1983; Neches et al., 1985; Swartout and Smoliar, 1987) focused on representing additional types of knowledge (see Figure 2.19) to answer an even wider range of questions. EES, like XPLAIN, used domain principles as well as a domain model. However, the methods contained in the domain principles were more generic. Control knowledge which drives the selection of subgoals is explicitly represented. "Tradeoffs" indicate beneficial and harmful effects of selecting a particular strategy to achieve a goal. "Preferences" are then used to set priorities among goals based on the tradeoffs. Furthermore, "integration knowledge" resolves conflicts among knowledge sources. Terminology is captured in one module that can be shared across domain principles by separating abstract terms and the concepts that realize them in the domain principles. Finally, "optimization knowledge" represents methods to efficiently control the execution of the derived expert system (e.g., performance-driven ordering of actions).

As a consequence of the explicit representation of the above knowledge, the program writer, previously limited to goal/subgoal refinement using a hierarchical planner, could generate a more expressive refinement structure. This results in a richer development history which thus allows for a wider variety of explanations.

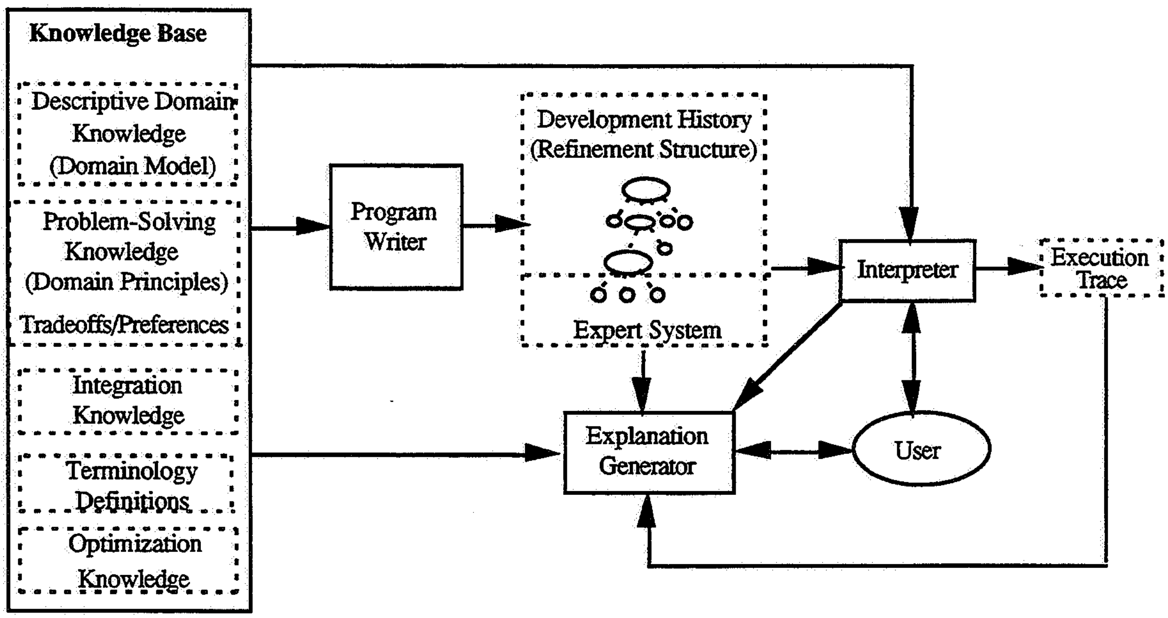



Figure 2.19 EES Framework

To exploit this strength, EES investigated a taxonomy of question types which could occur with expert systems ranging from those a programmer might ask, e.g., about timing, parameter usage, and procedure calls, to those an end user might ask concerning terminology, problem-solving methods, or intent. The goal is to associate different explanation strategies with types of input question. For example a user may ask "Why should the <recommendation> be followed?" In order to justify the recommendation EES uses the strategy outlined in Figure 2.20. This is actually a simplified version since it considers only goals and ignores preferences or tradeoffs.

```
1. Search the development history for the <method> that produced
   <recommendation>.
2. Search upward through the development history for the <goal> that
   this <method> is a plan for achieving. Continue searching upward
   until reaching a goal that the user shares. (The user is assumed to
   share the top-level goals of the system.)
3. State this <goal>.
4. State the general <method> that is used to achieve <goal>.
5. State how <recommendation> is involved in achieving <goal>.
```

Figure 2.20 "Justify Recommendation" Explanation Strategy in EES
(Neches et al., 1985, p. 1348)

The "justify recommendation" strategy can be illustrated using an example from the Program Enhancement Advisor (PEA), an expert system which tells the user how to improve the readability, maintainability, or efficiency of a given piece of Common LISP code. Thus the system might suggest the following:

```
The construct:
     (COND ((ATOMP X) (LIST X))
           (T X))
may be replaced by the following construct:
     (IF (ATOMP X)
         THEN (LIST X)
         ELSE X)
```

If the user then asks the system[3] to justify this recommended transformation, the system will apply the "Justify Recommendation" strategy to produce the text (Neches et al., 1985, p. 1348):

```
The system is trying to enhance the readability of the program by
applying readability enhancing transformations. COND to IF-THEN-ELSE is
a readability enhancing transformation because IF-THEN-ELSE has keywords
which identify its abstract components.
```

While EES provides a richer foundation than its predecessors upon which to build explanations, it still remains to be seen how well this will work in practice. Few of the explanation strategies have been implemented (Swartout, personal communication, 1989). Moreover, in relation to the PEA application, automatic programming is still an art whose difficulty is exacerbated by complex interaction between knowledge sources such as integration and optimization knowledge. In addition there is a large, up-front expense of explicitly encoding the various types of support knowledge. Finally, the extra support knowledge offered by this framework must be conveyed via independent linguistic capabilities and explanation strategies which are the focus of some current activity (Moore and Swartout, 1988ab).

### 2.4.4 Current Directions and Unresolved Issues in Representing Explanations

While research in explanation has yielded a number of important advances beyond the primitive canned text and template approaches, there are still many unsolved problems. By separating out different types of information (e.g., control versus causal knowledge), systems have been able to provide a wider variety of explanations more precisely. And the representation of knowledge at multiple levels of abstraction has allowed for explanations at various levels of detail. The early systems, the Digitalis Advisor and MYCIN, provided summary explanations by, respectively, listing procedure calls and listing rule names. The assumption was that the procedures or rules represented a chunk of conceptual knowledge that related to what the system was reasoning about "in general." However, this approach places a heavy organizational burden on the programmer and it fails to recognize that implementation details which may be incomprehensible or irrelevant from the user's point of view frequently outnumber domain-related

[3]The user's query is not stated in natural language but rather in a restricted command language.

concepts. NEOMYCIN's extraction of control knowledge allowed for the natural production of abstract/concrete explanations. But it soon became clear that more sophisticated models of explanation would be necessary to produce output sufficiently sensitive to user's expertise, their role, and the context of the dialogue.

Richer output was achieved, in part, by adding support knowledge which allowed for deeper justification and explication of terminology. But attempts to translate underlying decision-making representations to explain behavior made the fundamental assumption that the process of reasoning mirrors the process of explanation and there is evidence to the contrary. For instance, consider the following text produced orally by a field inspector as he evaluates the stability of a concrete dam (Franc, 1987 from Wick et al., 1988):

> The progressive opening of the cracks in the dam's wall suggests to me that the concrete may be weakening. Oh yes, there is heavy spalling. This indicates the concrete is breaking apart due to a chemical imbalance. But, there seems to be too many cracks for just this. Perhaps there is also a support problem. Yes, the drumming noise from under the dam indicates the slab might be cracked leading to an undermined foundation, that explains the cracked retaining walls. Also, the cracks parallel to the crest indicate further damage from weather. Overall, the dam has poor support, weather damage, and weakening concrete. As such, there is a large risk of gradual uncontrolled release of water. I would strongly recommend preventive action.

In contrast to this heuristic, data-driven problem-solving process, the expert "tells a story" to justify his or her conclusion after the evaluation is complete:

> The dam is highly unstable and should receive preventive action. In analyzing the dam for stability, three factors are used: the condition of the support, the level of the load, and the pre-existing conditions that could affect dam stability. In this case, the load level is fine. However, the support condition is strongly suspect. A drumming sound coming from the base of the dam indicates that the support slab might be cracked. The drumming sound is caused by the water "slapping" against the open crack ... Long cracks parallel to the crest have been caused by excessively cold weather contracting the dam beyond its safe limits. Also, the collection of concrete dust at the toe of the dam further supports the conclusion that the damage is due to weather. All told, the dam is far too weak to withstand the force of the water, thus the gradual uncontrolled release of water is highly likely.

The discussion of early explanation research suggests that problem-solving and explanation strategies may not be isomorphic, and this is clearly indicated by the texts just given. At the same time, these texts plainly show how a variety of linguistic devices help to produce a natural sounding, flowing discourse: focal links (e.g., "drumming sound" connects the fourth and fifth utterances), lexical connections (e.g., "however", "also", and "all told"), lexical choice (the use of the domain terminology "heavy spalling" in the first passage which indicates the breaking or chipping away of the dam due to chemical imbalance versus the more general description of "a collection of concrete dust" in the second), and rhetorical connection (e.g., providing evidence followed by causation). In addition to these presentational improvements, new knowledge about causes, symptoms, and background (e.g., that the weather was cold) are added to enhance the content of the explanation.

This example also indicates that experts seem to learn "compiled associations" whereby certain data may trigger likely hypotheses which allow for efficient convergence on a solution. Experienced problem solvers, as in the above example, focus on key symptoms and use heuristics such as asking screening questions followed by pinning down questions (Clancey and Letsinger, 1981) that allow them to efficiently associate evidence with causes. This seems to support the argument for distinct but interrelated cognitive processes of reasoning and presenting an explanation. Some presentations may indeed take advantage of an underlying representation to order and structure text (e.g., the description of physical objects based on their structure and function (Paris, 1987ab)). Yet other presentations, such as explaining the conclusion of a diagnosis or a planning activity, do not necessarily need to communicate the structure of the reasoning and for example may use rhetorical techniques (e.g., analogy, comparison/contrast) to make the conclusions more intelligible or believable. Indeed, providing analogies or comparisons are powerful techniques for explaining the domain to novice or intermediate users.[4] Not only are these communicative skills unimportant in solving the problem, but the distinguishing characteristics of objects and processes, essential for comparison, are typically not represented. Early explanation systems thus revealed that their knowledge sources were not only epistemologically incomplete but often unable to support linguistic tasks such as language generation.

These deficiencies extend beyond the system's stock of knowledge and apply also to the information retrieval mechanisms. We have already discussed how rehearsing function calls or rule traces provides poor and sometimes misleading explanations. In the future, we will need multiple views of an application's knowledge (Suthers, 1988) to support multi-perspective explanation. This raises the issue of explanation critique, so that only the best or most effective explanation in the current situation is selected to be communicated. Chandrasekaran (1986) emphasized the notion of *abduction*:, the "best explanation, critically assessed."

Richer representations of explanation yield richer explanation content, but this in turn demands more sophisticated natural language techniques to release their full power. In particular, natural language interfaces to more complex information sources require not only lexical, syntactic, and semantic knowledge but also pragmatic mechanisms that can disambiguate queries, construct and maintain user and discourse models, tailor output to the characteristics of the user and dialogue context, and recover from miscommunications.

[4]Analogy is of course also a powerful reasoning tool, but using it for presenting and outcome does not presuppose it has been used to obtain the outcome.

## 2.5 From Representing Explanations to Presenting Them

One of the most common approaches to explanation presentation -- following underlying program structure -- solved some of the consistency problems of canned text since output is a direct product of the knowledge or rule trace. However with this code translation approach (whether or not templates or more sophisticated sentence realization mechanisms are used), the only hope for cohesion and coherence over longer stretches of text rests with the intelligibility of the underlying plan or rule chain. Enriching the underlying knowledge, while encouraging more explicit representations, tends to complicate attempts to present output effectively because the presentation planner is obliged to consider a wider range of knowledge. Advances in explanation representation fueled a search for more sophisticated presentation mechanisms that go beyond application-motivated presentation heuristics and explicitly model the various levels of text planning and linguistic realization indicated at the beginning of this chapter in Figure 2.0.

Early work on explanation generation often ignored distinctions between different knowledge sources (e.g., lexical, syntactic, semantic). This is indeed a natural consequence of using canned text and templates, especially shallow ones. Equally, early research often failed to distinguish between the levels of processing in Figure 2.0 or, worse, made ill-motivated distinctions. Even with deep templates the distinction between text planning and linguistic realization is rather crude. The general failure to delineate knowledge sources and levels of processing was exacerbated by the fact that in moving from one level of processing to another, the interaction of constraints can be quite complex, especially if a system is attempting incremental language generation. Indeed the relationship between text planning and realization remains a controversial issue (Hovy et al., 1988).

The remainder of this chapter considers the evolution of a number of computational models that manipulate both linguistic and general knowledge to resolve the presentational issues of exactly what to say, given some message content, when to say it, and how to say it. The discussion progresses from words to phrases to sentences and finally to full paragraphs. Thus referring back to Figure 2.0, it begins with tactical processing by examining the realization of short passages of language using mechanisms that determine syntactic structure and make lexical selections. It then addresses strategic processing, first by analyzing techniques such as discourse strategies which were developed to plan larger stretches of text. Then it discusses how the content, words, grammar, point of view, and rhetorical structure of explanations can be tailored to individual users. Finally, it discusses recovering from miscommunications. The chapter concludes by indicating some current foci of research.

## 2.6 Linguistic Realization

Researchers initially concentrated on techniques for mapping established knowledge representation schemes onto English surface forms for isolated sentences as opposed to considering the issues involved in generating multi-sentence text. Early investigations focused on generating paraphrases from knowledge structures guided by network representations of sentence grammar. Others explored methods for word choice. More recently, a number of grammatical formalisms (e.g., TAG, systemic grammar), some with the aim of bidirection and bilinguality, have become the focus of intensive research. This section discusses each of these sentence level realization efforts in turn which leads into the next section which considers the production of extended text.

### 2.6.1 Realization from the Lexical Point of View

Early work in linguistic realization was varied in both form and intention, focusing on both sentence structure and lexical choice. Because the latter (interpreted broadly to include designators like referring expressions) both in itself and in its relation to structure determination is important, this subsection examines realization from the lexical point of view and then examines work specifically focused on lexical selection.

Winograd (1972) was able to generate very natural English for his more complex types of template and patterned response because he could associate SHRDLU's limited number of primitive functions fairly directly with English surface forms and was able to apply strong input-derived constraints to pattern choices. In contrast, Simmons and Slocum (1972) initiated linguistically-based realization by producing sentences from knowledge stored in a verb case semantic network. A typical network node, for example, consisted of a TOKEN (e.g., wrestle), a TIME (progressive past) and an AGENT (John), as well as perhaps information about MOOD (indicative) or VOICE (passive). The generator produced sentences by following the arcs of an Augmented Transition Network (ATN) grammar (Woods, 1970) which were labeled with the names of the relations in the underlying semantic network (e.g., TOKEN, TIME). By varying the starting point and the nodes present in the semantic network, the program could produce such alternatives as:

> John saw Mary wrestling with a bottle at the liquor bar. He went over to help her with it. He drew the cork and they drank champagne together.

or

> John saw Mary wrestling with a bottle at the liquor bar. John went over to help her with it before he drew the cork. John and Mary drank the champagne.

Unfortunately, most applications cannot be driven with a linguistically encoded knowledge base and so this approach was limited in its utility.

Building on Simmons and Slocum's (1972) research, Goldman (1975) developed BABEL, the sentence generator in MARGIE (Schank, 1975), a system that answered questions about, made inference from, and paraphrased Conceptual Dependency (CD) networks. Although he used Simmons' and Slocum's ATN generator to produce syntactic structures, Goldman pioneered the use of *discrimination nets* (in essence binary decision trees) to select lexemes, and through them case structures. Goldman focused on verb choice because of CD's bias toward language-independent primitive acts and because verbs are the base for sentence organization. Verb selection was accomplished by traversing the discrimination nets. For example, BABEL could express the CD primitive INGEST as a number of verbs such as "eat", "drink", or "breath" by considering the properties of the substance to be ingested. Each of these verbs actually suggested a class of further choices. For example, a non-human object would "lap" whereas a human would "drink" or "guzzle" depending upon the velocity of the ingestion (see Figure 2.21).

After choosing the appropriate verb, a case-based semantic template was formed which was then mapped onto surface form by using an ATN. The three paraphrases below illustrate the lexical (and consequently structural) variation achieved by Goldman's dictionary mechanism:

1. Othello strangled Desdemona.
2. Othello choked Desdemona and she died because she was unable to breath.
3. Othello choked Desdemona and she died because she was unable to inhale air.

Goldman's dictionary formulation influenced many subsequent generation systems even though he did not linguistically justify his paraphrase choices, which were simply alternatives that could be generated from the underlying CD form, or demonstrate contextual influence in multi-sentential output.

To produce more varied text, a number of researchers addressed lexeme selection by examining knowledge other than the semantic properties of an object. Some emphasized syntactic constraints. For

```
                          (equal? action INGEST)
                        T /                     \ F
              (property? object fluid)          ...
           T /                        \ F
      (human? actor)            (property? object gas)
      T /          \ F              T /        \ F
(equal? manner fast)  lap          breathe      eat
    T /      \ F
  guzzle   drink
```

Figure 2.21 BABEL's Discrimination Net for CD primitive INGEST example, Boguraev's (1979) "synonym driven" paraphraser, designed as a means of showing how input sentences had been lexically and structurally disambiguated, went beyond simple word substitution and examined paraphrases which required constituent manipulation. Unlike BABEL, Boguraev's paraphraser started from a representation with all major lexical items specified for the output. The system first chose a verb synonymic to the original verb by matching semantic knowledge to find the best fit (*sense selection*). Then it retrieved case frames (e.g., agent, patient, beneficiary) associated with the new verb. For example, when the system attempted to paraphrase the utterance "John asked Mary about the book," it first retrieved the three dictionary entries which correspond to the three groups of "ask" synonyms: QUESTION or INQUIRE, BEG or WANT, and REQUEST. By examining the grammatical case roles associated with each verb and the content of these roles (e.g., QUESTION requires a human as a direct object), the verb closest to the input was selected. In the above example, the verb INQUIRE is selected and its case frame is retrieved (Boguraev, 1979, p. 5.25):

```
@agent
       INQUIRE
       ABOUT @subj-matter
       FROM @recipient
```

The case frames specified collocational and syntactic restrictions which were used when filling in the slots of the case frames (*structure building*). The overall utterance was represented in an *environment net* -- "a linearly ordered, hierarchically organized, language dependent data structure" -- which contained case, syntactic, and lexical constraints (Boguraev, 1979, p. 5.25). A set of grammatical rules governed constituent ordering. Realization was accomplished by traversing the environment net left-to-right and assembling syntactic data which was then used to output lexemes. In the above example, the final output would be "John inquired about the book from Mary" or alternatively "John questioned Mary about the book."

In contrast to this use of semantic and grammatical information to govern lexical selection, Appelt (1985) focused on lexical choice, and more generally that of expression, based on the hearer's knowledge. When planning referring expressions, Appelt's system could say "Use the wheelpuller next" if the hearer knew about the wheelpuller already. If not, his system could identify the item by indicating its distinguishing physical properties: "Use the red tool on the table next" (McKeown and Swartout, 1987). And as we discuss later, by modeling a set of rhetorical registers (e.g., terseness, style, etc.) Hovy (1987) examined selection of words and phrases based on models of the relationships between the speaker and the hearer as well as the speaker and hearer's disposition toward the discourse topic.

In reaction to these attempts at ever more complex lexical choice mechanisms, Danlos (1984; 1987) emphasized that since there are so many sources of knowledge constraining lexical selection, any moderately sophisticated approach would quickly become computationally intractable. Instead she suggests the use of a "discourse grammar," a deep template that identifies not only discourse organization but also syntactic markers that are interpreted by a syntactic grammar. Unfortunately, her solution lacks generality: each new application requires hand encoding of the discourse patterns. It does not obviously deal either with the issue of choosing appropriately detailed referring expressions within discourse (e.g., "the big black box" or "the big box" or "the box" or "it"). Nevertheless, this approach may be most effective in restricted applications.

Danlos' approach is indeed just one form of the "sublanguage" strategy which has been widely exploited not only in semantic grammars (e.g., for database query), but especially in message and longer text processing. Sublanguages are linguistic systems that characterize domain specific grammatical structures and vocabulary. For example, weather bulletins omit articles and non-tensed verbs, and this is reflected in the METEO machine translation system (cf. articles in (Nirenburg, 1987)). Sublanguages deliver efficiency since they reduce the size of the grammar and constrain lexical ambiguities. Furthermore, sublanguages deal head-on with troubling issues such as semi-frozen phrases and idiomatic expressions that are difficult to represent in current syntactic and semantic formalisms (e.g., "by and large," "all of a sudden," and "kick the bucket")[5].

These phrasal expressions can be incorporated in the lexicon (Becker, 1975) as patterns with varying degrees of modularity and flexibility, and indeed can form the base to the whole process of interpretation and generation as in Jacobs (1985). Jacobs built a knowledge base of "pattern-concept" pairs used during English or Spanish interaction with the UNIX Consultant (UC) system (Wilensky et al., 1984; 1988). The pattern-concept pairs -- feature systems like those used in Generalized Phrase Structure Grammar and Functional Unification Grammar -- link phrasal patterns to a conceptual template. For example, Jacobs' generator can produce the phrases "Mary gave John a punch" and "John took a punch from Mary" from the same conceptual template. This template representation can capture the UNIX-world meaning of phrases like "working directory" or its Spanish equivalent "espacio de trabajo." PHRED, the generator, shares its knowledge with, PHRAN, the interpreter, providing cognitive economy and ease of maintenance.

While this work provides a technique for capturing and manipulating complex lexical phenomena, other researchers have focused on exploring new mechanisms to achieve lexical variation. Granville (1984) developed a system that chooses pronominalization, superordinate substitution (replacing an entity with a more general term), or definite noun phrase reiteration in order to enhance cohesion. Carter (1986) addressed the issue of providing optimally unambiguous referring expressions, as a by product of work in anaphor resolution. Joshi (1987) discusses the use of Tree Adjoining Grammars (which provide local definition of all syntactic dependencies, linear order, and preservation of argument structure) to vary word

[5]Note that the last of these examples can be interpreted using traditional grammar whereas the first two examples are fixed phrases.

order. As we discuss in the next section, Tree Adjoining Grammars are most effective in the incremental generation of grammatical structure.

Miezitis (1988) uses a "spreading activation" model to retrieve appropriate lexical units (individual words as well as idioms such as "bury the hatchet") from which an individual choice can be made based on syntactic, semantic or pragmatic criteria. The mechanism developed, called a Lexical Option Generator (LOG), dynamically matches a given situation represented in a frame formalism to a set of possible lexical choices. For example, given the input (love (agent mary1) (patient john1)), LOG retrieves information for the output "Mary loves John" or "John is the apple of Mary's eye," which can then serve as possible choices for a generator that imposes stylistic or pragmatic constraints (e.g., Watt, 1988). While Miezitis' hierarchical representation of the lexicon is not novel (see Quillian, 1966), the process in which nodes in the hierarchy attract or "magnetize" relevant information from the input specification is unique and is claimed, but not proven, to be more efficient than non-"spreading activation" approaches.

While these lexical mechanisms undoubtedly can add to the repertoire of surface generation techniques, they cannot really be the basis for the realization of text. This requires grammatical representations independent of any underlying knowledge representation formalism, and for any but the most limited applications, domain-independent grammatical formalisms which cover a reasonable range of English grammar. Though some earlier work exploited general-purpose syntax explicitly, the provision of appropriately powerful syntactic mechanisms has been a major concern of the last decade.

### 2.6.2 MUMBLE and Tree Adjoining Grammars

One of the most significant generation systems is McDonald's (1980 (unpublished), 1981, 1986) MUMBLE, and its modern version, MUMBLE-86 (Meteer et al., 1987), both which have the goal of being an efficient and application-independent linguistic realization component. From the beginning MUMBLE addressed knowledge representation independency. McDonald (1981) explored the generation of English from a variety of knowledge representations including predicate calculus, FRL (Frame-oriented Representation Language) (Goldstein and Roberts, 1977) and KL-ONE, a structured-inheritance semantic network formalism (Brachman, 1979).

```
                              INPUT

1. premise:
     ∃x (barber(x) and ∀y (shaves(x,y) <-> not shaves(y,y)))
2. existential instantiation (1):
     barber(g) and ∀y (shaves(g,y) <-> not shaves(y,y))
3. tautology (2):
     ∀y shaves(g,y) <-> not shaves(y,y)
4. universal instantiation (3):
     shaves(g,g) <-> not shaves(g,g)
5. tautology (4):
     shaves(g,g) and not shaves(g,g)
6. conditionalization (5,1):
     ∃x (barber(x) and ∀y (shaves(x,y) <-> not shaves(y,y)))
     -> (shaves(g,g) and not shaves(g,g))
7. reductio-ad-absurdum (6):
     not ∃x (barber(x) and ∀y (shaves(x,y) <-> not shaves(y,y)))

                              OUTPUT
```

*Assume that there is a barber who shaves everyone who doesn't shave himself (and no one else). Call him Giuseppi. Now anyone who doesn't shave himself would be shaved by Giuseppi. This would include Giuseppi himself. That is, he would shave himself, if and only if he did not shave himself, which is a contradiction. This means that the assumption leads to a contradiction. Therefore, it is false, there is no such barber.*

Figure 2.22 The Barber Paradox (McDonald, 1981, Figures 13 and 14)

Figure 2.22 illustrates output achieved by the original MUMBLE in translating a predicate calculus form of Russell's induction proof, the refutation of the Barber Paradox. This quite impressive text receives its structure from the underlying predicate calculus plan. The generator has semantic knowledge of key domain terminology. For example, the verb "assume" is used for the premise of a proof and the connective "therefore" begins the final utterance that represents the last formula in a proof. In contrast to this generation from predicate calculus, Genero (Conklin, 1983) started from a rule set of descriptive types and visual saliency to organize information about a house which was then realized with MUMBLE. A typical output was:

```
This is a picture of a two story white New England house with a fence
around it. The door of the house is red and so is the gate of the fence.
There is a driveway next to the house and a tree next to the driveway.
In the foreground is a mailbox.
```

In its latest form (Meteer et al., 1987), MUMBLE-86 builds and then traverses a surface syntax tree. It accomplishes this linguistic realization via three closely interacting processes: *realization*, *attachment*, and *phrase structure execution.* "Realization classes" map subcategories onto grammatical constituents. For example the subject-verb-object realization class maps the subcategorization (agent verb patient) onto (subject verb object) for a main, unmarked clause. "Attachment classes", on the other hand, govern the positioning of units in the surface structure, for example splicing a restrictive modifier before a noun. Finally, "phrase structure execution" (PSE), performs a depth-first traversal of the surface structure trees during which procedures are invoked that transform or enforce constraints on the partially ordered tree. PSE also performs morphology and actual output of lexemes.

During PSE, MUMBLE-86 can attach new units to the existing surface structure because legal "attachment points" are indicated on the tree. Thus, MUMBLE-86 can be viewed as employing Tree Adjoining Grammars (TAGs) where an initial phrase structure tree is extended through the inclusion, at very specifically constrained locations, of one or more "auxiliary" trees as detailed in McDonald and Pustejovsky (1985b). Joshi (1987) discusses the relevance and benefits of TAGs to generation including incremental production, constituent movement control, and word order variation. In MUMBLE-86, for example, when providing descriptions in a knowledge based system that contains objects with associated properties, properties can become modifiers of objects even if MUMBLE-86 already has begun constructing a noun group about the object (McDonald and Meteer, 1987). This raises the possibility of interleaving text planning and linguistic realization (see Figure 2.0 at the start of this chapter) as opposed to planning a whole sentence and then realizing it.

MUMBLE-86 has a number of other noteworthy properties. First, the generator is "description-directed," in other words it is guided by the representation of the input and the characteristics of the surface structure tree being constructed. This is in contrast to grammars that direct choices (as with systemic grammars). Second, the process is "indelible," that is decisions are un-retractable (this characteristic is analogous to Marcus' (1980) notion of deterministic parsing.) Third, McDonald claims that the process is psychologically motivated: processing is left to right, incremental, and produces errors similar to human speech errors (McDonald, 1986). Yet another distinguishing characteristic that arises from the use of attachment classes is that "both constituent relationships (including the filler-gap relationship) and linear precedence relationships are defined on the elementary syntactic structures. Adjoining preserves these relationships." (Joshi, 1987, p. 555). Later McDonald (1985b) added mechanisms to his model (COUNSELOR) to enforce prose style. For example, he can encourage complex or simple sentences, compounds or embeddings, and reduced or full relative clauses. While incorporating MUMBLE into multi-utterance planners appears to be a non-trivial task (Rubinoff, 1986), it has been ported to four applications at the University of Pennsylvania (McDonald, personal communication, 1990).

Other work builds on MUMBLE's formalism with the goal of identifying an intermediate level of representation which abstracts "away from the syntactic details of language" (Meteer, 1989, p. 9) and could therefore be used in generation as a layer intermediate between the underlying knowledge representation and MUMBLE's surface syntax tree. MUMBLE would work from this, which would incorporate the knowledge base message content, rather than from the surface syntax tree directly. Thus using MUMBLE as a realization component, the SPOKESMAN (Meteer, 1988, 1989) text generation system has produced a variety of texts including descriptions (e.g., operations orders, simulated radio messages), definitions, and paraphrases. A typical example is the following command post overview of some military units (Meteer, 1989, p. 3):

```
C/1 TB is to the east and its mission is to attack Objective GAMMA from
ES646905 to ES758911 at 141423 Apr.  A/1 TB is to the south.  B/1 TB and
HHC/2 are to the east.[6]
```

SPOKESMAN supports a data structure, called the text structure, which represents a tree structure consisting of nodes that are marked by their function with respect to their parents and siblings, such as coordinate, matrix, adjunct, head, or argument. For example, HEAD-ARGUMENT structure (termed a KERNEL tree), as well as a COMPOSITE of a mandatory MATRIX element and some optional supporting element (e.g., a complex noun phrase or a paragraph with a topic sentence), are illustrated in Figure 2.23. In examples (Meteer, 1989, p. 15) the MATRIX is always an EVENT composed of an action and its ARGUMENTS, though in principle "this structure is simply a constituent that is made up of one main subconstituent and zero or more subconstituents elaborating the main one. This kind of structure appears at all levels of text, from a paragraph with a topic sentence, to a main clause with subordinate clauses, to a head noun modified by adjectives" (Meteer, 1989, p. 17).

While much current research in text generation assumes sentences as the primitive units of texts (e.g., McKeown, 1985; McCoy, 1985ab; Maybury, 1987; Paris, 1987ab; Hovy, 1988a; see section 2.6), Meteer (as well as Appelt, 1985) recognizes that relationships cross sentence boundaries. For example, the syntactic notion of main clause and subordinate clause, as well as nucleus and satellite relationships between larger chunks of text (see discussion of Rhetorical Structure Theory (Mann & Thompson, 1987) in Chapter 3), are encompassed by SPOKESMAN's representation of matrix and adjunct knowledge. Furthermore, intraclausal structures such as clauses with adverbials or noun phrases with modifiers are also captured by the matrix and adjunct relations. This unifying representation allows the text planner to distribute information to the most effective surface position, independent of clausal or sentential boundaries. This is in contrast to other text planning work (e.g., Hovy, 1987) which is restricted to ordering and adjoining input units supplied as message content and cannot merge them together or distribute their elements to other parts of the text either in planning or realization.

[6] The following abbreviations apply: TB - tank battalion, A/1 TB - 1st company of the 1st tank battalion, B/1 TB - 2nd company of the 1st tank battalion, C/1 TB - 3rd company of the 1st tank battalion, HHC- Headquarters unit.

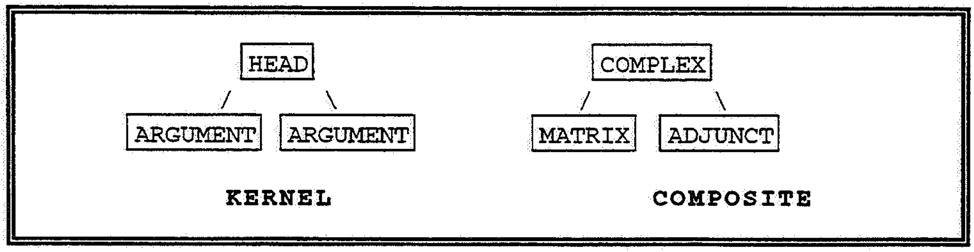



Figure 2.23 Text Structure Trees: Kernel and Composite (Meteer, 1989, Figure 3.1, p. 25)

Meteer's work thus embraces a more fundamental issue concerning levels of processing in generation. While much initial work in generation assumed that (sentential) content planning and linguistic realization were sequential processes (see Figure 2.0), Meteer's work takes the view that content selection and linguistic construction are concurrent. Similarly, Mellish (1987) developed a system (for generating natural language instructions) that examined and organized text elements into messages at a minimally linguistic level of abstraction and then allowed for "structure building" rules to incorporate additional portions of "local" messages into linguistic structures. At a more general level, Hovy (1988b) suggested *limited-commitment planning* whereby top-down "prescriptive" goals (e.g., convince the hearer of a proposition or compare two entities) activate "prescriptive" text plans which are tempered by bottom-up "restrictive" goals (pragmatic and stylistic choices like impress the hearer or make them feel socially subordinate) which guide linguistic realization (e.g., lexical and syntactic structure choice). The precise relationships of these levels of processing remains an open research issue.

Once a sequential approach to text planning and realization is discarded, this raises the potential for revision of content and linguistic form during generation. Meteer (1988) analyzed some revisions of actual text (e.g., replacing a weak verb and direct object with a strong verb as in "make a decision" -> "decide"). She identified three major categories of revision (restructuring the text, making it more concise, and making it more explicit) and then suggested how her notions of text structure (e.g., matrices and adjuncts) could be exploited for text revision. De Smedt and Kempen (1987) discuss psycholinguistically-motivated models of revision and identify three key processes: deletion, replacement, and reformulation of linguistic structures. These notions remain to be investigated computationally and could prove to be invaluable in increasing generator fluency.

### 2.6.3 Realization as Choice: Systemic Grammar and PROTEUS

In contrast to MUMBLE's tree adjoining grammar approach, systemic grammars (Halliday, 1976), formalized in (Patten and Ritchie, 1987), attempt to model a network of systems which encode choices among grammatical features such as number and mood. Each single choice point is a grammatical system which allows for a choice among grammatical features. Grammatical systems are connected together to

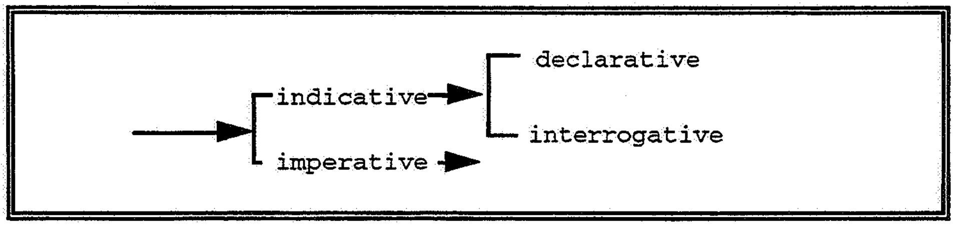



Figure 2.24 Systemic Network

form *system networks*. Figure 2.24 shows a fragment of a network containing two systems which classify clause types.

Just as BABEL travels through discrimination nets to select lexemes, systemic generators traverse a network of choices accumulating features as they go by using *inquiries* which interface between syntax and other knowledge to provide semantic, pragmatic, and even extra-linguistic information to guide decisions. They use procedural realization rules to take the resulting features and produce grammatical structures. Because systems are activated only when required, they tend to be more efficient than sequentially accessed phrase structure grammars.

Under this formalism, PROTEUS (Davey, 1978) produced commentary while it played tic-tac-toe. Davey's generator started with propositions from a record of moves in the tic-tac-toe game, for example, `[<proteus> 3]`, i.e., the computer took the top right hand corner of a 3 x 3 board numbered left to right, top to bottom. Using a representation of the state of the game and game tactics (e.g., "counter-attack" and "foiled-threat") this proposition is translated into the message: `[<proteus> <square 3> start <game> take <square 3>]`. To realize this message, PROTEUS first traverses its systemic grammar to obtain the features of the output sentence as a whole, here `(CLAUSE SIMPLE FINITE REMOTE TRANSITIVE ...)`. For efficiency, PROTEUS defaults to utterances that are INDEPENDENT, INDICATIVE, DECLARATIVE, and PAST TENSE (Patten, 1988, p. 135). Three levels of processing (feature translation, grammatical structure building, and function realization) then transform a message into a set of constituent feature-sets, in this case `[Ng subject] [Lexical Vg Remote Tensed] [Object Ng] [Prepg By]`. Constituent group specialists then construct the final surface form: "`I started the game by taking a corner.`" Unfortunately, some of PROTEUS' detail is rather unsatisfactory. Thus after the first two transformation processes the resulting feature-value pairs, in the above example `((SUBJECT INITR) (PROCESS FINITE PAST) (ACTOR POSTVERB) (APPENDIX BYOBJ))`, intermix a wide range of information[7] including (Houghton, 1988): syntactic (POSTVERB indicates object position), lexical (BYOBJ results in "by doing ..."), and morphological (FINITE denotes a tensed verb).

[7]Systemists argue that this interleaving is deliberate.

Textuality in PROTEUS' output arises from (1) the organization of the underlying tic-tac-toe program, (2) the grouping of related predicates into not more than three clauses per sentence, and (3) the use of connectives (e.g., "and", "however", "but") to ensure local cohesion. These mechanisms help PROTEUS produce the following commentary (Davey, 1979, p. 17):

```
The game started with my taking a corner, and you took an adjacent one.
I threatened you by taking the middle edge opposite that and adjacent to
the one which I had just taken but you blocked it and threatened me.  I
blocked yours and forked you.  Although you blocked one of my edges and
threatened me, I won by completing the other.
```

Thus while some of the linguistic realization details were unsatisfactory, PROTEUS produced some of the most fluent text yet generated by machine, in part because the underlying tic-tac-toe application allows an 'obvious' sequential structure to the text.

### 2.6.4 Nigel: Representing Ideational, Interpersonal and Textual Meaning

Nigel (Matthiessen, 1981; Mann and Matthiessen, 1983), a linguistic realization component developed for the PENMAN text generator (Mann, 1983), has perhaps the broadest grammatical coverage of any systemic generator to date (although Fawcett (1988) claims a systemic grammar with even broader coverage). Nigel travels down the system network to produce a collection of syntactic features by consulting a "chooser" at each branch in the system. The *choosers* (between 200-300 in the Nigel grammar (McKeown and Swartout, 1987)) can consult a wide range of linguistic and extralinguistic knowledge, using *inquiries*, when making decisions. In particular, Nigel represents three classes of meaning (termed the "metafunctions" of text): "experiential ideational" (i.e., referring to the knowledge base), "interpersonal" meaning (referring to the relationship of the speaker to the audience and the speaker to the subject), and finally "textual" meaning (referring to a discourse model). These three metafunctions are reflected in language (Bateman, 1988, p. 126):

> Experiential ideation strongly favors 'building block', constituency-style organizations; interpersonal meanings strongly favor 'prosodic' organizations that persist over stretches of text; and textual meanings favor 'pulse'-style organizations that may cut across the constituency and prosodic strands of organization.

So, for example, when identifying a referent in discourse a chooser may select among a variety of options (e.g., a definite or indefinite noun phrase, a pronoun, or by deixis), by using inquiries to examine (Matthiessen, 1987):

- the knowledge base to see if the object is an individual or an instantiable class
- the user model to see if the hearer knows the referent
- the text (discourse) history to see if the object is given or new

- information about the setting of the speech
  (e.g., allowing exophoric reference "Fudge Brownies: Heat the oven to 350°F".)

There are currently over 600 inquiries (Bateman, personal communication, 1990), some of which must be tied into each new application. PENMAN has been designed to provide as much of this support knowledge as possible in an application-independent manner (e.g., by having current text planners control textual inquiries and by providing interpersonal and ideational defaults). The choosers in the system network communicate via the inquiries to the *upper structure*, a knowledge representation based on KL-ONE object and events.

By examining constraints on linguistic realization that go beyond the level of syntax (e.g., ideational, interpersonal, and textual), systemics addresses a major unsolved problem in text generation: the nature of the interface between the text planner and the realization mechanism. Should their processes be sequential, parallel, or interleaved? Recent work has taken the first steps toward an integrated representation of information from that about clauses to that for multisentential text. Bateman (1985) investigated the realization of intersubjective effects in discourse. Patten (1988) included some semantic knowledge about carpentry tasks in the systemic formalism in SLANG (Systemic Linguistic Approach to Natural-language Generation), though his work was still limited to single utterance realization such as "first you do the painting." More recently, Matthiessen (1987) and Bateman (1988) have been investigating the extension of systemic functional linguistics to investigate utterances produced in their social context. Notwithstanding the problem of controlling the processes of text planning and realization, the systemic formalism holds promise as a common paradigm in which to both plan and realize text.

### 2.6.5 Functional Unification Grammar

In contrast to a systemic network description of choices, Functional (Unification) Grammars (FUG) (Kay, 1979) encode functional information as attribute-value pairs in the grammar. This is analogous to feature-value pairs found in Generalized Phrase Structure Grammar (GPSG) (Gazdar, 1982). Bossie (1982) incorporated functional categories such as topic, comment, and focus into the sentence generator for TEXT (McKeown, 1985). Appelt (1983) used FUG in his planner TELEGRAM. Unfortunately, the production of syntactic structure involves unifying an input message against the grammar which contains functional specifications. While the grammar can be simplified because of the explicit encoding of functional constraints (e.g., using metarules to govern constraint application over several rules), unification can be inefficient in comparison to deterministic systems of choice (as in systemic grammars) or MUMBLE's left-to-right, indelible realization process. For example, a sixty-fold speed up was achieved when MUMBLE replaced the FUG sentence generator in TEXT (Rubinoff, 1986). In fact, Ritchie (1986) has shown FUGs to be NP-complete, although McKeown and Paris (1987) claim a FUG implementation with efficiency similar to MUMBLE. However, regardless of their efficiency, functional grammars continue to be a clear method for expression of grammatical knowledge.

### 2.6.6 Bidirection

A chief benefit of declarative linguistic knowledge is its independence from process (be it parsing, generation, paraphrase, or translation). A number of researchers (Appelt, 1987; Jacobs, 1988; Shieber, 1988) have suggested general bidirectional architectures (i.e., for parsing and generation). This is distinct from a stronger sense of process bidirection suggested by Kay (1980) in which generic processing strategies (termed algorithm schemata) operate on common grammars to both generate and parse using a chart data structure (i.e., a well-formed substring table). The principal advantages of using bidirectional grammars are economy of representation (i.e., no duplication of grammatical knowledge), and consistency of linguistic coverage in input and output. While declarative grammars have been present in many systems in the past, only a few systems have used them for both parsing and generation. Simmons and Chester (1982) generated sentences using bidirectional grammars in PROLOG. We have already discussed PHRED and PHRAN's use of a common "pattern-concept" pair for both parsing and generation. GENNY (Maybury, 1987b) exhibited bidirection by producing text using the same (GPSG inspired) unification grammar and dictionary used previously in a parsing system for knowledge base query (Maybury, 1987a).

Levine and Fedder (1989) have also investigated bidirection. As in Maybury's (1987ab) grammar (discussed later in Chapter 8), each phrase structure rule (e.g., `S → NP VP`) is augmented with features (e.g., `count`) which can be bound to particular values (e.g., `singular`, `plural`). This allows for efficient coding of constraints, as in the syntax rule `S[tense` *t*`] → NP[count` *x*`] VP[tense` *t*`, count` *x*`]` (where feature variables for tense and count are italicized). Associated with each syntax rule is a semantic rule based on Montague semantics (Montague, 1974). In contrast to Maybury's grammar, Levine and Fedder's syntax grammar allows for a number of discourse features including theme (the initial entity in the syntactic structure or "immediate focus" of attention), linguistic focus ("the contextually non-bound (i.e., new) portion" of an utterance, taken to be the final noun phrase), and emphasis ("a Boolean flag which is set to true when special emphasis is applied on the linguistic focus of a sentence by the use of an it-cleft"). Two distinct processes use these bidirectional syntactic and semantic rules to interpret and generate the following dialogue about Cambridge colleges (Levine, 1989):

Q: When was King's founded?
A: It was founded in 1441.
Q: Was Kings founded by Wren?
A: No, it was founded by Henry the Sixth.
Q: Did Henry the Sixth found Trinity?
A: No, it was Henry the Eighth who founded Trinity.
Q: Is Trinity open today?
A: No, it's open on Wednesday and Saturday.

Levine (1990, forthcoming) is presently investigating extending the notion of bidirection to include plan-based communication which shares knowledge about physical and linguistic actions in the domain and the beliefs and plans of the user. The aim is to move beyond simply question-answer pairs to the production of longer stretches of discourse.

## 2.7 Discourse Strategies for Structuring Text

The previous section examined a number of techniques for linguistic realization including lexical choice, phrase construction, and sentence generation (the tactical stage in Figure 2.0). Other research has focused on planning multisentential text (the strategic stage in Figure 2.0) and on how linguistic realization is influenced by the context in which an utterance occurs. In particular, research has keyed in on developing mechanisms to select information and then to focus, group, and order it over longer stretches of text.

Initially, Mann and Moore (1981) suggested a "fragment-and-compose" approach whereby grouping and ordering are performed independently. First the message (the representation of content) is divided into elementary propositions. Next, these are ordered using rules of aggregation (e.g., chronology). The resulting possible orderings are evaluated by means of preference values and the best organization is selected. This idea of structure evaluation plays an important role in the production of text.

To model longer texts for output planning purposes, however, researchers turned to techniques that could represent the structure of texts as wholes. In contrast to early bottom-up approaches, the text structure view can be characterized as essentially top-down. BLAH and TEXT were the first systems of this sort, producing connected, multi-utterance text by following models of text structure guided by local focus constraints.

### 2.7.1 Weiner's "Explanation Grammar"

With the goal of structuring texts so that they are easier to understand, Weiner (1980) pioneered the method of analyzing naturally occurring explanations in order to gain insight into strategies that people seem to apply to structure information. He found that humans justify statements by offering reasons, showing conditionality (if X then Y), providing supporting examples, or showing that all other alternatives are implausible. He formalized these ideas in an "explanation grammar" which represents several relations among propositions, including IF/THEN, STATEMENT/REASON, GENERAL/SPECIFIC, AND, and OR. He then implemented a system, BLAH, which constructed hierarchical trees of propositions and relations from the explanation grammar rules shown in Figure 2.25.

```
             ________[STMT/RSN]________
          /                            \
Peter is a                   _________[AND]______________
dependent of                /           |                 \
Harry's              [STMT/RSN]  [GENERAL/SPECIFIC]    [STMT/RSN]
                     /      \        /        \          /      \
        Peter makes     Peter    Peter is   Peter's   Harry     Harry provides
        less than $750  does     under 19   age is    supports  more than 1/2
                        not work            15        Peter     of Peter's support
```

Figure 2.26 BLAH Explanation Tree

e -> AND e e $(e)^n$
e -> OR e e $(e)^n$
e -> STMT/RSN e e
e -> THEN/IF e e
e -> GENERAL/SPECIFIC e e
e -> EXAMPLES e e e $(e)^n$
e -> ALT e e $(e)^n$
e -> simple text

where e indicates a primitive expression (or proposition) and
$(e)^n$ means one or more occurrences of expression e.

Figure 2.25 BLAH's Explanation Grammar (Weiner, 1980, p. 24)

Weiner recognized that previous systems (e.g., the Digitalis Advisor) simplistically measured the degree of detail in an explanation as equivalent to the depth travelled in the underlying reasoning tree. Unfortunately, this assumes a hierarchical and a well-written knowledge base. In contrast, BLAH achieves more appropriate content by suppressing inferable propositions, deleting non-nuclear propositions (e.g., you can provide a statement without giving the reason for that statement) and breaking up the overall explanation into smaller components which would appear in the output text as smaller sentences than the earlier "rule translations". BLAH infers what the user knows by keeping an independent model of the assertions and rules that the system and user know. Hence BLAH, in a well-motivated manner, conforms to Grice's (1975) Maxim of Quantity: "Do not make your contribution more informative than is required."

To see how this process results in clearer text, we consider a detailed example in the domain of US income tax law (Weiner, 1980, Figure 12, p. 33). We begin with the explanation tree after non-relevant information is pruned away as shown in Figure 2.26.

Below is the text produced from reading this explanation tree depth-first in a straightforward way. Individual propositions are realized using shallow templates (analogous to Winograd's (1972) fill-in-the-blank templates) associated with each proposition type.

```
Peter is a dependent of Harry's because Peter makes less than 750
dollars because Peter does not work and Peter is under 19, in fact Peter
is 15, and Harry supports Peter because Harry provides more than half of
Peter's support.
```

Notice that connectives such as "because", "and", and "in fact" are used to indicate relationships between propositions. Unfortunately, this text would be confusing to a reader who did not have the benefit of the graph in Figure 2.26. Information has been lost in the linearization of the graph. This is exacerbated by the multiple embedding of justifications.

To minimize information loss and the confusion associated with multiple embeddings, BLAH structures the explanation tree. First it breaks up complex trees into smaller components which can be presented in increments. Then it determines the linear order of terminal nodes (which represent focus), to ensure connected surface form. BLAH indicates subordinated nodes through the use of structure markers such as "uh", which indicates a shift in focus. This additional processing results in the following improved text (Weiner, 1980, p. 34):

```
Well, Peter makes less than 750 dollars, and Peter is under 19, and
Harry supports Peter so Peter is a dependent of Harry's.  Uh Peter makes
less than 750 dollars because Peter does not work, and Peter is a
dependent of Harry's because Harry provides more than one half of
Peter's support.
```

So unlike previous systems which simply traced some underlying knowledge structure to generate text, Weiner's BLAH is guided by descriptive strategies. The relations among propositions that Weiner provides, however, are few and too general and, as a result, he produces only a small class of expository texts. His system cannot define terminology, describe a process, compare or contrast objects, or persuade the hearer to do something. As we discuss next, McKeown (1985) and others specify a richer class of relations among propositions which enables them to structure texts in more varied ways.

### 2.7.2 McKeown's "Constituency Schema"

Like Weiner, McKeown (1982, 1985ab) analyzed short samples of descriptive text to discover discourse strategies that humans use to identify, describe, and compare objects. She implemented her ideas in TEXT, a system that generated textual responses to questions about the Office of Naval Research (ONR) database on ocean vessels. McKeown identified three types of user requests to the ONR database: requests for definitions, requests for available information, and requests for the difference between two objects (from McKeown, 1985a, p. 41). These three requests were represented in TEXT as invoking the communicative goals *define*, *describe*, and *compare*. Associated with each of the request types were one or two general discourse strategies (shown below) from a set of four *text schemas*: identification, constituency, attributive, and compare and contrast.

- Requests for Definitions
  - identification
  - constituency
- Requests for Available Information
  - attributive
  - constituency
- Requests About the Difference Between two Objects
  - compare and contrast

Each text schema consisted of a sequence of sentence types corresponding to Grimes' (1975) *rhetorical predicates* (e.g., attributive, constituency, analogy).

One of the four strategies that McKeown identified in natural text is characterized by the *constituency schema*, which describes an object using the steps shown in Figure 2.27. For example, when the system is asked, "What is a guided projectile?" (the user actually types in `(definition GUIDED)`), the constituency schema is used to generate the following text (McKeown, 1985a, p. 30):

```
A guided projectile is a projectile that is self-propelled. There are 2
types of guided projectiles in the ONR database: torpedoes and
missiles. The missile has a target location in the air or on the
earth's surface. The torpedo has an underwater target location. The
missile's target location is indicated by the DB attribute DESCRIPTION
and the missile's flight capabilities are provided by the DB attribute
ALTITUDE. The torpedo's underwater capabilities are provided by the DB
attributes under DEPTH (for example, MAXIMUM OPERATING DEPTH). The
guided projectile has DB attributes TIME TO TARGET & UNITS, HORZ RANGE &
UNITS, and NAME.
```

Unlike BLAH, TEXT could choose between alternative strategies to define or describe an object based on the object's location in the knowledge base generalization hierarchy. For instance, if the user asks for the definition of a ship, instead of choosing the "constituency schema" (which details subparts), TEXT

1. Identify the object, its class, and distinguishing attributes
   IDENTIFICATION predicate
2. Present the constituents of the item (subparts or subentities)
   CONSTITUENCY predicate
3. Present characteristic information about each constituent in turn
   DEPTH-ATTRIBUTIVE predicate
4. Present additional information about the item to be defined
   ATTRIBUTIVE predicate

Figure 2.27 McKeown's Constituency Schema selects the "identification schema" (which details defining characteristics) because ship occurs below a predetermined level in the hierarchy (McKeown, 1985b, p. 29).

Figure 2.28 indicates the principal processes (in rectangles) and knowledge sources (in ovals) of McKeown's system. Based on the user's question, TEXT first selected a subset of the knowledge base termed the *relevant knowledge pool* (analogous to Grosz's (1977) notion of global focus). For example if the user asked TEXT "What is a ship?" (actually typed in functional notation `(definition SHIP)`), the system would select the database attributes, relations, superordinates, and subordinates of ship in the knowledge base hierarchy (in some examples siblings and their subordinates are also included). Next, a text schema (e.g., Figure 2.27) was selected based on the discourse goal (define, describe, or compare) and the amount of information in the relevant knowledge pool, as is the choice between constituency and identification for a ship. Walking through the selected schema of rhetorical predicates (represented as an ATN), TEXT selected individual rhetorical propositions (rhetorical predicates instantiated with information from the knowledge base) constrained by ordered rules of local focus shift (Sidner, 1983):

1. Shift focus to an entity mentioned in the previous proposition
2. Maintain the focus in the current proposition
3. Return to the topic of a previous discussion
4. Select a proposition with the greatest number of implicit links to

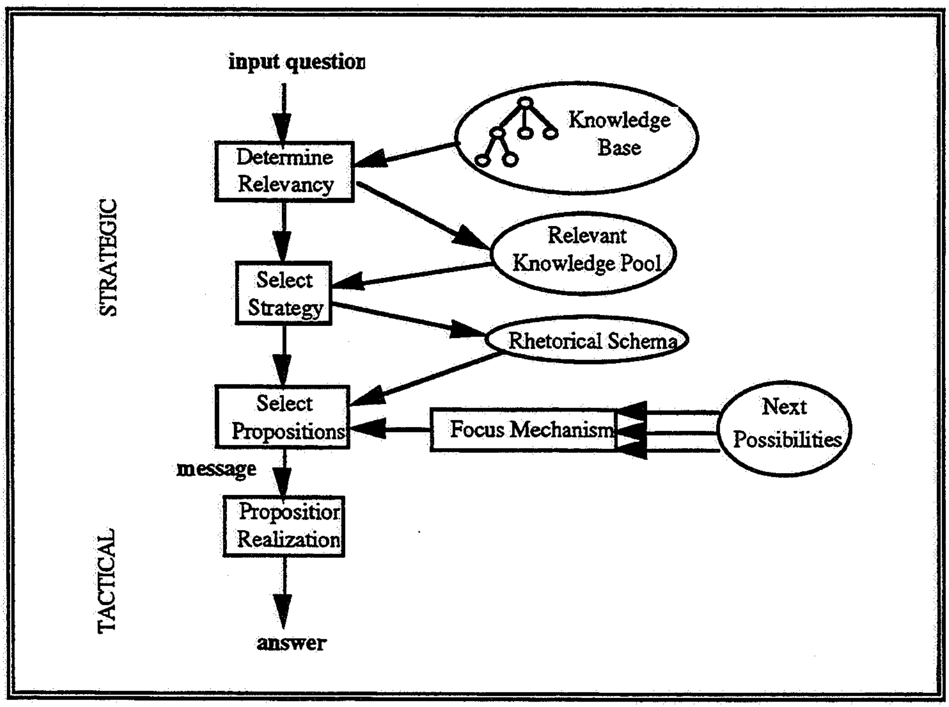



Figure 2.28 TEXT System Overview

(i.e., entities mentioned in) the previous proposition.[8]

Unlike Weiner's BLAH which considered an entire proposition as the focus, TEXT identified the arguments of a predicate as focused entities. Another distinguishing feature of TEXT is that, unlike BLAH, content selection (not just realization) is guided not only by discourse structure but also by an explicit model of local and global focus. Furthermore, McKeown allows focus information to be used to select individual rhetorical predicates during strategic generation to influence syntactic structure in realizations (e.g., pronominalization, active versus passive voice, there-insertion). TEXT's tactical component (Bossie, 1981) translates rhetorical predicates into English using a functional grammar, based on Kay's (1979) formalism. A final important characteristic is that the rhetorical predicates in TEXT were connected to the database via a "predicate semantics" so, in principle, TEXT was independent of the particular knowledge representation formalism (although it was tested only in the ONR domain).

The underlying application had to be enhanced for TEXT to produce its output. McKeown developed a meta-level representation of the ONR database schema which included a generalization and attribute hierarchy. While parts of this were automatically generated (McCoy, 1982), distinguishing descriptive attributes -- properties of an object that distinguish it from its siblings and thus provide crucial extra information to support explanation -- had to be hand encoded for each of the objects. This was effortful, but is not necessarily a defect of TEXT: it is difficult to automatically generate rich knowledge bases from the sparse knowledge represented in conventional database systems. Nevertheless, as detailed in the discussion in Chapter 4 concerning logical definition, it is possible to automatically select the features of an entity in a generalization hierarchy that distinguish it from its relatives.

McKeown's generator is unable to produce ellipsis, informal phraseology, or stylistic variation. But she has extended her model to tailor explanations for the user (McKeown et al., 1985) in an advisory system for course selection. She is currently investigating the integration of graphical explanations and their relation to discourse schema that instruct or provide directions (McKeown, 1989).

### 2.7.3 Paris' "Process Trace"

While BLAH and TEXT used top-down strategies and focus to control generation, Paris (1987), after examining a variety of texts from children's encyclopedias, recognized that texts are often organized around a trace of some underlying process, though in the behavior of an external entity rather than in the reasoning chains of expert systems. Paris' *process trace* (Paris, 1987b, p. 59) describes the function of a complex physical object by exploiting the causal connections in the underlying knowledge base to order rhetorical propositions. Figure 2.29 details the process trace algorithm. To illustrate the distinction from McKeown's work above, we consider two texts (both produced by Paris' system) that describe the same

---

[8]Rule #4 is not founded on obvious linguistic principles. It was added by McKeown after empirical tests necessitated its use.

object, a microphone. The first text (Figure 2.30) was generated using the constituency schema and describes the component parts and attributes of the microphone.

1. Find the main sequence of events that take place when the object performs its function: the MAIN PATH. An algorithm (p. 82) computes this MAIN PATH by distinguishing three connections, in the following order of preference:

a. control links (cause, enablement, or interruptions)
b. temporal links (e.g., A before B, Y after X)
c. analogical links (equivalence, correspondence)

Then starting with the first event in MAIN PATH:

2. Follow the next causal link in this sequence
3. Provide ATTRIBUTIVE information about a subpart just introduced (optional)
4. Include a side link related to the MAIN PATH (optional)
5. Follow substeps (optional)
6. Return to 2 until at the end of MAIN PATH.

Figure 2.29 Paris' Process Trace

```
The microphone changes soundwaves into current.  The microphone has an
aluminium disc-shaped diaphragm, and a doubly-resonant system to broaden
the response.  The diaphragm is clamped at its edges.  The system has a
button, and a cavity.
```

Figure 2.30 Structural description of a microphone (constituency schema)

In contrast, the second text (Figure 2.31) was produced from the same knowledge base, but this time by following the process trace algorithm as described above. While the constituency text in Figure 2.30 might be appropriate for an expert user who could fill in the causal interconnections between subparts, the process trace output of Figure 2.31 is more appropriate for someone who is unfamiliar with the mechanism and needs to be explicitly told how it works.

```
The microphone changes soundwaves into current.  A person speaking into
the microphone causes the soundwaves to hit the diaphragm of the
microphone.  The diaphragm is aluminum, and disc-shaped.  The soundwaves
hitting the diaphragm causes the diaphragm to vibrate.  The
soundwave_intensity increasing causes the diaphragm to spring forward.
The diaphragm springing forward causes the granules to compress.  The
granules compressing causes the resistance of the granules to reduce.
The resistance reducing causes the current to increase.  The
soundwave_intensity reducing causes the diaphragm to spring backward.
The diaphragm springing backward causes the granules to decompress.  The
granules decompressing causes the resistance to increase.  The
resistance increasing causes the current to reduce.  The diaphragm
vibrating causes the current to vary.  The current varies like the
intensity varies.
```

Figure 2.31 Functional description of a microphone (process trace)

It is evident that the two texts are quite different both in content and structure. Paris' work implicitly indicates the important idea that selection and ordering of content of the knowledge base should be controlled by a variety of knowledge sources, not only models of text and focus. In the main body of this dissertation, a range of sources which contribute to the formulation of texts is discussed.

Although not a key goal of Paris' work, it should be noted that Paris' characterization of the connections between events in her MAIN PATH is not exhaustive. For example, spatial as well as temporal organization is central to many texts. In addition, many other organization strategies are possible such as order of importance, from least to most obvious, or from least to most probable. Also, the structural/functional organization choice is made solely in relation to user expertise while a variety of sources might influence this, such as the current task or goal and the discourse context.

### 2.7.4 Problems with the Text Schema Approach

One of the chief problems with the approach of BLAH and TEXT is the inflexibility of schemata of rhetorical predicates. This is manifest in the repetitive structure of the resulting text. Another problem is that the rhetorical predicates which serve as the building blocks of texts often conflate or vary their emphasis on several distinct properties including linguistic form, content, communicative role, and discourse relation. For example some predicates are closely tied to linguistic form as in analogy, which is typically realized as a simile (e.g., "Planes are like birds."). Other predicates, however, have more to do with ontology as in the constituency predicate, which deals with part/sub-part information (e.g., "Planes have wings). Still other predicates identify the communicative role or communicative act underlying an utterance as in a concede predicate (e.g., "I agree that planes have wings."). Finally, some rhetorical predicates are identified by their relation to previous discourse, as in a counter-claim predicate (e.g., "On the other hand, ...").

As we will encounter in the next chapter, Rhetorical Structure Theory (Mann and Thompson, 1987) emphasizes the latter discourse relation property (termed *rhetorical relations*) and the intended effect of utilizing these relations on the addressee. In fact, the failure to account for the intended effects of uttering different kinds of propositional content is an equally significant flaw of BLAH, TEXT, and TAILOR. The effect on the addressee is crucial to the success of an explanation. For example, providing motivation for an action or event increases the desire of the addressee to perform that event. Similarly, providing the justification underlying a causation (e.g., "The nail caused the bicycle tire to pop because it is a sharp object.") may increase the addressee's belief in the causation. So too, clarifying the purpose of an entity or an action (e.g., "You can use a flat-head screwdriver to manipulate flat-head screws, scrape paint, or chisel wood.") may increase the addressee's willingness and/or ability to use an object or to perform some action. The failure of current generators to incorporate such knowledge is addressed by TEXPLAN's text planning approach which represents and reasons about the structure, focus, and perlocutionary effects of text, and delineates among the various characteristics of text including linguistic form, content, communicative acts, and rhetorical relations.

## 2.8 Tailoring Explanations to the Addressee

In the initial section on explanation representation, we saw how explanation content could be selected by controlling the degree of abstractness of the explanation (e.g., NEOMYCIN, XPLAIN) and later how a greater variety of explanations could be addressed by adding support knowledge, such as terminology, methodology, and intent (e.g., EES). During the explanation representation investigations, it became apparent that it would be necessary to tailor not only the content but also the language and the form of an explanation to the system user. This section discusses efforts to tailor lexical choice, grammatical structure, rhetorical form, and perspective. It concludes by examining some reactive feedback mechanisms recently developed to recover from miscommunications in explanatory dialogue.

As this section concentrates on work geared to producing explanations and on generation of multisentential text, it does not address many dialogue issues. Cooperative response in question answering has been investigated since Kaplan (1982) and is overviewed in Webber (1987b). Cawsey (1989) considers the generation of explanatory dialogue. However, this section does assume that the addressee is individuated at least by particular features of the prior discourse (e.g., goals, focus of attention), but probably also by user class (e.g., novice) or other personal information (e.g., age) acquired and represented in some way as in (Wilensky et al., 1988; Kass and Finin, 1988; Carberry, 1988; Quilici et al., 1988).

### 2.8.1 Lexical and Grammatical Variance

The most obvious and straightforward way to tailor output is to select words and syntactic structures which are consistent with the type of individual being addressed. MYCIN could select words based on the type of user (e.g., "triggers" for an engineer versus "is strongly associated with" for a medical professional.), but in a fixed way without reference to local discourse context. Appelt (1985) varied referents based on a model of the user's knowledge and the context of previous utterances and Hovy (1987) selected words and phrases based on assumed models of speaker/hearer/topic relationships.

### 2.8.2 Varying Rhetorical Form

In addition to altering words and surface structure, researchers recognized the advantage of varying text structure. As we discussed in the section on TEXT, McKeown was able to select different rhetorical schema to structure a response based on the type of communicative goal (e.g., define, describe, or compare) she assumed input processing had identified. Moreover, TEXT was able to choose between the constituency schema and the identification schema to define an object based on pre-defined levels at which the object appeared in the generalization hierarchy.

Paris used an assumed model of the user's knowledge to decide how to structure the information in her TAILOR (1987) system. Paris' process trace (Figure 2.29), in contrast to the structural orientation of McKeown's constituency schema (Figure 2.27), ordered propositions guided by the causal connections in the underlying knowledge base. But after implementing this, Paris developed a decision algorithm which could decide when to choose between process and constituency orderings of text (Figure 2.32). On the basis of analysis of children and adult encyclopedia explanations of complex physical objects, Paris (1987) found that the adult entries were typically constituency based (presumably because adults could infer functional relations from structural descriptions), whereas children encyclopedia explanations typically traced the underling process of the complex physical object.

```
1) Is there a mechanism associated with the object to be described?
   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 the 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 the process trace
```

Figure 2.32 Paris' Schema Selection Algorithm (Paris, 1988, p. 21, Figure 7)

Then in a further refinement, Paris placed "decision points" within McKeown's constituency schema to allow the generator to switch to the process trace while executing the constituency schema, to achieve more finely tailored output. Thus after the IDENTIFICATION predicate mentions the superordinate of an object, a process trace of this superordinate can be provided, and after the CONSTITUENCY predicate mentions a subpart of the object, this subpart can be described using the process trace. Similarly, Paris added a decision point to her own process trace allowing it to call McKeown's constituency schema after mentioning ATTRIBUTIVE information about a subpart just introduced.

Figure 2.33 shows the result of mixing discourse strategies in this way. The example describes a telephone which has two parts: a transmitter (an instance of a microphone) and a receiver (an instance of a loudspeaker). If a user has local expertise about the receiver but not about the transmitter of a phone, then TAILOR will start generating with the constituency schema (because the user knows one of the two components). However, it will switch to the process trace (*italicized text*) when it begins to discuss the transmitter since the user model indicates that the user has no expertise about it. The result is a text tailored to the user's particular knowledge of subparts. (Sentences 7-13 in Figure 2.31, which describe the functioning of the microphone, disappear in Figure 2.33: it is unclear if the exclusion is based on the user model, conciseness considerations, or other criteria.)

Although it produces shorter responses, UNIX Consultant (UC) (Wilensky et al., 1988) is also able to vary rhetorical form based on the expertise and knowledge of the user (as inferred from the dialogue). UC can "format" content using patterned responses for definition, example, or simile. In the first case a heuristic is used that finds a category dominating the one in the query, and then finds information that distinguishes the category from it (although the latter procedure is not detailed). A typical definition is:

```
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 with a small diaphragm. 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 diaphragm vibrating 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 2.33 Constituency Schema and *Process Trace* (Paris, 1988, Figure 10, p. 26)

```
U1:  What is a directory?
S1:  A directory is a file that is used to contain files.
```

The second type of "rhetorical format" is a simile which expresses concepts in terms of other concepts that the user is believed to know:

```
U1: What does ruptime do?
S1: Ruptime is like uptime, except ruptime is for all machines on
    the network.
```

In addition to giving definitions and similes, if the user is a novice UC can use examples.

```
U1: What can I delete a file?
S1: Use rm.
    For example, to delete the file named foo, type 'rm foo'.
```

Linguistic realization in UC is based on patterned templates like "To `(gen goals)` comma `(gen plan)`" which mix actual words ("To") and punctuation ("comma") with function calls (e.g., `(gen plan)`) to produce text such as "To delete a file, use rm". These are similar to Winograd's (1972) more complex templates. Unlike Paris' TAILOR, UC is based on a constructed and not assumed user model, although UC's rhetorical variance is much more primitive than that produced by TAILOR. UC illustrates that simple rhetorical techniques can be employed, based on the user's expertise and knowledge, to express ideas more effectively.

### 2.8.3 Tailoring using Pragmatic Information: ERMA and PAULINE

In contrast to rhetorical variance, a number of researchers have investigated increasing the sensitivity of textual output to the user and the situation by exploring pragmatic models of conversation. A pioneer system, ERMA (Clippinger, 1974), modeled false starts, hesitations, and suppressions characteristic of speech by incorporating a series of sophisticated modules. These included CALVIN (topic collection and filtering), MACHIAVELLI (topic organization and phraseology), CICERO (realization), FREUD (monitoring the origins of rhetorical plans), and LEIBNITZ (a "concept definition network"). While some of their functions clearly include issues addressed previously, others suggest a much broader influence on text (e.g., self monitoring).

PAULINE (Planning and Uttering Language in Natural Environments) (Hovy, 1987) has been viewed as a parameterization of ERMA, for written output. PAULINE characterizes conversational setting in terms of conversational atmosphere (the speaker, the hearer, the speaker-hearer relationship), and characterizes the interpersonal goals of the hearer and the speaker-hearer relationship. In a particular discourse, for example, the speaker is represented in terms of his knowledge of the topic (expert, student, novice), interest in the topic (high, normal, low), opinions of the topic (good, neutral, bad) and emotional state (happy, angry, calm). Depending on these parameters, PAULINE, in the role of speaker, generates pragmatically distinct text. For example, PAULINE produces the following when acting as a student describing events that took place at Yale University in April, 1986 (Hovy, 1987, p. 8):

```
I am angry about Yale's actions.  The university had officials destroy a
shantytown called Winnie Mandela City on Beinecke Plaza at 5:30 am on
April 14. A lot of concerned students built it in early April.  Not only
did Yale have officials destroy it, but the police arrested 76 students.
After the local community's huge outcry, the university allowed the
students to put the Shantytown up there again.
```

On the other hand, PAULINE generates quite a different description from the same underlying knowledge when PAULINE takes the role of a Yale official who is (notionally) talking in person (and hence informally) about the events. Both texts are forceful, but note the differences in lexical choice (e.g., vulgarity), informality, and biased interpretation of the events (Hovy, 1987, p. 9):

```
It pisses me off that a few shiftless students were out to make trouble
on Beinecke Plaza one day; they built a shantytown, Winnie Mandela City,
because they wanted Yale University to pull their money out of companies
with business in South Africa.  I am happy that officials removed the
shantytown one morning.  Finally, Yale gave in and let the shitheads put
it up again, and Yale said that a commission would go to South Africa to
check out the system of apartheid.[9]
```

PAULINE works with a set of rhetorical goals which act as intermediaries between the pragmatic parameters of the system (the interpersonal goals and conversational setting) and the grammatical decisions (made by syntactic experts operating on a phrasal lexicon). These rhetorical goals include formality,

[9]As this is notional rather than actual speech, it still has more well-formedness than might occur in practice.

```
((deny (classification OBJECT POSITED))
 (state (classification OBJECT REAL))
 (concede (share-attributes OBJECT POSITED ATTRIBUTES1))
 (override (share-attributes -- POSITED ATTRIBUTES2))
 (override (share-attributes OBJECT REAL ATTRIBUTES3)))
```

Figure 2.34 McCoy's DENY-CORRECT-SUPPORT Strategy (McCoy, 1985b, p. 40)

simplicity, timidity, partiality, detail, haste, force etc. Formality, for example, can have the values highfalutin, normal or colloquial.

Hovy argues for this distinct level of stylistic representation since pragmatic effect is seldom the result of a single rhetorical goal but often rather a complex interaction of many (see discussion Hovy, 1987, pp. 36-38). Rhetorical goals offer a practical (but partial) attempt at the problem-laden field of pragmatics. In a related vein, Haimowitz (1989) seeks to combine knowledge about the user with the nature of the information to be output from an expert system to tailor presentation so as, for example, to give a nervous elderly person a reassuring view of a recommended hospital stay. This work indicates exciting uncharted territory for further exploration.

### 2.8.4 Recovering from Miscommunication

Oftentimes addressers are so insensitive that they not only do not couch language in terms which are optimal to the addressee, but they leave out critical details, are imprecise, over-specific, or ambiguous. This confuses or misleads the uninformed addressee. In other cases, perhaps due to inattention or inexperience, addressees are to blame for miscommunication.

In ROMPER (Responding to Object-related Misconception) McCoy (1985ab) examined two particular types of miscommunications that a person (a hypothetical consultation system user) can make: misclassification and misattribution. For example, if someone identifies a whale as being a fish it is a misclassification since whales are mammals. McCoy identifies this as a "like-super" misclassification because the object (whale) shares properties of the superordinate (fish). Both are fin-bearing and live in the water but whales are mammals because they breathe through lungs and breast-feed their young.

McCoy suggests a general strategy for recovering from these miscommunications. By analyzing transcripts of a radio program that provided callers with financial advice, McCoy noticed that misconception correction follows the pattern DENY-CORRECT-SUPPORT. Figure 2.34 shows the result when the DENY-CORRECT-SUPPORT pattern is expanded into the generic strategy for responding to "like-super" misclassifications.

For example, to correct a misclassification ("Whales are fish"), the expert would first deny the incorrect statement ("Whales are not fish"), then state the correct fact ("Whales are mammals"), then perhaps concede some similarity ("It is true that whales live in water and have fins"), and finally override this conceded information ("However, they breathe through lungs and feed their young with milk").

Importantly, McCoy's representation of text structure encodes not only the type of rhetorical proposition under which information is subsumed (e.g., "classification", "share-attributes") but also the communicative role this content plays in the discourse (e.g., "deny", "state", "concede", or "override"). However, communicative role does not necessarily impose any order on the associated propositions so there is the problem of guaranteeing well-structuredness in the output text as a whole.

In addition to object confusions (i.e., misclassification and misattribution), the user can be confused about domain plans. Joshi, Webber, and Weischedel (1984) suggest several strategies to correct user misconceptions about plans. For example, if their system believes that a user's goal cannot be achieved by the user's current plan, then it advises the user of alternative plans or else says that the goal is unattainable. Similarly, Pollack (1986) discusses a system that detects misconceptions underlying user's plans for action in the domain of electronic mail, and Quilici's (Quilici et al., 1988; Quilici, 1989) UNIX advisor "debugs" ill-formed user plans that are the result of mistaken user beliefs by accessing a number of "Justification Patterns" (e.g., plan causes unachievable goal, plan indirectly thwarts goal, plan indirectly precludes effect).

In addition to user misconceptions about domain entities and plans, there are a number of other causes of miscommunication including unsignaled focus or context shifts, over- or under-specifications, and the use of poor analogies. Cohen (1981) examined proper reference identification and its role in a plan-based theory of communication. Goodman (1986) investigates a number of forms of miscommunication including referent confusion, action confusion, goal misunderstanding, and cognitive overload. Litman and Allen (1987) present a series of plans which can be used to model topic change, clarification, and correction subdialogues.

Instead of investigating specific strategies that allow a system to recover from particular types of object and plan miscommunication, (Moore and Paris, 1989; Moore and Swartout, 1989ab, 1991; Moore, 1989) suggest a set of "recovery heuristics" to react to explanation failure (i.e., if the user rejects a system generated explanation as unsatisfactory). In particular, they address clarification subdialogues (Allen and Litman, 1987). Thus as with McCoy above, the system is generating within a dynamic interaction, as opposed to the simpler situation of, for example, TEXT. For instance, to recover from a failed strategy Moore and Swartout (1989, p. 29-30) suggest:

- If another plan exists for achieving a discourse goal, try it.
- If the discourse goal is to describe a concept, provide examples.
- If the discourse goal is to describe a concept and there are similar objects, provide an analogy.

The important point here is that instead of treating explanation as a one-shot game -- computing the single, ideal response -- the system merely keeps on slugging away with different strategies until the user is satisfied.

This reactive approach brings up another issue. Recall that blame for miscommunication can lie with either the speaker or the hearer. Up to this point we have only considered hearer misconceptions. In contrast, Moore and Swartout's reactive planner keeps track of assumptions it made during the planning of a response. These can become candidates for causes of the miscommunication in the speaker rather than the addressee. Therefore, it uses another heuristic:

- If any assumptions were made in planning the last explanation, plan responses to make these assumptions true.

The reactive approach is provocative since it underscores that communication is an ongoing affair, with give and take between addresser and addressee. In a similar manner, TEXPLAN can automatically replan responses to satisfy user goals; however in contrast TEXPLAN keeps both a discourse model of the interaction with the user as well as an explicit model of what effects the system believes it has achieved on the addressee. Other contrasts are indicated in the discussions of text planners in the next chapter.

### 2.8.5 Tailoring to a Point of View

One way to avoid miscommunication in the first place is to be more precise in producing output. This can be accomplished by "placing yourself in the listener's shoes," that is seeing things from their perspective. In the context of Advisor, a system which recommends courses to students, McKeown et al. (1985) produce varied output depending upon what point of view the system believes the user is taking with respect to the problem. By crosslinking hierarchies of entities and processes in the underlying knowledge base, the system selects a particular view of the knowledge which is relevant to the current request and user goal. For example, student course selection can be seen from a variety of different perspectives. It can be viewed as (McKeown et al., 1985, p. 794):

- a process of meeting requirements
- a state model process (i.e., which courses should be completed when)
- a semester scheduling process
- a process of maximizing personal interests

Hence, if the student asks the system "Should I take both discrete math and data structures this semester?" a variety of responses can be generated depending upon the perspective selected as appropriate for the user. Consider:

1. Requirements: `Yes, data structures is a requirement for all later Computer Science courses and discrete math is a corequisite for data structures.`

2. State Model: `Yes, you usually take them both first semester, sophomore year.`

3. Semester Scheduling: `Yes, they're offered next semester, but not in the spring and you need to get them out of the way as soon as possible.`

4. Personal Interests (e.g., AI): `Yes, if you take data structures this semester, you can take Introduction to AI next semester, and you must take discrete math at the same time as data structures.`

McCoy (1985ab) implemented an analogous notion of perspective in a system that recovers from miscommunications. In her system ROMPER (discussed previously), McCoy dynamically highlights attributes in the knowledge base as the dialogue progresses depending upon shifts in the focus of attention. She exploits this "attribute filtering mechanism" to select an appropriate strategy to recover from user misclassifications or misattributions. For example, if the user misstates that "whales are fish" (when in fact they are mammals), ROMPER is able to tailor its response based on the currently active perspective. If a perspective on 'body-characteristics' of the objects is active, the response generator will first test the similarity of whales and fish. The similarity of two objects is computed by summing the salience values of the attributes that the two objects have in common and subtracting this from the salience values of the attributes not common to both objects (Tversky, 1977). Since body characteristics on whales and fish are similar (as measured by Tversky's metric), McCoy uses her 'like-super' strategy (see above) to generate the following corrective response:

```
USER:   Whales are fish.
SYSTEM: No, they are mammals.  You may have thought they were fish
because they are fin-bearing and live in the water.  However, they are
mammals since (while fish have gills), whales breathe through lungs and
feed their young with milk.
```

On the other hand, say a different perspective is active: the "common-people's-perception," which highlights such attributes as "large-aquatic-creatures" and "scare-people." From this point of view, ROMPER would find whales and fish dissimilar and the above "like-super" strategy would not be applicable. So ROMPER then looks for fish descendents that are similar to whales in the current perspective, finds one (sharks), and then applies the "like-some-super" schema to produce the response:

```
USER:   Whales are fish.
SYSTEM: No, whales are not fish, they are mammals.  You may have
thought that they were fish since they are like the fish, sharks, in
that both are large aquatic creatures and both scare people.  However,
whales are mammals since (while fish have gills), whales breathe through
lungs and feed their young with milk.
```

While most application knowledge representations allow different categorizations of objects, most do not incorporate mechanisms for changing selection of categorizations. However, Suthers (1988a, 1988b) is investigating a "View Retriever" that selects information from the knowledge base given a topic, rhetorical relation, model (clinical, physical, etc.), and a depth boundary. An equally difficult issue is the complex reasoning that must be undertaken to infer the user's perspective from dialogue. As Advisor and ROMPER illustrate, however, determining and using the user's perspective on the task at hand can allow a generator to produce text that is much more meaningful and helpful.

### 2.8.6 Tailoring: Advantages and Limitations

The work discussed in this section illustrates investigations into a variety of techniques which tailor content, lexemes, syntax, perspective, rhetorical form, and pragmatic force. Together with the text realization and text planning techniques discussed in previous sections, these mechanisms enable generators to produce language that is more natural, individuated, and effective. But while mechanisms for tailoring explanations to a particular user may indeed enhance performance on a particular task (e.g., tutoring, consultation), we should be careful at how quickly we drop general strategies for personally tailored ones because of weakness in both the quality and the quantity of the information from which we can model the user (cf. Sparck Jones, 1989).

## 2.9 Summary and Future Directions

### 2.9.1 Summary

We have examined the origins and the current directions in the generation of explanations. Canned text, while as fluent as the composer, is adequate only for the most basic of applications. While code conversion can cope with changes in the underlying formal representation, longer texts introduce significant coherency problems. Recent research efforts have resulted in linguistically-motivated models from which we can build generation systems for longer and more sophisticated texts. Some researchers have contributed more general techniques for lexical selection and phrasal choice. Others have identified domain and text-independent discourse strategies that characterize lengthier stretches of text. Finally, techniques have been developed not only to tailor text to the user but also to recover from miscommunications.

### 2.9.2 Future

There are many unsolved issues in the field of explanation generation. What support knowledge, representation schemes, retrieval mechanisms, and tailoring techniques are required to capture the breadth and depth of human explanative capabilities? From an engineering perspective, what is the relationship between the text planner and the realization component (the "res" and the "verba") with respect to shared knowledge, control, and interaction (e.g., should the components be sequential, parallel, or interleaved)? (Paris et al., 1988; Hovy et al., 1988)

The remainder of this dissertation addresses one of the major issues in text generation: communication as a plan-based activity. Planning when, what, and how to utter is guided by the interaction between the addresser and the addressee, their respective knowledge, beliefs, and desires, all in the context of the current discourse. Previous work has focused primarily on planning speech acts to satisfy communicative goals (e.g., Appelt, 1985). Researchers have only recently (e.g., Hovy, 1988a; Moore, 1989) begun to investigate planners that reason about pragmatic and rhetorical strategies to produce multisentential text embodying sequences of speech acts. TEXPLAN extends past work beyond simple descriptive or comparative texts to characterize expository, narrative, and persuasive texts in terms of communicative acts. Utilizing dynamic communicative strategies including a notion of rhetorical acts, traditional illocutionary speech acts, and surface speech acts (Appelt, 1985), TEXPLAN is able to plan texts that have the potential to change not only the beliefs and knowledge of the addressee, but also the addressee's ability and desire to perform actions. The next chapter, therefore, examines plan-based models of communication, an approach to provide a foundation for the explanation production system detailed in the remainder of the dissertation.

# Chapter 3

# EXPLANATION PLANNING

Without knowing the *force* of words it is impossible to know men.
Confucius, Bk XX, 3

## 3.1 Introduction

This chapter considers the notion of language as planned action and purposeful behavior. The chapter begins by examining a fundamental flaw in schema-based generation systems and the practical need and philosophical basis for a planned-based approach to communication. This leads to a consideration of computational systems that plan natural language utterances. This discussion progresses from initial systems that planned physical actions, through those that planned single-utterance speech acts, to more recent efforts to plan multisentential text. The strengths and weaknesses of these systems and their plans are indicated. The chapter concludes by indicating the need for an explicit text plan. This sets the background for the computational theory of planned communicative acts which is introduced in the following chapter. In order to contrast the explanation planning approach described in this dissertation with other work, however, we first summarize the salient characteristics of TEXPLAN, which is more fully described in subsequent chapters.

Motivated by an analysis of human-produced explanations, this dissertation claims that multisentential text can be characterized by three integrated levels of communicative acts: rhetorical acts (e.g., describe, define, narrate), illocutionary acts (e.g., inform, request), and surface speech acts (e.g., command, recommend). TEXPLAN, a computer system that both plans and realizes multisentential English text, distinguishes between rhetorical acts (e.g., describe, define), which identify the communicative function of text, and rhetorical predicates (e.g., logical-definition, attribution), which identify types of utterances based on their content. Rhetorical predicates which not only identify content but also relate different pieces of text are termed rhetorical relations (e.g., evidence, motivation). Communicative acts, just like physical acts, have associated effects and are formalized as over sixty compositional plan operators in TEXPLAN.

Following traditional planning formalisms, TEXPLAN represents a communicative action in the `header` of a plan operator, the goal in the `effect`, and the subgoals of the action in the `decomposition`. The operators also represent the `constraints` on the action occurring as well as its `preconditions` (enablements), the distinction being the latter can be achieved or planned. These distinctions are important because constraints and preconditions help guide plan operator selection. Also, by recording the effects of its utterances distinct from its plan for communication, TEXPLAN constructs a model of the expected effects on the user. Taken as a whole, the plan operators characterize four text types including (1) entity description (2) event narration (3) plan, process and proposition exposition, and (4) argument, each of which are intended to have unique effects on the user's knowledge, beliefs, and desires. Finally, TEXPLAN exploits three distinct but related notions of focus -- discourse, temporal, and spatial -- to guide the order and realization of text. Having briefly characterized my approach to explanation planning, we now consider planned communication in general.

## 3.2 The Need for Planned Communication

As described in the last chapter, McKeown's (1982, 1985a,b) text schemas provided a method of selecting and ordering content (text planning in Figure 2.0) by encoding prototypical discourse organizations that were guided by local focus constraints. The principal weakness of this text schema paradigm is its excessive simplicity. Text schemas enumerate standard patterns found in human produced texts. Unfortunately, schemas do not characterize *why* these patterns exist -- their motivation -- nor do they take account of their intended effect on the hearer. For this reason they can be viewed as compiled plans: the result of a decision-making process involving inference about speaker and hearer models[1], presentation strategies, and domain knowledge. Not surprisingly, one of the characteristics of text schemas is their efficiency. While standard rhetorical patterns are common in discourse (e.g., a formal ceremony), in many situations speakers must reason about their audience, subject, and context in order to tailor text or to deal effectively with unexpected situations. Plans capture the basis for these decisions.

From the start, researchers have attempted to integrate a theory of linguistics with action-based planning. Central to this effort was the philosophical foundation of communication as a goal-oriented endeavor (Austin, 1962). Austin claimed that all utterances perform actions by *locution*, *illocution*, and *perlocution*.[2] The first kind of act, locution, is simply that of uttering words using the phonology, syntax, and semantics of a language. Illocution, in contrast, is the communicative act conveyed by a locution, such as stating, requesting, warning, ordering, or apologizing. Each of these illocutionary acts presents some

[1]The term speaker and hearer are used because this is conventional in speech act theory although this is not meant to imply anything other than written communication. Furthermore, in the context of natural language generation system, the speaker typically refers to the generator and the hearer to the user.

[2]In *How to do Things With Words*, Austin first distinguishes between constatives (which can be true or false) and performatives (that can be felicitous (successful) or infelicitous). Later, he classifies all utterances as performatives.

propositional content with an illocutionary force that characterizes the nature of the act. The third form of a linguistic act is perlocution, the act defined by the effect an utterance has. For instance while an illocutionary act may inform an audience of a proposition, the corresponding perlocutionary act may be to *convince* them of its truth (but it might equally *insult* or *frighten* them). Perlocutionary acts in turn produce *perlocutionary effects*. Thus convincing produces belief, frightening produces fear.

Searle (1969, 1975) formalized the necessary and sufficient conditions for the successful performance of illocutionary acts (often simply called speech acts, and so referred to in these terms), both direct and indirect (i.e., explicit and implicit). Searle classified all illocutionary speech acts as one of the following:

| | |
|---|---|
| *assertives*: | Commit the speaker to truth of expressed proposition. |
| *directives*: | Attempt to get the hearer to do something (e.g., questions and commands). |
| *commissives*: | Commit the speaker to future action. |
| *expressives*: | Express a psychological state (e.g., apologize, praise) |
| *declarations*: | Correspondence between propositional content and reality (e.g., pronouncing a couple to be married) |

Bruce (1975) was the first to suggest a plan-based model of speech acts. Several researchers then focused on representing speech acts in plans with preconditions (Searle termed these preparatory conditions), effects (termed essential conditions by Searle), and bodies. These plans could apply both to interpretation and generation. In fact some of the first computational implementations investigated both speech act interpretation (Allen, 1979) and production (Cohen, 1978). Others examined the identification of referents (Cohen, 1981), indirect and surface speech acts (Allen and Perrault, 1980; Hinkelman and Allen, 1987), and planning referring expressions (Appelt, 1985) as action-based communicative endeavors. Grosz and Sidner (1986) investigated the relationship between intentional structure and the discourse segmentation (represented computationally as a stack): manipulating goals is an essential feature of the whole view of speech acts based on communicative intentions. In a related vein, Litman and Allen (1987) examine plan recognition of topic change, clarification, and correction subdialogue in task oriented conversations.

The remainder of this chapter considers the evolution of plan based approaches to communication that select, organize, and realize natural language utterances, thus applying planning to the gamut the text planning and linguistic realization levels of Figure 2.0. It is important throughout this discussion to distinguish between systems that plan both physical and linguistic actions (typically using the same planner, as in Power (1974), Meehan (1976) and Appelt (1982)) from systems that are plan-based communicative components (Hovy, 1988a; Moore, 1989) which are attached to non-planning systems. Unlike previous systems, TEXPLAN, a plan based communicative component, has been applied to both planning application systems (e.g., KRS, a mission planner (Dawson et al., 1987)) and non-planning systems (e.g., LACE, a knowledge based simulation (Anken, 1989)).

## 3.3 Power's Mary and John

One of the first implementations to attempt to integrate communication with action was Power's (1974, 1979) STRIPS-like planning system in which two robots, Mary and John, could collaborate to achieve a goal (e.g., for John to get out of a room). The robots could pass each other questions or assertions to get information necessary to achieve the goal. Just as the STRIPS (Fikes and Nilsson, 1971) planner represents a precondition, body, and effect of an action, a typical act in Power's system was:

```
ACTION:       (robot push)
SITUATION:    (bolt up)
RESULT:       (door move)
```

This states that in order to get the door to move, the robot would have to push, provided that the bolt was up (i.e., the door was unlocked). Three actions were represented in the system: MOVE, PUSH, and SLIDE. Power's formalism was limited to unary acts and subacts and it did not represent multiple effects or more complex preconditions or situations (e.g., disjunction). Moreover, Power's plans and planning mechanism were extremely simple: all plans are constant, except for "robot" which was a variable instantiated to either Mary or John.

In order achieve their non-linguistic goals, Mary or John might have to seek or provide information. They executed their conversations through "games": stereotypical methods of asking, telling, and so on. For example, if John needed information he executed the ASK game (essentially a speech act) and initiated the conversation by uttering "May I ask you a question?" Unfortunately, as Power himself recognized, the relationship between the speaker's goals and the games is implicit and fixed in his program: there is simply a function call. That is, his system cannot infer which game to select and why, given dialogue context. Yet, as Power recognized, people know to how choose appropriate linguistic actions (e.g., request, warn, or inform) to achieve their goals. This was exactly the question Cohen would later address in his system OSCAR (1978).

## 3.4 Meehan's TALE-SPIN

In contrast to Power's focus on conversations, Meehan (1976, 1977) designed a system that constructed simple stories about agents who made plans to achieve their goals. The system, TALE-SPIN, would describe the frustrations of the agents when situations and events impeded the execution and/or success of their plans. While TALE-SPIN did not actually produce language, agents in the story could plan linguistic actions such as TELL, ASK, and PERSUADE in order to achieve their extra-linguistic goals such as getting to a location or controlling some object. The STRIPS-like plans encoded preconditions, postconditions, and postactions. They therefore suffered from the same drawbacks as Power's plans. Interestingly, however, some plans could be achieved by a number of alternative subgoals. For example, actors could PERSUADE by simply requesting, proposing a good reason, bargaining, or threatening. Chapter 5 discusses TALE-SPIN's relationship to narrative generation.

## 3.5 Cohen's OSCAR

Cohen (1978) focused on planning illocutionary speech acts (cf. Cohen and Perrault, 1979).[3] While Cohen's system, OSCAR, did not generate English output, it did select an appropriate speech act, determine which agents were involved in the speech act, and choose the propositional content of the utterance. Like Power's research, conversations in OSCAR concerned a robot world, except that the door had to opened by a key in OSCAR. Unlike Power's model, linguistic as well as non-linguistic acts are described in a STRIPS-like formalism that represents an ACTION that will achieve some EFFECT provided certain PRECONDITIONS hold. As in STRIPS (Fikes and Nilsson, 1971), all aspects of the world are assumed to stay constant except as described by the operator's effects and logical entailments of those effects (TEXPLAN's planner also makes this assumption although it is hierarchical, i.e., plan operators have decompositions). The linguistic acts include INFORM, REQUEST, CONVINCE, and CAUSE-TO-WANT. For example, the plan to have AGENT1 inform AGENT2 about some proposition, P, is:

```
ACTION:          INFORM(AGENT1, AGENT2, P)
EFFECT:          AGENT2 believes AGENT1 believes P
PRECONDITION:    AGENT1 believes P and
                 AGENT1 wants to inform AGENT2 that P
```

That is, if you INFORM someone of something (and you believe it and want to inform them of it), then they will believe that you believe it. Notice, however, that they themselves might not believe it. In order to achieve this, the speaker must CONVINCE the hearer of it. The plan for this is:

```
ACTION:          CONVINCE(AGENT1, AGENT2, P)
EFFECT:          AGENT2 believes P
PRECONDITION:    AGENT2 believes AGENT1 believes P and
                 AGENT1 wants to convince AGENT2 that P
```

Unfortunately, planning in OSCAR stops here. There is no description of what AGENT1 does to convince AGENT2, as in TALE-SPIN's representation of persuasion. So while these acts formalize speaker and hearer preconditions and the effects of individual actions, they do not tell how to perform the act (i.e., its decomposition). In contrast, TEXPLAN indicates explicit rhetorical actions found in naturally occurring text that achieve communicative goals (e.g., to CONVINCE the hearer of the truth-value of a proposition, provide evidence).

Like the INFORM act above, the plan operator for a REQUEST is defined as:

[3] Similar speech act operators were used by Allen (1979) to recognize speech acts (c.f., Allen and Perrault, 1980).

```
ACTION:          REQUEST(AGENT1, AGENT2, ACT)
EFFECT:          AGENT2 believes AGENT1 want ACT
PRECONDITION:    AGENT1 believes AGENT2 can do ACT and
                 AGENT1 wants to request AGENT2 to ACT
```

Thus if you REQUEST someone to perform an action (and you believe they can do it and you want to request them to do it) then as a result they will believe you want the action. To actually get the hearer to want to do the action, a CAUSE-TO-WANT act is planned after the REQUEST act, just as a CONVINCE act is planned after the INFORM act to cause the hearer to believe some proposition. As with the CONVINCE act above, however, the CAUSE-TO-WANT act does not indicate what the speaker does to achieve this effect (i.e., there is no decomposition of the plan).

Cohen recognized that his plans require further elaboration and referred to his algorithm as "the lowest common denominator" speech act planner. In general, a weakness of most speech act research is its limitation to single utterances in limited context (e.g., no dialogue history). Single speech act planning fails to capture cooperative interactive dialogue strategies, exemplified in part by Power's system. Cohen also did not address issues like achieving communicative goals by direct or by indirect speech acts.

Cohen and Perrault (1979) later formulated INFORMREF and CONVINCEREF operators based on Allen (1979)[4]. These operators work with the REQUEST operator to plan wh-questions (e.g., "Where is Mary?") where the speaker does not know the referent (e.g., Mary's location) but believes the hearer does. Similarly, Cohen and Perrault provide INFORMIF and CONVINCEIF operators. These also work with the REQUEST operator but they characterize yes/no questions (e.g., "Is Mary in the room?"), where the speaker does not know the truth-value of a proposition but believes the hearer does.

Cohen (1981) later argued that speakers often plan to get the hearer to identify referents. He provides evidence for an IDENTIFY act. For example, phraseology of speaker requests such as "Find X" or "Notice Y" may encourage a physical search which can result in a hearer response "Got it" or "uh huh." Just as hearers must identify referents, speakers must plan them. The next section examines Appelt's Knowledge And Modalities Planner which was able to plan referring expressions by reasoning about a number of sources of knowledge.

## 3.6 Appelt's KAMP

Appelt (1982, 1985) extended Cohen's suggestions by planning not only speech acts but also syntactic structure and lexical choice. Like Cohen, Appelt viewed speech acts as communicative actions which could be modeled by the same planning process that handled extra-linguistic actions. In general, he saw goal satisfaction as a complex interaction between physical and linguistic actions, ultimately motivated by the speaker's desires. Within the framework of task oriented dialogues and assuming knowledge of the

[4]See footnote 18 in Cohen and Perrault (1979), p. 488. Names for these acts were suggested by W. Woods.

state of the world and mutual and individual beliefs, Appelt's system, KAMP (Knowledge And Modalities Planner), planned utterances such as "Tighten the screw with the long Phillips screwdriver" in order to accomplish some desired goal state (e.g., get a pump attached to a platform). KAMP planned appropriate referring expressions (e.g., "the long Phillips screwdriver") by reasoning about domain objects, the setting, and the knowledge of the hearer (cf. Chapter 2, Section 2.5.1 on lexical choice). Unlike TEXPLAN, KAMP focused on the production of isolated utterances.

Figure 3.1 shows KAMP's hierarchy of linguistic actions which included illocutionary acts, surface speech acts, conceptual activation (i.e., description selection), and utterance acts. The illocutionary and surface speech acts can be viewed as operators on the propositional content which is then specialized by the selection of particular forms of concept description before the whole is realized by the final utterance act. In particular, KAMP plans INFORM and REQUEST acts which are realized using three "surface speech acts": COMMAND (for imperative sentences), ASK (interrogative sentences), and ASSERT (declarative sentences). KAMP's hierarchical planning mechanism was based on procedural nets (Sacerdoti, 1977) which allow knowledge from many different sources to interact to solve a problem.

Appelt's planner was based on a possible-worlds-semantics (Moore, 1980) where goals are stated with respect to a potentially infinite sets of possible worlds. In order to constrain the search of this infinite space during planning, Appelt's system summarized actions heuristically before verifying their validity within the possible-worlds formalism. This was a significant limitation because not only was the full-power of the formalism unavailable during planning, but if the selected plan failed during execution, a replanner had no access to the rich, possible world semantics. This made recovery from a failed communicative plan difficult if not impossible. Moreover, despite its restricted domain (two agents and a few objects), KAMP was very slow because of its use of the possible-worlds formalism (approximately 20 minutes per utterance) although subsequent work achieved speeds of 1-2 minutes per utterance, including planning, reasoning, and linguistic generation tasks (Appelt, personal communication). To express the linguistic knowledge of the system in a more modular and less ad-hoc way than in the initial system, a functional unification grammar, TELEGRAM, was later integrated. The hierarchical planning mechanism used in TEXPLAN is less sophisticated than the possible world formalism used by KAMP. However, because goals in TEXPLAN are not stated with respect to a potentially infinite sets of possible worlds, computational complexity is reduced, reflected in a more efficient implementation (individual utterances take only a few seconds to plan and linguistically realize).

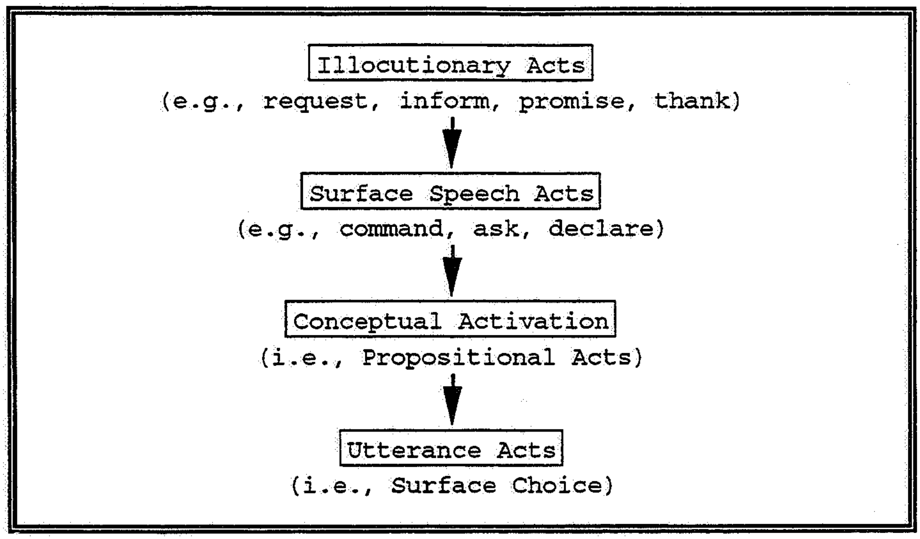



Figure 3.1 Appelt's Planning Hierarchy, KAMP (from Appelt, 1985, p 9).

In summary, Appelt's abstraction of planning levels into illocutionary acts, surface speech acts, conceptual activation, and utterance acts was novel. However, only a few illocutionary acts and surface forms were investigated and they were tested in a limited domain using a very powerful and computationally complex planning mechanism.

## 3.7 Control: Planning and Realization

As mentioned earlier, an important consideration in a natural language generation system is the nature of control between the text planner and the linguistic realization component (Hovy et al., 1988). The interaction runs the gamut from a pipeline approach, where the content and form of the message are wholly pre-planned before the realization component is invoked (as implied by Figure 2.0) to an interleaved one where the planning and realization components are able to interact. As planning is an intrinsically active process it brings up this general issue about the generation strategy in a particularly sharp way. Since Appelt's system represented multiple alternatives at multiple levels of abstraction, from illocutionary acts to surface expressions, the planner was able to retract previous choices when active subplans failed. This was a computational manifestation of Appelt's disbelief in the *conduit metaphor* which describes language as a pipeline relaying information from the speaker (writer) to the hearer (reader). Appelt's system worked, however, because he allowed only a limited pragmatics scope for the hearer's knowledge and state. Hence, the constrained search space made backup computationally feasible. It is unclear if this approach would scale up to a domain which included many communicative and physical actions, domain objects, agents and their beliefs.

In contrast, in an early version of MUMBLE McDonald employed limited-commitment planning, allowing for two-way communication between his planner and realizer. Hovy (1987, 1988b) later suggested a confluence of prescriptive or top-down constraints, and restrictive or bottom-up constraints. TEXPLAN recognizes the need for flexible planning, and allows lack of information or failed subgoals (e.g., failed preconditions, failed subplans, or negative user feedback) to signal to the text planner to select another approach to the current linguistic or extra-linguistic goal. But future generators must interleave content selection, structuring, and phrasing in a yet more flexible manner.

## 3.8 Rhetorical Text Structure

With the push toward planning multiple utterances, there emerged a need for formal representations of the components and relations underlying text. Rhetorical Structure Theory (RST) (Mann and Thompson, 1987), influenced by previous work (Grimes, 1975; Weiner, 1980; McKeown, 1985), resulted from analysis of a wide variety of texts, identifying 23 rhetorical relations that exist between parts of texts. For example, the PURPOSE relation indicates that an utterance such as "in order to avoid cavities" provides rationale for the utterance "Brush your teeth." These rhetorical relations indicate the function or role an utterance plays in relation to the other elements of a text. Relations include EVIDENCE, ILLUSTRATION, ELABORATION, BACKGROUND, etc. Associated with each relation are key (though not obligatory) connective phrases. For example, the PURPOSE relation typically is signalled by the phrases "in order to," "so that," etc. Text can be characterized at all levels by these relations (e.g., ELABORATION can apply to utterances, paragraphs, or sections). This is analogous to Grimes' (1975) notion of recursive rhetorical predicates. In fact, four of the ten sentence-level rhetorical predicates in McKeown's (1982, 1985) TEXT system (e.g., identification) had paragraph-level correlates although the recursive issue was only partially investigated. Suthers (1989, p. 54) compares the sets of rhetorical relations of Grimes (1975), McKeown (1985), and Mann and Thompson (1987).

While RST endorses some of the findings of text linguistics (e.g., Grimes, 1975), it formalizes the effects that individual rhetorical relations can have on the hearer, and so emphasizes the communicative function of discourse. For example, providing EVIDENCE can increase the hearer's belief in some claim. Similarly, the relations ENABLEMENT and MOTIVATION provoke the reader to action, the first by increasing their ability and the second by increasing their desire as in:

| TEXT | | |
|---|---|---|
| `Come to my party.` | REQUEST | (speech act) |
| `It's at 4095 Silvan Ave.` | ENABLEMENT | (rhetorical relation) |
| `You're guaranteed to have a great time.` | MOTIVATION | (rhetorical relation) |

RST's notion of "nuclearity" is illustrated by the above text in which the nuclear request "Come to my party." is supported by the "satellite" propositions which enable and motivate the reader to perform the requested action. The nuclear and satellite relationships between the constituents in the text are indicated graphically in Figure 3.2. In addition to encoding the effect of particular relations, RST indicates the constraints on the nucleus and satellites (e.g., the satellite of the EVIDENCE relation, which is some piece of evidence, should be believable by the reader). But while RST specifies interclausal relations, it does not characterize the illocutionary act associated with an utterance (as in the above request) and so it is therefore only a partial account of communicative behavior.

In general, there are no ordering constraints on relations. However, during text analysis RST researchers identified strong patterns of ordering which they term "canonical orders": strong tendencies rather than constraints. For example, they found that the relations ANTITHESIS, BACKGROUND, CONCESSIVE, CONDITIONAL, JUSTIFY, and SOLUTIONHOOD typically precede the nucleus. In contrast, ELABORATION, ENABLEMENT, EVIDENCE, PURPOSE, and RESTATEMENT are satellites that follow the nucleus.

Unfortunately, RST as presented in Mann and Thompson (1987) is purely descriptive and so fails to provide a computational account of text production (or interpretation). More importantly, while RST is concerned with the general rhetorical relations which hold between the parts of all genres of text, my work attempts to characterize the specific rhetorical relations underlying texts which explain behavior, justify conclusions, or provide advice, i.e., texts which have particular global functions. Though Mann and Thompson allow for levels of RST characterization they do not consider characteristic functional structures. RST does not provide any clear indication of whether there is any sort of discourse grammar imposing constraints on the way the relations characterizing greater spans of discourse compose to give those for longer ones or whether the way text is built up is freely determined by context. In fact, no one has produced texts longer than several sentences using RST and so this issue has not come to the forefront. Finally, since the basis for RST was a broad range of texts, from letters to bulletin board messages to newspaper articles, their relations may not capture the rhetorical nuances of specific genre such as instructions or arguments. This last point is supported by the fact that the implementation of some RST relations has revealed that the set is not complete and individual operators may require further refinement (Hovy, personal communication).

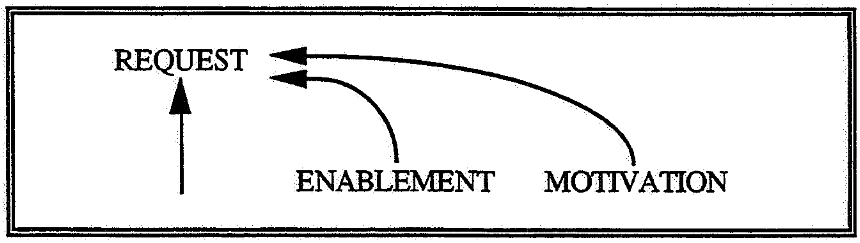



Figure 3.2 RST relations supporting a speech act

## 3.9 Hovy's "structurer"

Hovy (1988a, p. 168) was the first to encode some (currently about 20%) RST relations as plans. Rhetorical relations are represented in NOAH-like (Sacerdoti, 1977) plan operators that include preconditions, effects and subgoals as well as connective phrases. The plans are coded in terminology based on RST and represent the name of a plan, its results, its nucleus and potential satellites, as well as the constraints on the nucleus and satellites. In addition, the plans encode the order of the nucleus and satellite and "relational phrases" which act as connectives/cues in the text of the rhetorical relation. Figure 3.3 illustrates the SEQUENCE relation as a plan operator which presents actions in some sequential order (e.g., temporal). For clarity, I have dropped the mutual belief notation `(BMB SPEAKER HEARER proposition)` which would appear around every proposition in Figure 3.3.

One problem with Hovy's implementation of RST is illustrated by the SEQUENCE operator. The SEQUENCE relation used in the example consists of a number of subrelations in a fixed order: the `CIRCUMSTANCES-OF`, the `ATTRIBUTES-OF`, the `PURPOSE-OF`, or the `DETAILS-OF` some action. Furthermore, while the SEQUENCE relation tells when these subrelations can be added or instantiated, it does not specify what *goal* these relations are satisfying or why these relations are being chosen *when* they are. So the SEQUENCE relation is nothing more than a plan operator which states that its child relations are ordered, but leaves the rationale for that ordering implicit. Thus, for example, Hovy's SEQUENCE relation does not recognize a distinction between temporal, spatial, or other ordering. Planning is dynamic with respect to the choice and order of RST relations, but no order or choice applies within relations (despite the fact that much information is being added here). As a consequence, apart from the satellite/nucleus distinction, the result of planning with Hovy's system is a structure which is very similar to McKeown's schemas: it tells what to say when, but not why.

```
Name:                          SEQUENCE
Results:                       ((SEQUENCE-OF ?PART ?NEXT))

Nucleus
requirements/subgoals:         (AND (MAINTOPIC ?PART)
                                    (NEXT-ACTION ?PART ?NEXT))
growth points:                 ((CIRCUMSTANCE-OF ?PART ?CIR)
                                (ATTRIBUTE-OF ?PART ?VAL)
                                (PURPOSE-OF ?PART ?PURP))

Satellite
requirements/subgoals:         ((MAINTOPIC ?NEXT))
growth points:                 ((ATTRIBUTE-OF ?NEXT ?VAL)
                                (DETAILS-OF ?NEXT ?DETS)
                                (SEQUENCE-OF ?NEXT ?FOLL))

Order:  (NUCLEUS SATELLITE)
Relation-phrases:  ("" "then" "next")
Activation-question:  "Could ~A be represented as start-point, mid-
point, or end-point of some succession of items along some dimension?
-- that is, should the hearer know that ~A is part of a sequence?"
```

Figure 3.3 Illustration of Hovy's SEQUENCE relation

The SEQUENCE operator is used to "structure" a given set of propositions so that all the information in the input is included and the resulting text is coherent. Hovy's (1988c) system starts with a small database of 17 propositions concerning U. S. Naval vessels and events (e.g., (ENROUTE E105) (DESTINATION.R E105 SASEBO), etc.). These propositions are then enriched and grouped into 6 clause-sized chunks of related information using domain-specific rules.

For example, when a ship is observed to be ENROUTE and then to be stopped to LOAD, an ARRIVE event is constructed along with its time, agent (i.e., the ship KNOX), and temporal relations (i.e., the prior ENROUTE event and subsequent LOAD event). After constructing the 6 clausal groupings of propositions, the SEQUENCE relation of Figure 3.3, along with the CIRCUMSTANCE and ATTRIBUTE relations, is used to structure the groups into the text structure shown in Figure 3.4.

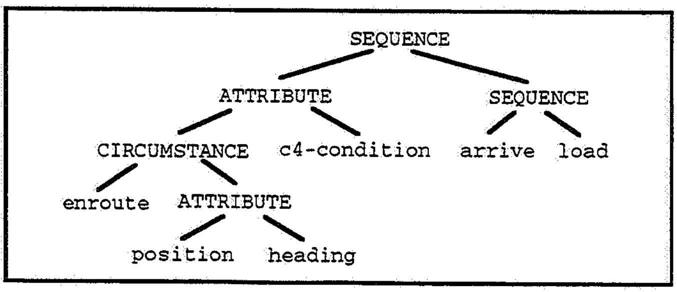


Figure 3.4 Hovy's text structure

The PENMAN systemic generator then realizes the structure in Figure 3.4 as (where C4 indicates the condition of the ship):

```
Knox, which is C4, is en route to Sasebo.
Knox, which is at 18N 79E, heads SSW.
It arrives on 4/24.
It loads for 4 days.
```

While the sentences in the above text are grammatical, they are simple, except for the embedded restrictive clauses in the first two sentences. Unfortunately, at the same time the text as a whole is unnatural. One reason is that the anaphoric references in the third and fourth utterances seem odd. A human speaker would be more likely to use referring expressions such as "the ship" or "the vessel." But this was not the focus of Hovy's work. He was, however, attempting to group information rhetorically, which in fact is also one of the problems with the above passage. In particular, the text first introduces the CIRCUMSTANCE (i.e., condition) of the Knox and then enumerates a SEQUENCE of events. While this may be structurally appropriate, it fails to bring together semantically connected propositions (e.g., grouping of direction information: "heads SSW" and "en route to Sasebo").

There is, moreover, no temporal ordering among the events (what is the relation of the events "en route," being at 18N 79E, arriving on 4/24 and loading?) The content could be conveyed much more naturally given a more sophisticated representation of verb tense and aspect (cf. Allen, 1988), although this necessitates a more sophisticated temporal/tense representation (Reichenbach, 1947) and event ontology (Moens and Steedman, 1988). To be fair, no current generator solves this difficult issue (although see Ehrich, 1987).

To overcome the shortcomings of a strictly RST-based approach, Hovy and McCoy (1989) investigated using Focus Trees (McCoy and Cheng, 1991) to guide the ordering and interrelationships of sentence topics. The nodes of Focus Trees are "topics" (objects, attributes, settings, actions, or events) that are introduced into the discourse by the participants. A node in the Focus Tree (a topic) is subordinate to another node if, during the conversation, it is introduced as a subtopic of the parent node. A legal shift from topic to topic (i.e., a tree traversal) is based on the type of the current entity. For example, if it is a *object*, then the conversation can shift focus to its attributes or an action in which it plays a role (i.e., create the corresponding subnodes). In contrast, shifts from an *action* node in the Focus Tree can be to an actor or object in the action, to the action's purpose, or to subactions. To illustrate how the use of a Focus Tree to constrain generation could improve text coherence, Hovy and McCoy (1989, p. 7) regenerated the above Knox text:

```
With readiness C4, Knox is en route to Sasebo.
It is at 79N 18E heading SSW.
It will arrive 4/24 and will load for four days.
```

The legal traversals of the Focus Tree control the selection and, hence, restructuring of propositional content which results in a more focused text.

In another representative example of his initial approach (Hovy, 1988a, p.166), the PURPOSE, SEQUENCE, and ELABORATION relations are used to plan the following text from the Program Enhancement Advisor (Neches et al., 1985), an expert system that recommends improvements to the efficiency, readability, and maintainability of Common LISP code. In the example below, the first clause elaborates how the program enhances code generation and the second clause provides the purpose for the activity:

| TEXT | RHETORICAL RELATION |
|---|---|
| In particular, the system scans the program | ELABORATION |
| in order to find opportunities to apply transformations to the program | PURPOSE |

Providing PURPOSE is clearly one aspect of explaining. Hovy's current implementation (Hovy, personal communication) includes six major relations (CIRCUMSTANCE, ELABORATION, PURPOSE, SEQUENCE, SOLUTIONHOOD and VOLITIONAL-RESULT). Of course explanations involve other rhetorical relations such as EVIDENCE and CAUSE, for example.

A great advantage of Hovy's approach is that individual relations, such as SEQUENCE, are formalized in a NOAH-like planning language which encodes the constraints on the nucleus and satellite, their respective growth points, and their preferred order. Unlike TEXPLAN's plan operators, however, some of Hovy's rhetorical operators include domain dependent information. For example, the CIRCUMSTANCE operator refers to underlying relations such as HEADING.R and TIME.R of naval events. And unlike Hovy's planner, which structures a given set of propositions, TEXPLAN performs content selection and text structuring concomitantly. Furthermore, as the SEQUENCE operator illustrates, Hovy's subactions in plans appear in a fixed order. In natural text, particularly in the case of SEQUENCE, ordering is much more sophisticated, based on, among other things, temporal, causal, spatial, attentional, and conventional constraints on order. A text planner must reason explicitly about these kinds of information to choose a particular output order.

A final problem is that though RST is in principle concerned with communicative functions, Hovy's text structuring procedures do not refer either to illocutionary or to perlocutionary acts -- indeed they are text structures not communicative plans to be executed. Yet communicative actions, perlocution and illocution in particular, are central to a plan-based approach to language. Actions like INFORM are to some extent implicit in his RST plans, but only in a weak and generalized way. The SEQUENCE relation, for example, simply has the effect of making the speaker and hearer mutually believe that there is a sequence between two propositions. There is no representation of informing, requesting, or persuading the hearer, nor of the perlocutionary effect such illocutionary acts have on the hearer (like having them know, believe, desire, or do something). The explicit formalization of these notions and their relationship to rhetorical relations enables the speaker not only to reason about the presentation of an explanation but also to execute the associated communicative acts. This is one of the key contributions of my work.

## 3.10 Moore's "Reactive Planner"

Moore's (Moore and Swartout, 1988ab, 1989; Moore and Paris, 1988, 1989; Moore, 1989) reactive planner (introduced in Chapter 2) is also based on RST, but it focuses specifically on clarification subdialogues (Litman and Allen, 1987) after producing a text that achieves a given goal (e.g., persuade the hearer to do some act). Unlike Hovy's text structurer, Moore's system constructs a text plan/text structure which includes rhetorical relations but also has individual speech acts as leaf nodes. Moore accounts for texts like that shown previously in Figure 3.2 by using the top-level plan `recommend-enable-motivate`, shown in Figure 3.5 in its LISP-like form (from Moore and Paris, 1989).

```
NAME:          recommend-enable-motivate
EFFECT:        (BMB S H (GOAL H Eventually(DONE H ?act)))
CONSTRAINTS:   nil
NUCLEUS:       (RECOMMEND S H ?act)
SATELLITES:    (((BMB S H (COMPETENT H (DONE H ?act))) *optional*)
                ((PERSUADE S H (GOAL H (GOAL H Eventually(DONE H ?act))) *optional*)
```

Figure 3.5 Moore's recommend-enable-motivate plan operator

This plan indicates that the effect (that the speaker and hearer mutually believe that the hearer has the goal of eventually accomplishing some act) can be achieved by recommending the action, (optionally) making sure that both the speaker and hearer believe the hearer is able to perform the action, and (optionally) persuading the hearer to do so.

Each operator in Moore's plan library encodes either a particular discourse goal/plan (e.g., `persuade-by-motivation`) or characterizes a rhetorical relation (e.g., `motivation`) supporting the goal or plan. The leaf nodes of a completed plan consist of illocutionary speech acts (inform or recommend) as in:

```
(recommend  SPEAKER HEARER
            (replace (actor user)
                     (object car-function)
                     (generalized-means first-function)))
```

which is sent off to the PENMAN sentence generator and realized as:

```
You should replace (car x) with (first x).
```

In contrast, TEXPLAN's leaf nodes indicate not only the speech act and its propositional content, but also the type of rhetorical predicate (e.g., attributive, logical-definition, evidence), thus abstractly marking the kind of propositional content contained in the proposition in order to guide higher level planning and also to guide subsequent linguistic realization (e.g., the use of cue words to signal the class information being conveyed or its connection to other propositions). A more significant distinction is that TEXPLAN explicitly distinguishes illocutionary speech acts such as `inform` and `request` which are penultimate leaf nodes in the text plan from surface speech acts (Appelt, 1985) such as `assert`, `command`, and `recommend` which are leaf nodes in the text plan.

```
NAME:          persuade-by-motivation
EFFECT:        (PERSUADE S H (GOAL H Eventually(DONE H ?act)))
CONSTRAINTS:   (AND (GOAL S ?domain-goal)
                    (STEP ?act ?domain-goal)
                    (BMB S H (GOAL H ?domain-goal)))
NUCLEUS:       (FORALL ?domain-goal
                    (MOTIVATION ?act ?domain-goal))
SATELLITES:    nil[5]
```

Figure 3.6 Moore's persuade-by-motivation plan operator

There also are some problems with Moore's plans that are a consequence of her system's close tie to RST. For example, consider the `persuade-by-motivation` plan operator shown in Figure 3.6. This plan operator characterizes an attempt to persuade the hearer to do an `?act` by telling the hearer that the `?act` achieves some mutual `?domain-goal`. The problem is that while these plans make the important distinction between nucleus and satellites (indicated in TEXPLAN as non-optional portions of the decomposition), they do not distinguish between constraints on the nucleus and constraints on the satellites (as does RST and Hovy's implementation thereof).

On a semantic level, Moore's plans do not consistently distinguish between action and effect. Actions are events executed by some intentional agent; effects refer to states of affairs that are the result of actions. But notice in the `recommend-enable-motivate` plan how the satellite includes both an effect (the speaker and hearer mutually believe the hearer is competent) and an action (persuade). Similarly, the *effect* of the `persuade-by-motivation` plan above <u>should</u> be the *state* that the hearer has the goal of eventually doing `?act`. Persuade is the *action* that achieves this effect (which incorrectly appears in the effect slot in the above plan). Furthermore, Moore's plans mix pragmatic acts and rhetorical relations (e.g., in the `recommend-enable-motivate` plan, `recommend` is a speech act but `enable` and `motivate` are RST-based rhetorical relations). The consequence is that actions (e.g., speech acts), rhetorical relations, and effects all appear in Moore's resulting text plan. In contrast, the decompositions of TEXPLAN's plan operators include only rhetorical acts or speech acts. Rhetorical relations appear as special types of rhetorical predicates on leaf nodes (e.g., evidence) of the text plan or are simply consequences of the planning process (over longer stretches of text). Thus, following traditional planning formalisms, TEXPLAN

[5]Moore's publications vary the value of empty satellites between "nil" and "none" and some figures omit the satellite altogether. These cases are assumed to be equivalent.

represents a communicative action in the `header` of a plan operator, the goal in the `effect`, and the subgoals of the action in the `decomposition` (which can be explicitly marked to distinguish between the nucleus and satellites). The plans also represent the `constraints` on the action occurring as well as its `preconditions` (enablements), the distinction being the latter can be achieved or planned. These distinctions are important because constraints and preconditions help guide plan selection. Also, by recording the effects of its utterances distinct from its plan for communication, TEXPLAN constructs a model of the expected effects on the user.

While Moore argues that Hovy's RST plans "look much like the schemas of McKeown's (1985) TEXT system" (i.e., they order rhetorical relations in standard patterns), at the same time she admits to a high-level schema at the `recommend-enable-motivate` level (Moore and Swartout, 1988b, p. 12). This is a fixed order of an illocutionary act supported by two rhetorical relations as shown in Figure 3.2. There are, of course, occasions where a significant amount of persuasion (achieved by providing evidence, motivation, purpose, or cause, for example) must precede rather than follow a recommendation. This is the case if the user has a strong bias against the idea or if an explication of the advisor's reasoning is useful or necessary (as in tutorial applications). In the following example, the speaker provides evidence *before* making a recommendation:

> The X-rays revealed a slight fracture of your femur. And the hematologic tests indicate low white blood cell count. Also, you may have internal bleeding near the lacerations and bruises. You really shouldn't play in the game on Saturday.

On the other hand, if the user model, task, or situation indicate haste, then an immediate request without motivation is likely:

> You should stop by tonight. (See you, I have to go now.)

or in the command

> Duck! (The ball is coming this way.)

On the other hand, the request can be implicit, if the hearer can readily infer the intended action, as in the advertisement

> You can get $500 cash back and a 5 year, 50,000 mile warranty.
> (Buy a Pontiac now.)

Focus also affects ordering as demonstrated by Hovy and McCoy (1988). Moore defines "global context" as the top level goal and entity, so the global focus (in the sense of (Grosz, 1977)) in Moore's example above includes the entities: user, car-function, and first-function. In contrast, we hold that global focus includes all entities closely related to these domain entities (e.g., associated subacts, effects, actors and so on) and so some local focusing mechanism is required. However the local selection of rhetorical propositions (rhetorical relations plus propositional content) is not guided by local focus constraints in Moore's system but rather is implicit in the binding of variables in plan operators during planning. Local focusing is necessary to achieve the Gricean maxim of relevancy, as McKeown (1985) demonstrated by her use of models of focus of attention to guide content selection and ensure text cohesion. Explicit models of local focus are necessary to characterize the interaction between attention and surface form (e.g., pronominalization, passive and active voice selection, and definite/indefinite noun phrase distinctions).

Finally, Moore's `recommend-enable-motivate` strategy is but one (preordered) pattern (with optional components) in a family of operators that can be used to get the hearer to do something. Moore's (1989) system has only one alternative recommendation plan in addition to `recommend-enable-motivate`. This is called the `recommend-by-simple-statement` shown in Figure 3.7.

```
NAME:      recommend-by-simple-statement
EFFECT:    (BMB S H (GOAL H Eventually(DONE H ?act)))
NUCLEUS:   (RECOMMEND S H ?act)
```

Figure 3.7 Moore's recommend-by-simple-statement plan operator

Despite these limitations, Moore not only offers the notion of a reactive explainer that reasons about the "effect of text", but her work also contributes a number of selection heuristics which can be used to choose among competing plan operators which achieve a given discourse goal. There are five such heuristics used by Moore's system including:

1. Avoid plans that make assumptions about the hearer's beliefs.
2. Prefer operators that refer to previously mentioned concepts.
3. Prefer operators that refer to concepts the hearer knows.
4. Prefer specific operators over more general ones.
5. Avoid operators that generate verbose responses.

A weighted sum of numeric measures of the above allows competing plans to be rank ordered.

In summary, while Moore's approach is provocative, humans (and the texts they produce) employ many other surface forms (e.g., commands versus recommendations versus suggestions) and a wide range of rhetorical strategies (e.g., invoking authority) to get the hearer to perform some action. Also, while Moore's plan operators improve upon Hovy's by explicitly representing effects and optionality, her operators do not generate lengthy text (e.g., multiparagraph) and are still fixed at the highest level.

## 3.11 Cawsey's Discourse Planner

Moore's reactive approach begins to explore the role of discourse in explanation. Cawsey's (1989, 1990) EDGE discourse planner goes beyond this and plans both content and discourse moves. As in

Hovy's and Moore's system just described, EDGE uses plan operators to formalize content planning (cplan) and discourse planning (dplan). For example, to plan the content of a device description, EDGE will use the `how-it-works` plan operator shown in Figure 3.8, which takes as a parameter the name of a device (e.g., a light-unit). If the operator's constraints are satisfied, i.e., the knowledge base contains structural information about the device, then the subgoals have EDGE plan content that indicates the process and behavior associated with the device. This `how-it-works` plan operator, together with those for process and behavior, were used to produce the partial text plan shown in Figure 3.9 which first describes the structure, then the process, and finally the behavior of a light-unit. The structure of a light-unit was included because an assumed user model indicated the user did not know about it, and knowing the structure of the device is a precondition for indicating the process or behavior of a device (see Figure 3.8).

```
                      how-it-works (device)
CONSTRAINTS:          ((getslot device 'structure))
PRECONDITIONS:        ((structure (device))
SUBGOALS:             ((cplan process (device))
                       (cplan behavior (device)))
```

Figure 3.8 `how-it-works` Content Plan Operator (cplan)

In addition to planning content, EDGE also has a number of discourse level plan operators that structure interactions with the user, using techniques analogous to Power's (1974) dialogue games. For example, EDGE has plan operators that open and close exchanges. Figure 3.10 illustrates the above description of the light-unit embedded in a dialogue. The overall informing transaction in Figure 3.10 is first opened with a framing move ("ok") which is followed by a focusing move ("I'll explain how the light detector circuit works"). The discourse plan next conveys the content describing how the light-circuit works. Finally, the closing exchange is initiated by a request to close ("Is that enough about how the light detector circuit works?") which is acknowledged by the user ("OK"), which ends the transaction.

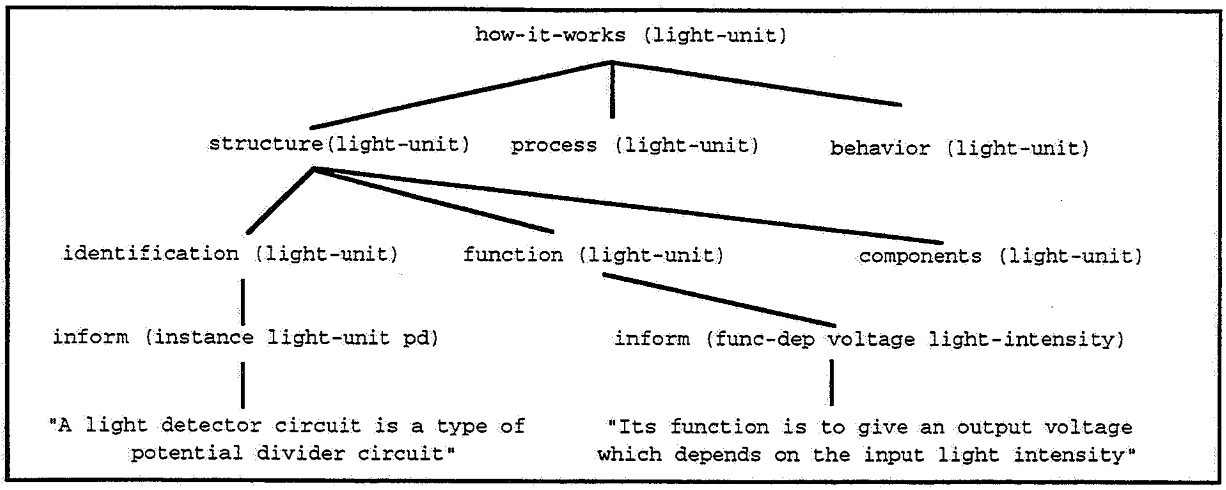



Figure 3.9 EDGE Content Plan

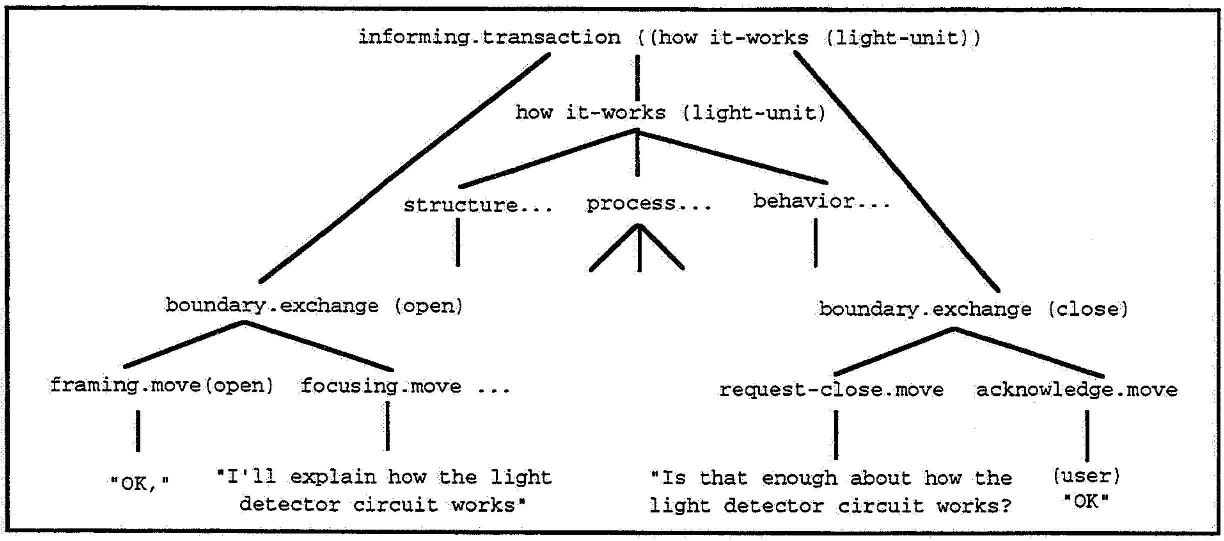



Figure 3.10 EDGE Discourse Plan

Cawsey's system shows how a plan-based approach to communication can be extended to incorporate dialogue control. EDGE's content plan operators, however, are limited to giving explanations of how devices work and have no notion of rhetorical acts such as describe, compare or narrate (although leaf nodes include inform and request illocutionary acts). In contrast, this dissertation provides plan operators that address a broader range of explanations although it does not address the issue of explanatory dialogue.

## 3.12 Illocutionary Schema

Researchers have identified plan-based strategies in text and exploited these to organize text. We have just examined Moore's `recommend-enable-motivate` strategy. At the same level of abstraction, Maybury (1989) suggested the plan `identify-support-recommend` which identifies a problem (in this case a rule constraint failure in an expert system), supports this identification with evidence, and finally tells the hearer how to recover from it. Similarly, McCoy's (1985ab) work on addressing misconceptions (misclassifications and misattributions) uses the strategy `deny-correct-support` to "debug" the knowledge or belief state of the hearer. Finally, Rankin (1989) guides the critique of a doctor's diagnosis/suggested-treatment using the plan `warn-justify-suggest_alternative`. These plans are analogous to McKeown's (1982) schemas which capture standard patterns of rhetorical predicates, except that the emphasis here is on sequences of communicative acts, be they speech acts or rhetorical acts. I term these pragmatic plans *illocutionary schemas*. One aim of this dissertation is to break down these illocutionary schemas into their primitive elements so that they can be recomposed guided by the context of the conversation. Decomposing these (illocutionary) schemas into their "uncompiled" versions requires not only optionality of text constituents, but also flexibility of order. For example we may want to recommend an action at the beginning or end of a text (i.e., before or after providing motivation) based on the information we are conveying as well as the context (e.g., the state of the speaker and hearer).

An alternative to this top-down decomposition approach is to have the pragmatic organization of the text arise from reasoning about the speaker's goal, the hearer's knowledge and belief state, and the propositional content of the text (e.g., the recommended action). This is analogous to the observation that rhetorical schemas are "frozen plans" that tell *how* to compose a text but fail to indicate *why* choices are made. In more effective, targeted output, composition requires planning at the level of *perlocution* (the intended effect), *illocution* (the sequence of speech acts), and *rhetoric* to achieve a communicative goal. The ordering of relations (e.g., MOTIVATION) which satisfy higher goals (e.g., CONVINCE) should therefore be selected in a principled way dependent on discourse context, not because they are the first operators to unify against the higher level plan. As mentioned previously, RST (the theory) recognizes (but does not formalize) that relational orderings are mere preferences and could be modified by, for instance, conversational context.

## 3.13 Why Build a Text Plan?

As the accounts given in this chapter so far show, there have been several systems that plan individual speech acts or multisentential text, research in the latter has conflated the distinct notions of *text plan* and *text structure*. Thus, although both Hovy and Moore use a hierarchical planner that attempts to achieve a discourse goal using a library of plans formalizing RST relations, the end result of their planning process is a hierarchical text structure of rhetorical relations which structure propositions (in Hovy's case) or illocutionary actions (in Moore's case). The problem is that (hierarchical) planning is an activity which produces a plan, i.e., a decomposition of actions and subactions, which can then be executed to achieve a particular effect (or effects). But rhetorical relations are not executable actions. In contrast, TEXPLAN produces a decomposition of (communicative) actions which are then executed, the process of which results in a (potentially multisentential) text. Thus rhetorical relations are a by-product of rhetorical actions (e.g., giving evidence implies an evidence relation) just as perlocutionary effects are the by-product of illocutionary actions. This final section argues for the need to construct a text plan as opposed to a text structure (although it can be argued that the former is included in the latter as it is inherently hierarchical.)

As the planner in TEXPLAN (detailed in the following chapter) attempts to achieve some given discourse goal it reasons along pragmatic (i.e., intention and belief), rhetorical, and epistemological dimensions. The history and result of this planning activity is recorded in a hierarchical text plan. This text plan is needed for a number of reasons. First, as pointed out in Chapter 2 and summarized at the beginning of this chapter, text schemas (a la McKeown, 1982, 1985) are inadequate because while they record standard patterns of organization (in essence "frozen" plans), they fail to capture the rationale underlying those patterns (i.e., their intended effect). This makes miscommunication recovery or replanning impossible except at the highest level of organization (i.e., attempting to use another schema). Second, a text plan is needed from both epistemological and linguistic perspectives. In particular, in most knowledge bases there is not enough structure (e.g., attribute structure), order, or indication of what is salient for efficient and effective communication between a human and machine. Linguistically, a text plan is needed to abstract away from syntactic and semantic details and to relate domain content and discourse structure to higher-level discourse goals. Therefore, the text plan acts as a communicative interface to the propositional content of the underlying application. Third, the text plan acts as a historical record of both the system's reasoning about pragmatics, rhetoric, and epistemology and the expected effect of this on the hearer's knowledge, beliefs, and desires.

In addition to the advantages of a deeper and richer representation, the text plan's hierarchical structure helps to guide surface choices. For instance, rhetorical relations between different parts of a text (their order and subordination) help guide the selection of connectors which can aid in local cohesion. For example the "illustration" relation suggests the connective "for example" and a "justification" or "cause-effect" relation between two parts of text may motivate the use of "because" as in "X because Y".

Just as rhetorical structure can motivate connectives, the subordination relationships captured by the text plan can be exploited by a focus model to make appropriate context space shifts. For example, when the text plan indicates the introduction of subclasses or subparts of an entity (e.g., an object, action, process, or state) these new entities can be pushed onto the current focus space (assuming local focus is implemented as a stack (Sidner, 1983; Grosz and Sidner, 1986)). In contrast, when the underlying application indicates the need to communicate a warning or an urgent message, the resulting text plan can signal to the focus mechanism to suspend (or push) the current attention space. Equally, hearer feedback can interrupt the current discourse focus, introduce new foci, or even return to old ones. Furthermore, the text plan implicitly incorporates a notion of focus, although independent mechanisms for global focusing (Grosz, 1977) and local focus shift (Sidner, 1979) are necessary to guide planning, as in the selection of alternative propositions (McKeown, 1982; Hovy and McCoy, 1989).

A plan-based approach to communication thus yields a number of theoretical and computational advantages. First, plan operators capture the conditions for using utterances designed to achieve specified perlocutionary effects. A second benefit is the computational efficiency offered by plan operators: they constrain the search space and encode constraints on the selection and ordering of the primitive elements of text. Pruning the search space will become increasingly important as text generators are scaled-up to capture hundreds of alternative structuring and ordering possibilities. Third, and most significant, the resulting text plan incorporates a more detailed representation of the structure and function of text which allows for greater control over the communication than previous approaches. As was indicated, text structure can guide attentional shifts as well as surface choices to aid local cohesion. Finally, because the resulting text plan is "self-conscious", it can react (by replanning) if queried or "poked" by the reader (Moore, 1989). Given these general arguments for plan-based models of communication and their corresponding text plans, the remainder of the thesis considers four major types of text from this point of view: description, narration, exposition, and argument.

## 3.14 Conclusion

A plan-based approach to communication has proven fruitful in integrating linguistic and extra-linguistic action. While initial research focused on the generation of isolated speech acts and referring expressions, the extension of planning to multisentential text aims to formalize the perlocutionary effects of rhetorical devices, record dialogue history, and replan failed communications. However, previous work has only partially investigated the illocutionary structure and rhetorical form of explanations and has therefore, only partially dealt with the requirements of multi-sentence generation. Thus the next chapter details how TEXPLAN uses communicative plans to generate descriptive texts.

# Chapter 4

# DESCRIPTION

Praise no man before thou hearest him speak, for this is the test of men.

*Ecclesiastes* 27:7

## 4.1 Introduction

This chapter begins by defining text. In doing so it classifies four major types of text -- *description*, *narration*, *exposition*, and *argument* -- each of which perform distinct functions in communication. The four major classes are types of text rather than genre and may be mixed in actual prose. With the aim of formalizing these text types as plans, the chapter describes an extended first-order predicate calculus plan operator language which is used to represent the preconditions, constraints, effects, and decomposition of communicative acts (rhetorical acts and speech acts). Taken as a whole, these plan operators capture the rhetorical structure, pragmatic function, and epistemological content of the four classes of text. The chapter presents some human descriptive text, and follows these with a formalization of the main types of description as plan operators. The descriptive plan operators include *definition*, *detail*, *division*, *extended description*, *comparison*, and *analogy*. Throughout, the chapter illustrates how TEXPLAN's explanation planner reasons about these plan operators to produce hierarchical text plans which are then linguistically realized as English text. The chapter concludes by discussing a direction for future research: figures of speech.

## 4.2 What is Text?

To constrain the scope, this discussion focuses on written text and does not consider properties of spoken discourse such as prosody. The Concise Oxford English Dictionary defines "text" as:

```
(1) original words of author as opposed to a paraphrase or commentary on
    them.
```

```
(2) a passage of scripture quoted as authority especially as chosen as
    subject of sermon etc;  subject, theme.
```

But the definition of "texture" is more suggestive:

```
arrangement of threads etc. in textile fabric, characteristic feel to
this; arrangement of small constituent parts, perceived structure;
representation of structure and detail of objects in art; quality of
sound formed by combining parts.
```

Perhaps this characterization led Halliday and Hasan (1976, p. 2) to state that "a text has texture and this is what distinguishes it from something that is not a text ... the texture is provided by the cohesive relation."

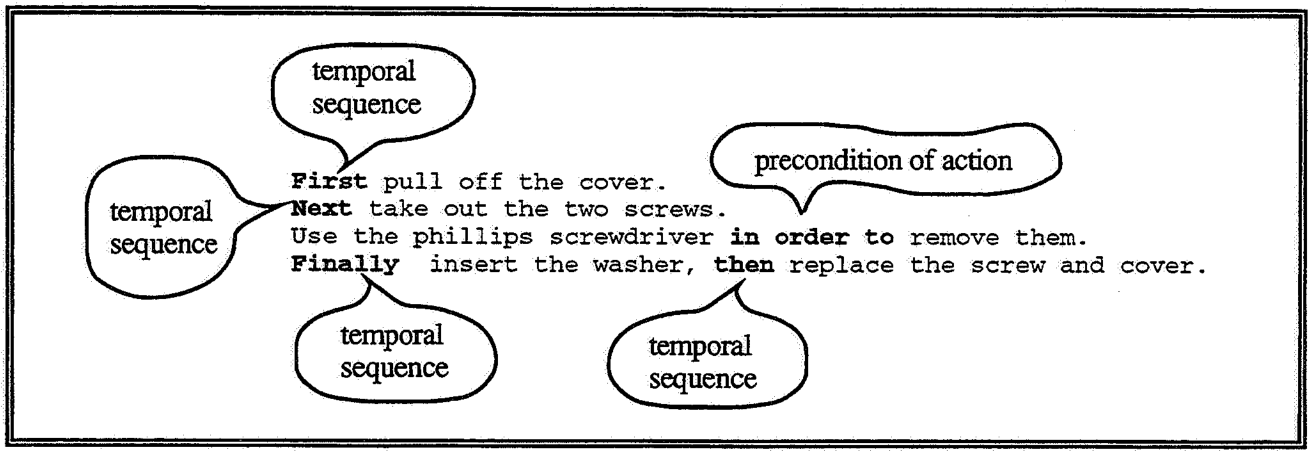
### 4.2.1 Cohesion

This connective relationship manifests itself in text when interpretation of an utterance presupposes knowledge of a previous utterance. For example, a cohesive relation can exist as an anaphor. Consider the following utterance, 1, with the alternative successors, 2a-2e:

*1. Jack swatted the flies.*
2a. He killed them.
2b. This killed them.
2c. It was horrible.
2d. One of them got away.
2e. I had difficulty saying that.

In 2a, the personal pronoun "he" refers to "Jack". In 2b, the definite pronoun "them" refers to the preceding definite noun phrase "the flies". In 2c, the sentential "it" refers to the entire preceding utterance. The indefinite, one-anaphora in 2d, in contrast, refers to some member of the set of flies. In addition to backward referring anaphora, discourse can be connected with cataphora (forward reference) and also exophora (extra-textual reference). Furthermore, a text may refer not only to an object but also to an action ("this" in 2b above) or to segments of discourse ("that" in 2e above) (Webber, 1988). Referring expressions are not restricted to noun phrases as evidenced by temporal and locative adverbials (e.g., "three minutes later", "five miles away") which refer to the time and location of previously introduced events or entities in the discourse. As these adverbial examples show, the referring expression need not point directly at a previously mentioned entity (as in pronominalization), but rather at some characteristic (e.g., time or location) associated with a preceding entity in the text.

Utterances can also be unified through formal markers such as "and", "however", "for example", and "then". The following series of instructions illustrates how markers can indicate discourse relations:




Several grammarians have classified connectives (Quirk and Greenbaum, 1973; Halliday and Hasan, 1976). Halliday (1985, p. 302-307) offers a taxonomy of such markers: elaboration, extension, enhancement. Extension, for example, can be additive ("and", "also", "moreover", "in addition"), adversative ("but", "yet", "on the other hand", "however"), or variation ("on the contrary", "apart from that", "alternately"). He relates surface forms with these connectives, illustrating their cohesive function in discourse. Similarly, Reichman (1985) enumerates a number of "cue words" and their relation to argument structure. However, these words are often in themselves ambiguous (e.g., "and" in Halliday's terminology can be both additive and variation). Also, while cue words may indicate underlying structure, they are not always present. Connective relation of text can be implied rather than explicit as in a list of historically significant dates, a series of actions, or a sequence of events.

Many devices aid the cohesion of text beyond coreference and connectives, including ellipsis, deixis, lexical relationships (synonymy, hyponymy, part-whole, collocability), structural relationships like clausal substitution (e.g., "so am I"), syntactic repetition, consistency of tense and stylistic choice (see Quirk and Greenbaum, 1973, pp. 284-308) as well as maintaining focus of attention (e.g., Sidner, 1983).

### 4.2.2 Coherence: The Form and Function of Prose

Whereas cohesion arises from textual linkages, *coherence* stems from the conceptual integration of the text content. Halliday and Hasan (1976, p. 229) claim that the heart of coherence "is the underlying semantic relation" and suggest a taxonomy of *coherence relations* such as "elaboration" or "contrast" which can hold between utterances. Hobbs (1979) investigates the processing required to establish that coherence relations hold between two given sentences and argues that anaphoric resolution is aided by the hearer's recognition of underlying coherence relations. Cohesion, while it might be seen as a consequence of coherence, is more properly viewed as support for coherence. Both are necessary for effective text since neither alone is sufficient to guarantee it.

Text coherence, the less understood of coherence and cohesion, relies to a great extent upon the structure and function of the discourse. While cohesion arises from local connective devices, global coherence is maintained by the selection, structuring, and ordering of content that is relevant to the goals of the discourse.

Not surprisingly, human writers learn special forms of discourse to produce specific effects on their readers (Brooks and Hubbard, 1905; Kane and Peters, 1959; Brown and Zoellner, 1968; Dixon, 1987; Pickett and Laster, 1988). In particular, they are taught how to write about people, places and things (*description*), events (*narration*), ideas and methods (*exposition*) as well as convictions (*argument*). These types of text are shown in Figure 4.0. Humans learn how to produce particular *rhetorical forms* (such as definition or comparison/contrast), how to compose types of text like instructions or formal proofs, and how to write longer forms such as technical reports, market surveys, and stories, which can exploit these text types in various, though often conventional ways. We explicitly distinguish between a *rhetorical act* (e.g., to define) and the result of that action, a *rhetorical form* (e.g., a definition).

The four principal types of text -- description, narration, exposition, and argument -- have distinct purposes. Description informs the hearer about entities (e.g., a person, place, thing, action, event, process, or state), and uses (see Figure 4.0) rhetorical techniques such as definition, detail, division, comparison/contrast, and analogy, as illustrated by dictionary entries or travel brochures. The purpose of narration is to relate events to the hearer so its uses rhetorical techniques like temporal sequencing ("on the first day...", "on the second day ...") or spatial sequencing (e.g., "in England ...", "in France ...") as in newspaper or weather reports. As Figure 4.0 illustrates, narration includes not only reports, but also stories and biographies. Exposition is intended to make the hearer understand complex methods, processes, or ideas, although the hearer may not actually subscribe to them, and so uses not only descriptive and narrative techniques but also devices that identify entities, enable actions, and indicate cause/effect relations as in operational instructions. To convince the hearer to believe a proposition and/or persuade the hearer to act requires the final form of text: argument. Arguments that make claims support these with evidence, causes, and logical reasoning as in the three forms of syllogism shown in Figure 4.0 -- categorical, disjunctive, and hypothetical; arguments that attempt to evoke action use techniques such as indicating the purpose or positive consequences of the action as in advertisement. These distinctions between forms are broad. Particular devices like definition may support different purposes, and instances of one form may subsume instances of others, e.g., exposition may subsume description. These types of text can and often do serve multiple purposes (e.g., to simultaneously inform and frighten).

More particular types of text have individual characteristics such as lexical/syntactic properties (e.g., process instructions use second person, present tense verbs whereas detective stories commonly use third person, past tense) as well as organization attributes (e.g., the content of locational directions, travel brochures, and room descriptions (Linde and Labov, 1975) is typically grouped and ordered spatially). Despite these types of genre-specific characteristics, however, some general principles apply to all discourse. All text achieves effect, limited though it may be. Furthermore, discourse seems to be governed by general rules concerning focus maintenance and shift and communicative structure (Sidner, 1983;

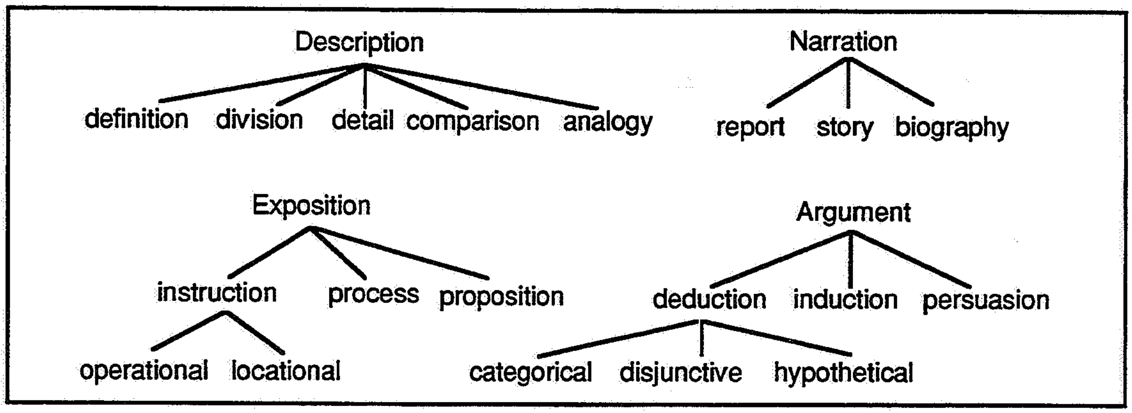



Figure 4.0 Classification of Text Types

Grosz, 1977; Sidner and Grosz, 1986). It has been claimed (van Dijk and Kintsch, 1983) that text is organized hierarchically so lower level elements are clearly related through higher level conceptual grouping. Perceptual saliency (van Dijk, 1977) also seems to determine normal ordering so that general comes before particular, whole before component, set before element, including before included, large before small, outside before inside, and possessor before possessed.

Just as humans learn to write prose, so too a machine must be taught to compose by capturing the structure and purpose of text classes as in Figure 4.0. The remainder of this thesis investigates the computational representation and production of these four classes of text. The thesis is organized around these four classes, discussing description first (current chapter), narration second (Chapter 5), exposition third (Chapter 6), and argument last (Chapter 7). We begin by examining perhaps the most common form of text: *description*. Its purpose is to inform the hearer of some entity. Authors describe things using a variety of rhetorical forms including definition, detail, division, comparison/contrast, and analogy. Furthermore, these techniques can be combined to produce more complex descriptive texts. The remainder of this chapter illustrates each of these techniques and formalizes them as plan operators for TEXPLAN.

## 4.3 Logical Definition and Entity Differentia

A common method of describing an entity (i.e., an object, action, event, process, or state) is to define it. Perhaps the oldest form of definition is the logical (also called formal) approach which was first espoused by Greek orators (e.g., Plato, Aristotle, Cicero, Isocrates). It consists of identifying a term (*species*) by its class (*genus*) and its distinguishing characteristics (*differentia*). Consider:

A parallelogram is a quadrilateral whose opposite sides are parallel.
*(species)* *(genus)* *(differentia)*

The order of the elements of a logical definition is variable: "A polygon of three sides is a triangle." TEXPLAN captures this technique in the *rhetorical predicate*, `logical-definition`. Logical definition is so common that one model under consideration for the lexeme template of the proposed Oxford machine-readable dictionary defines an entry with respect to a genus and a number of differentia (Atkins, 1989). Since the parents of entities are generally explicitly encoded in a generalization hierarchy found in most knowledge based systems, the genus of an entity can be easily retrieved. Differentia are more complex. In current systems the distinguishing features of the entity (e.g., a brain is unique from other organs because of its function and location) are hand-encoded in the knowledge base (McKeown, 1985; Maybury, 1987). In contrast to this labor-intensive approach, and motivated by Tversky's (1977) set theoretic approach to object similarity as well as the psycholinguistic experiments of Collins and Quillian (1969) and Rosch (1973), a differentia algorithm was developed as part of this research which automatically generates an entity's distinctive characteristics in a domain independent manner by reasoning about the attributes and values of the entity as well as those of closely related entities.[1] Since differentia selection is "on-line", it can be modulated by context and perspective. Because of the length of the derivation of the algorithm, it is presented in Appendix A. It is based on two numerical measures which are used to select distinguishing attributes and values (i.e., the differentia) of a given entity. The first measure, P, indicates the *prototypicality* of a given attribute or attribute value pair (i.e., its commonness). The second measure, D, indicates the *discriminatory power* of a given attribute or attribute value pair (i.e., its uniqueness). Both measures are dependent upon the context of related entities in a generalization hierarchy (e.g., if some feature `f`, is characteristic of some entity, `e`, as well as of all its siblings, then `f` is not very discriminating of `e`.) A composite of prototypicality and discriminating power yields the *distinctive power*, DP, of an attribute or attribute value pair of an entity. Using this measure, the distinctive features of an entity -- its differentia -- can be selected.

### 4.3.1 Computing Entity Differentia

Motivated by Rosch's psycholinguistic examples, and in order to test the illustrative plausibility of the differentia algorithm, a vertebrate knowledge base was implemented as an experimental domain. For example, calculating the distinctive features of the class object *vertebrates*, the algorithm uses the above measures of prototypicality (P), discriminatory power (D), and distinctive power (DP) to collect features common to its children (e.g., vertebrates have a nervous system and a segmented spinal column) and then to determine which features are unique with respect to its siblings (e.g., invertebrates don't have spinal columns).

Table 4.1 shows the calculated values of P, D, and DP for attribute-value pairs of the class *bird* ordered in terms of DP. Using the DP value we can select the most distinctive features of a bird: its flying

[1]Distinctive features are important not only in entity discrimination in a logical definition, but also in referent identification as in definite noun phrase interpretation or generation.

motion, wings, feathers, and seed-eating characteristics. Similarly, a canary is identified as a domesticated, singing, yellow bird from the Canary Islands.

| ATTRIBUTE-VALUE PAIRS | P | D | DP |
|---|---|---|---|
| (movement flies) | 1.00 | 1.00 | 1.00 |
| (propellors wings) | 1.00 | 1.00 | 1.00 |
| (covering feathers) | 1.00 | 1.00 | 1.00 |
| (eats seeds) | 0.88 | 1.00 | 0.94 |
| (blood warm) | 1.00 | 0.75 | 0.88 |
| (subparts (crest crown bill tail ...)) | 1.00 | 0.00 | 0.50 |
| ... | | | |
| (segmented-spinal-column t) | 0.00 | 0.00 | 0.00 |

Table 4.1 Prototypicality (P), Discriminatory power (D), and Distinctive Power (DP) of attribute value pairs of object class #<bird>.

The generator uses the differentia algorithm to select the propositional content of a logical definition by (1) retrieving the parent(s) of the entity to be defined and (2) selecting the characteristics with maximum DP. This is illustrated below with logical definitions generated by TEXPLAN from knowledge bases in a variety of domains (Maybury, 1990; 1988, 1987, 1989, respectively):

A canary is a yellow bird with a Canary Islands origin, that sings, and is domesticated.
A brain is an organ located in the skull consisting of gray nerve tissue and white nerve fibers.
An optical lens is a component for focusing located in a camera.
An A-10 is a fighter for air-to-ground interdiction.[2]

Note that the most distinctive characteristic(s) can be given a prominent surface position (or intonation), such as modifying the head noun in the object position. For example, notice above how "yellow", the most salient property of a canary, modifies the subject "bird."

A variation on the logical definition replaces the differentia component with the purpose or constituents of the entity. Consider:

> A bicycle is a light vehicle having two wheels, one behind the other, a steering handle, a saddle seat(s), and pedals by which it is propelled. (Webster's New Collegiate Dictionary, 1957)

The use of purpose in place of differentia is illustrated by the fourth example given above. The distinguishing characteristics of an A-10 are computed by recognizing that other classes of aircraft (e.g., tankers/cargo, reconnaissance, etc) have similar attributes (e.g., speed, empty and loaded weights, turn-times, etc.) and only slightly differing values. However, they do have unique tactical roles or purposes,

[2]All military data is unclassified and was obtained from public sources (e.g., *Jane's Aircraft Almanac*).

and this is what distinguishes the A-10 from them. While function and subpart information may appear as part of an unconventional logical definition, subsequent sections illustrate their use in other forms of description.

## 4.4 Synonymic and Antonymic Definition

As Figure 4.0 indicates, there are three principal forms of definition: logical, synonymic, and antonymic.[3] In contrast to logical definition, synonymic definition, while less explicit, can be very effective when the hearer knows entities related to the one being defined. For example, to define the term despot, a speaker can say, "A despot is a tyrant" and s/he will thus define the term synonymically. Similarly, consider Soviet or American fighters. They can be identified logically using their technical name ("A MiG-25 is a Soviet fighter for air-to-air interdiction.") or synonymically using their nickname ("A MiG-25 is a Foxbat").[4]

Antonymic definition, on the other hand, contrasts the entity with what it is not. Consider the following passage which first defines an action synonymically and then antonymically.

> To madden, which means to infuriate or enrage, is the opposite of to calm, pacify, or assuage.

Apart from logical, synonymic, and antonymic definition, a writer can tersely describe an entity by detailing it. Detail concerns a number of techniques including characterizing the key features of an entity (*attribution*) and indicating the *purpose* of an entity. An author can also give a particular *illustration* or example of the unknown thing, as in "A Yorkshire Terrier is an example of a dog."

In addition to detail, a writer has access to the two techniques of division: classification and constituency. *Classification* is related to logical definition except that instead of defining an entity in terms of its superordinate(s) in a generalization hierarchy, the speaker identifies its subordinate(s):

Plane figures are circles, squares, rectangles, and triangles.

Note, however, that the axis of classification may be varied. For example, we may classify triangles as scalene, isosceles, or equilateral. But if we characterize triangles by their angles and not their sides, there are two subclasses: right and oblique. (These subsets must be mutually exclusive to yield a rigorous classification.)

---

[3]Etymological definition is another form not addressed by this thesis as this information is typically not available in most knowledge based systems.
[4]Synonyms are often used to identify entities in definite noun phrases (e.g., "The MiG-25 Foxbat shot down the F-15E Eagle.").

While classification is based on subtypes, *constituency* identifies the constituents or subparts of an entity, for example the bicycle description given earlier that details a bicycle's components. Equally, one can describe a road by its segments, an entree by its ingredients, or a set by its elements. Similarly, an event can be decomposed into subevents, a process into subprocesses, and actions into subacts.

Apart from detail and division, there are several common methods for entity description. Frequently writers provide an incomplete logical definition and describe an entity's differentia but not its genus, assuming this can be inferred. Two other, possibly lengthier methods entail comparing or contrasting the unknown entity with entities the hearer/reader is familiar with and, alternatively, using an analogy. In each of these cases, it is essential that the information used to describe the entity (be it characteristics, examples, or some related entity) is familiar to the hearer. In this manner, the hearer can forge a correlation between the unknown entity and familiar entities. These techniques are found in many forms of communication and so sections 4.9 and 4.10 are dedicated to their formalization.

Following the philosophy of communication as a planned activity, the next section begins formalizing the rhetorical actions which underlie the text types shown in Figure 4.0. Rhetorical acts are formalized as plan operators which capture the necessary and sufficient conditions for using individual rhetorical acts as well as their expected effect on the addressee. These plan operators are manipulated by a general mechanism (a *text planner*) which reasons about individual operators to produce a hierarchical *text plan* which can then be executed (i.e., linguistically realized or uttered) in an attempt to achieve some given discourse goal. A *discourse goal* is stated in terms of some intended effect on the addressee's knowledge, beliefs, or desires (e.g., convince the addressee to believe some P). The text planner relates discourse goals to the effects of individual plan operators in the plan library. The remainder of this chapter details TEXPLAN's descriptive plan operators beginning with the formalization of logical, synonymic, and antonymic definition.

## 4.5 The Formalization of Definition as Plan Operators

The act of defining is encoded in a plan language much like those discussed in the previous chapter. Each *communicative act* -- either a rhetorical act, an illocutionary speech act, or a surface speech act -- is represented as an operator in a library of plans which are reasoned about by a hierarchical text planner (Sacerdoti, 1977) inspired by previous text planning formalisms (Hovy, 1988a; Moore, 1989). Communicative acts have specific enabling conditions, effects on the hearer, and decompositions. A *rhetorical act* concerns a more general level of abstraction than speech acts (e.g., describe, define, compare) and may employ other rhetorical acts and/or speech acts to achieve its goals. A *speech act* (Searle, 1969, 1975) refers to the illocutionary force of utterances (e.g., inform, request, warn, promise). Illocutionary speech acts are achievable by *surface speech acts* (Appelt, 1985) (e.g., assert, command, ask, recommend) which characterize the locutionary form of utterances and therefore are associated with particular surface forms (e.g., declarative, imperative, interrogative).

The propositional content of a speech act may be a *rhetorical predicate* whose function is to abstract particular kinds of information from a knowledge base (e.g., constituency predicates refer to subparts of entities whereas classification predicates refer to subtypes of entities). In order to maintain domain independence, predicate semantics, like those used in TEXT (McKeown, 1982, 1985), connect rhetorical propositions such as classification and constituency to the underlying application knowledge. The independence of the plan operators is illustrated by generating text from several knowledge formalisms (e.g., frames, rules, FRL, PCL) in a variety of domains (e.g., neuropsychological diagnosis, mission planning (KRS), battle simulation (LACE), vertebrate classification, and photography fault detection) using the same rhetorical predicates as primitive text elements. Some rhetorical predicates, called *rhetorical relations*, characterize both propositional content and by their nature associate different parts of a text. That is, `evidence` is a rhetorical relation that characterizes how a proposition (evidence) supports some statement (claim). In contrast, `logical-definition` is simply a 'stand-alone' rhetorical predicate that includes the genus and differentia of an entity.

Like conventional planners, each plan operator defines the preconditions that must hold before a communicative act can be executed, the constraints on the act occurring, its intended effect, as well as its refinement or decomposition into subactions. Constraints encode both physical and cognitive restrictions on the model of the world or the models of the agents that guide plan operator selection (e.g., if there are no instances of a concept in the knowledge base then an exemplification plan operator cannot be invoked). Unlike constraints, preconditions indicate states of affairs that enable the action to occur and so the planner can attempt to achieve these if they are false when the plan operator is invoked (e.g., if the hearer does not know about something that they should then the planner can try to achieve it). Preconditions distinguish between *essential preconditions* of a communicative act and *desirable preconditions*. This distinction allows the planner to make more sensitive choices when it has multiple alternatives that achieve the current goal.

```
NAME            describe-by-defining
HEADER          Describe(S, H, entity)

CONSTRAINTS     Entity?(entity) ∧ (HASTE(S) ∨ HASTE(H))
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, entity) ∧
                WANT(S, KNOW-ABOUT(H, entity))
  DESIRABLE     ¬ KNOW-ABOUT(H, entity)

EFFECTS         KNOW-ABOUT(H, entity)

DECOMPOSITION   Define(S, H, entity)
```

Figure 4.1 Uninstantiated describe-by-defining Plan Operator

For example the describe-by-defining plan operator shown in Figure 4.1 encodes the communicative act of the speaker (S) describing an entity (e.g., an object, action, event, process, or state) by defining it so that the hearer (H) knows about it. That is, given some entity, if the speaker has the intention that the hearer know about it, this can be achieved by any communicative act (formalized as a plan operator) that defines an entity. The constraints, preconditions, and effects of plan operators are encoded in an extension of first order predicate calculus. Boolean algebra notation is used to indicate logical conjunction and disjunction (∧ and ∨ respectively). Predicates have true/false values and are in lower-case type with their initial letter capitalized (single argument predicates are further distinguished by a trailing question mark (e.g., Entity?)). In contrast, functions return a range of values and appear in lower-case type. Both predicates and functions appear in constraints, preconditions, and effects of plan operators. In contrast, communicative acts appear in the header or decomposition of plan operators and are in lower-case type with their initial letter capitalized. The decomposition of a plan operator may include optional and alternative communicative acts. Arguments to predicates, functions, and communicative acts include variables and constants. Variables are italicized (e.g., *entity*) and constants appear in upper-case plain typeface. For example, in Figure 4.1 the header Describe(*S, H, entity*), indicates the name of the communicative act, Describe, and the three arguments to it, the variables *S, H*, and *entity*.

Intensional operators, such as WANT, KNOW and BELIEVE appear in capitals. KNOW details an agent's specific knowledge of the truth-values of propositions (e.g., KNOW(H, Red(ROBIN)) or KNOW(H, ¬Yellow(ROBIN))) where truth or falsity is defined by the propositions in the knowledge base. That is, KNOW(H, P) implies P ∧ BELIEVE(H, P) and so an agent can hold an invalid belief (e.g., BELIEVE(JOHN, Yellow(ROBIN))). It follows, then, that any particular agent's knowledge, which is called the agent's model, is a subset of the knowledge base. KNOW-ABOUT is a predicate that is an abstraction of a set of epistemic attitudes of some agent toward an individual. An agent can KNOW-ABOUT an entity or action (e.g., KNOW-ABOUT(H, ROBIN) or KNOW-ABOUT(H, FLYING)) if they KNOW its superordinate, characteristics, components, subtypes, or purpose.

Because models of the knowledge, beliefs, and desires of the speaker and hearer as well as a representation of the discourse structure are maintained, the system is able to avoid repetition and prune its communications. Unfortunately, a complication is introduced when reasoning about knowledge and beliefs, namely that of infinitely reflexive beliefs. Since the speaker must reason about what the hearer knows and believes, this includes what S believes that H believes, what S believes that H believes that S believes that H believes, and so on, ad infinitum. One common approach is to limit inference chains (say to three or five inferences). Another approach is recursive nested databases (Cohen, 1978) whereby the knowledge base is partitioned into sections termed *belief spaces*, where each belief space represents an agent's first-order knowledge. As belief space themselves can be objects, they can be linked to represent an agent's beliefs about other agent's knowledge (Allen, 1987, p. 462). TEXPLAN does not address this problem and avoids infinitely recursive beliefs by representing only an agent's beliefs about their own knowledge and beliefs about other agent's knowledge at one level of recursion. Furthermore, the system does not infer the logical consequences of all current beliefs, i.e., agents believe their explicit beliefs, not their inferable ones. Even humans demonstrate limitations in their ability to infer the logical consequences of all of their beliefs, so this restriction is less severe than it initially appears. Finally, belief representation and modification was not a principal focus of this work, although it would be possible to replace this belief component of TEXPLAN with one which deals with implicit beliefs, knowledge about the beliefs of others, mutual belief, and so on (cf. Allen, 1987; Levine, forthcoming).

TEXPLAN uses plan operators like that shown in Figure 4.1 to construct hierarchical plans to achieve specific discourse goals. Figure 4.2 shows the general flow of control during planning. The system begins to plan a text when a discourse goal is posted by the *discourse controller*, be it a communicative goal such as to get the hearer to know about X or a physical goal such as to make the hearer do Y. Both communicative and physical goals and actions are represented in the same plan language, thus joining linguistic and extralinguistic goals and actions.

```
1. DISCOURSE CONTROLLER POSTS DISCOURSE GOAL
2. FIND OPERATORS THAT ACHIEVE CURRENT GOAL
3. INSTANTIATE PLAN OPERATORS AND THEN SELECT THOSE
      THAT SATISFY CONSTRAINTS & ESSENTIAL PRECONDITIONS
4. PRIORITIZE PLAN OPERATORS
5. loop until succeed or no more plan operators
   a. SELECT NEXT MOST PROMISING PLAN OPERATOR
   b. PROCESS DECOMPOSITION OF CURRENT PLAN OPERATOR
      loop until succeed or fail
      for each SUBGOAL/PRIMITIVE ACT
        if SPECIAL-OPERATOR then PROCESS IT
        ELSE POST SUBGOAL AND RECURSE/EXECUTE PRIMITIVE ACT
6. RECORD PROGRESS/FAILURE IN HIERARCHICAL TEXT PLAN
```

Figure 4.2 Flow of Control for Text Planner

### 4.5.1 An Extended Example

The operation of the planner is best conveyed with an example. The following example is implemented in the domain of Air Force mission planning (Dawson et al, 1987). Assume that the hearer (in this example the user) asks the speaker (in this case the system) about KC-135[5] aircraft. Perhaps the user queries the system "What is a KC-135?" TEXPLAN assumes a linguistic interpretation component is able to translate this to the speaker's goal KNOW-ABOUT(H, KC-135) which is posted to the text planner as Step 1 in Figure 4.2 (so the text planner's actual input is currently a direct formal language input of the kind just mentioned). Next (Step 2) all operators whose effect matches this goal are found. This includes the describe-by-defining plan operator of Figure 4.1, which is instantiated to that shown in Figure 4.3. It also includes other definition plan operators, such as synonymic and antonymic, which are defined later. Those plan operators that satisfy the constraints and essential preconditions (Step 3) are then prioritized (Step 4). Working from this list of plan operators, the planner tries to execute the decomposition of each until one succeeds (Step 5). This involves processing any special operators (such as optional) or quantifiers (∀ or ∃) as well as distinguishing between subgoals and primitive acts.

For instance, assume the planner chooses the plan operator shown in Figure 4.3 (see the preference metric below for choosing among alternatives). The planner then attempts to execute its decomposition, Define(S, H, KC-135) (Step 5). Because it is a subgoal (as opposed to a primitive act), the planner recurses to Step 1. The text planner then uses a unification algorithm to find all plan operators from the library whose HEADER portion matches the current goal (Step 2). Because there are a variety of ways to define an entity, the planner instantiates the alternative subordinate plan operators and tests their constraints

[5]The printed version in the example (implemented in the Symbolics Common Lisp Object System) is #<KC-135>. For clarity, it is presented here simply as KC-135.

```
NAME            describe-by-defining
HEADER          Describe(S, H, KC-135)

CONSTRAINTS     Entity?(KC-135) ∧ (HASTE(S)  ∨  HASTE(H))
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, KC-135) ∧
                WANT(S, KNOW-ABOUT(H, KC-135))
  DESIRABLE     ¬ KNOW-ABOUT(H, KC-135)

EFFECTS         KNOW-ABOUT(H, KC-135)

DECOMPOSITION   Define(S, H, KC-135)
```

Figure 4.3 Instantiated `describe-by-defining` Plan Operator

and essential preconditions (Step 3). If, for instance, there is no knowledge of superordinates of a `KC-135` then the system may attempt a synonymic definition. If, however, multiple plan operators match the current posted goal and meet the constraints and essential preconditions, then these are ordered via a *preference metric* (Step 4). The preference metric prefers plan operators that:

- have fewer subplans (cognitive economy)
- have fewer new variables (limiting the introduction of new entities in the focus space of the discourse)
- meet all desirable preconditions (no need to plan other actions to enable current action)
- are more common or preferred in natural text (e.g., logical definition is preferred by rhetoricians over synonymic or other methods because of its precision)

While the first three preferences are explicitly inferred, the last preference is implemented by the sequence in which plan operators are listed in the plan library (Step 5a).

The plan operator `define-by-logical-definition` (see Figure 4.4) is one of the plan operators that can define an entity. A *logical definition* informs the hearer about the superordinate(s) and differentia of an entity, provided it is indeed an entity with a superclass and, preferably, that the hearer knows the entity's superordinate but does not yet know about the entity. While the example illustrates the logical definition of an object (a `KC-135` refueling aircraft), the plan operators for describing actions, events, processes, and states are analogous to that for objects where notions of classification, decomposition, and attributes/values are common to these entities. When the action `Define(S, H, KC-135)` unifies against the header of this plan operator, the variable *entity* is bound to the object `KC-135`, *S* is bound to S, and *H* is bound to H. These header bindings are used to instantiate the skeleton logical-definition plan operator of Figure 4.4 to that shown in Figure 4.5. Other plan operators are similarly selected and instantiated. The constraints and essential preconditions of the instantiated plan operators are tested and then any remaining plan operators are prioritized.

```
NAME            define-by-logical-definition
HEADER          Define(S, H, entity)

CONSTRAINTS     ∃c Superclass(entity, c)
PRECONDITIONS
  ESSENTIAL     ∃c Superclass(entity, c) ∧KNOW-ABOUT(S, c)
  DESIRABLE     ¬ KNOW-ABOUT(H, entity) ∧
                ∃c Superclass(entity, c) ∧KNOW-ABOUT(H, c)
EFFECTS         ∀x ∈ superclasses(entity)
                     KNOW(H, Superclass(entity, x)) ∧
                ∀y ∈ differentiae(entity)
                     KNOW(H, Differentia(entity, y))

DECOMPOSITION   Inform(S, H, Logical-Definition(entity))
```

Figure 4.4 Uninstantiated `define-by-logical-definition` Plan Operator

```
NAME            define-by-logical-definition
HEADER          Define(S, H, KC-135)

CONSTRAINTS     ∃s Superclass(KC-135, s)
PRECONDITIONS
  ESSENTIAL     ∃c Superclass(KC-135, c) ∧KNOW-ABOUT(S, c)
  DESIRABLE     ¬ KNOW-ABOUT(H, KC-135) ∧
                ∃s Superclass(KC-135, s) ∧KNOW-ABOUT(H, s)

EFFECTS         ∀x ∈ superclasses(KC-135)
                     KNOW(H, Superclass(KC-135, x)) ∧
                ∀y ∈ differentiae(KC-135)
                     KNOW(H, Differentia(KC-135, y))

DECOMPOSITION   Inform(S, H, Logical-Definition(KC-135))
```

Figure 4.5 Instantiated `define-by-logical-definition` Plan Operator

If the logical definition is selected, each item in its DECOMPOSITION (see Figure 4.5) becomes a subgoal which is posted to be achieved by the planner (Step 5b). Each subgoal in the DECOMPOSITION may involve processing the bold, special operator **`optional`** which allows for non-mandatory but possible text constituents. The first-order predicate calculus plan language allows for conjunction ($\wedge$) and disjunction ($\vee$) as well as quantification ($\exists$ and $\forall$). These will each be discussed below as they arise in examples.

If and when the decomposition of a plan operator succeeds, the communicative act represented by that plan operator is incorporated into a hierarchical structure representing the current text plan (Step 6). This hierarchical text plan records any selected plan operators, untried subplans whose constraints and essential

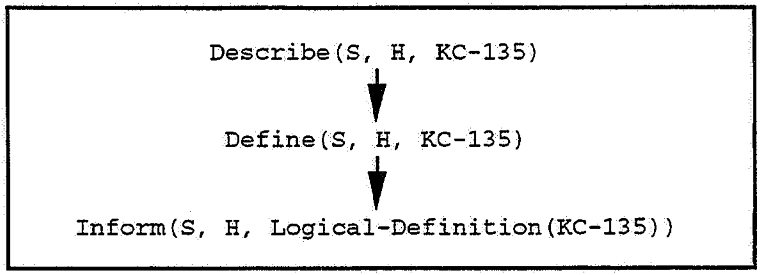



Figure 4.6 Partial Text Plan

preconditions were successful, failed plan operators whose constraints or essential preconditions failed, and plan operators that where tried but that the hearer rejected (reacting to user feedback is discussed in section 4.5.3). At this point in the `KC-135` example the text plan has the decomposition shown in Figure 4.6. In other words, in order to get the hearer to know about a `KC-135`, the speaker describes it by defining it, which is accomplished by informing the hearer of its logical definition.

Next the planner attempts to achieve the leafnode act `Inform(S, H, Logical-Definition(KC-135))`. This matches the header of the `inform-by-assertion` plan operator shown in Figure 4.7. This definition of inform is different from that found in some speech act work (e.g., Allen, 1987) in that the effect is not that the hearer believes the proposition, but simply that they believe the speaker believes it. Chapter 7 details techniques that can be used to convince the hearer to believe a proposition. In our example, the variable *proposition* in the plan operator unifies with the proposition `Logical-Definition(KC-135)` and the skeletal plan operator is instantiated to that shown in Figure 4.8. Since the constraints and essential preconditions are satisfied (i.e., it is a proposition, the speaker knows it, and the speaker wants to convey it to the hearer), the planner attempts to execute the decomposition.

At this point planning is halted. This happens in two ways. First, the planner may be unable to achieve a goal because no plan operators achieve the current goal(s) or because all possible operators failed since their constraints, essential preconditions, or decompositions failed. If this is the case, the planner backtracks and tries previous, untried alternatives. Second, the planner will stop if it encounters a primitive act. In the example, ASSERT is a primitive act. These planning primitives, called surface speech acts (Appelt, 1985), include ASSERT, COMMAND, ASK, and RECOMMEND and operate on individual rhetorical propositions. For instance, just as the speech act INFORM can be achieved by the surface speech act ASSERT (corresponding to declarative mood), the speech act REQUEST can be accomplished by the surface speech acts COMMAND, ASK, or RECOMMEND (corresponding to imperative, interrogative, and "obligatory" or "should" modal surface forms).

```
NAME            inform-by-assertion
HEADER          Inform(S, H, proposition)

CONSTRAINTS     Proposition?(proposition)
PRECONDITIONS
  ESSENTIAL     KNOW(S, proposition) ∧
                WANT(S, BELIEVE(H, BELIEVE(S, proposition)))

EFFECTS         BELIEVE(H, BELIEVE(S, proposition))

DECOMPOSITION   Assert(S, H, proposition)
```

Figure 4.7 Uninstantiated `inform-by-assertion` Plan Operator

```
NAME            inform-by-assertion
HEADER          Inform(S, H, Logical-Definition(KC-135))

CONSTRAINTS     Proposition?(Logical-Definition(KC-135))
PRECONDITIONS
  ESSENTIAL     KNOW(S, Logical-Definition(KC-135)) ∧
                WANT(S, BELIEVE(H,
                          BELIEVE(S, Logical-Definition(KC-135))))

EFFECTS         BELIEVE(H, BELIEVE(S, Logical-Definition(KC-135)))

DECOMPOSITION   Assert(S, H, Logical-Definition(KC-135))
```

Figure 4.8 Instantiated `inform-by-assertion` Plan Operator

For example, to request someone to open the window you can command them ("Open the window."), ask them ("Can you open the window?"), or recommend it to them ("You should open the window"). In TEXPLAN, the choice among surface speech acts, encoded in the constraints of the plan operators, is based on the class of text being produced. For example, requests to perform actions in instructions are realized as commands (e.g., "First take off the bolt.") whereas requests to perform actions in an argument are realized as recommendations (e.g., "You should take your medication."). An alternative approach would be to reason from first principles about the pragmatics of the conversation to determine the appropriate surface speech act (e.g., if the speaker is the hearer's supervisor, then a command is appropriate) although this requires detailed user modeling, not a principal focus of this work. Other plan operators for surface requests (e.g., "Can you ...") (Litman and Allen, 1987) or other indirect speech acts (Hinkelman and Allen, 1989) could be added to the plan library to operate at this level.

When the planner encounters the action `Assert(S, H, Logical-Definition(KC-135))`, it recognizes that ASSERT is a primitive surface speech act and queries the knowledge base for the logical definition of the object, KC-135. A procedure called `instantiate-rhetorical-predicate` returns the genus and differentia of the object KC-135 as the rhetorical proposition[6]:

```
(Logical-Definition
              ((KC-135))
              ((tanker (FUNCTION air-refueling))
               (transport-vehicle (FUNCTION cargo-transport))))
```

This rhetorical proposition is stored in the leaf node of the text plan, along with its surface speech act (i.e., ASSERT).

The final decomposition of the text plan is shown in Figure 4.9. A depth-first search routine linearizes this tree by following the selected paths. The resulting ordered list of surface speech acts with associated rhetorical propositions are mapped onto text by the linguistic realization component detailed in Chapter 8. The above example produces the final surface form:

```
A KC-135 is a tanker for air-refueling and a transport vehicle for cargo
transport.
```

Following McKeown (1982), a default focus position is associated with each type of rhetorical predicate so that discourse focus information can be extracted from the selected rhetorical proposition. For example, in the above logical definition, the entity in the first position of the rhetorical proposition (i.e., KC-135) is the default current discourse focus. There are also default focus positions for temporal and spatial focus (defined in the next chapter) in order to track changes in time and space, for example for predicates that convey information about events, states, and locations.

By taking advantage of a model of the hearer, the definition of a KC-135 could be even more effective. If, for instance, the user was identified as a member of a personnel airlift organization, then the utterance could identify only those classes and properties that would be of interest to that user and utter: "`A KC-135 is a transport vehicle for cargo-transport.`" In contrast, selecting information salient to a member of an air-refueling team might simply produce "`A KC-135 is a tanker for air-refueling.`" The danger of this limited disclosure, however, is that a hearer may draw false inference about object classification and properties, and TEXPLAN does not address the complex matter of user modelling in this sense.

[6]Details of the rhetorical proposition language can be found in (Maybury, 1987).

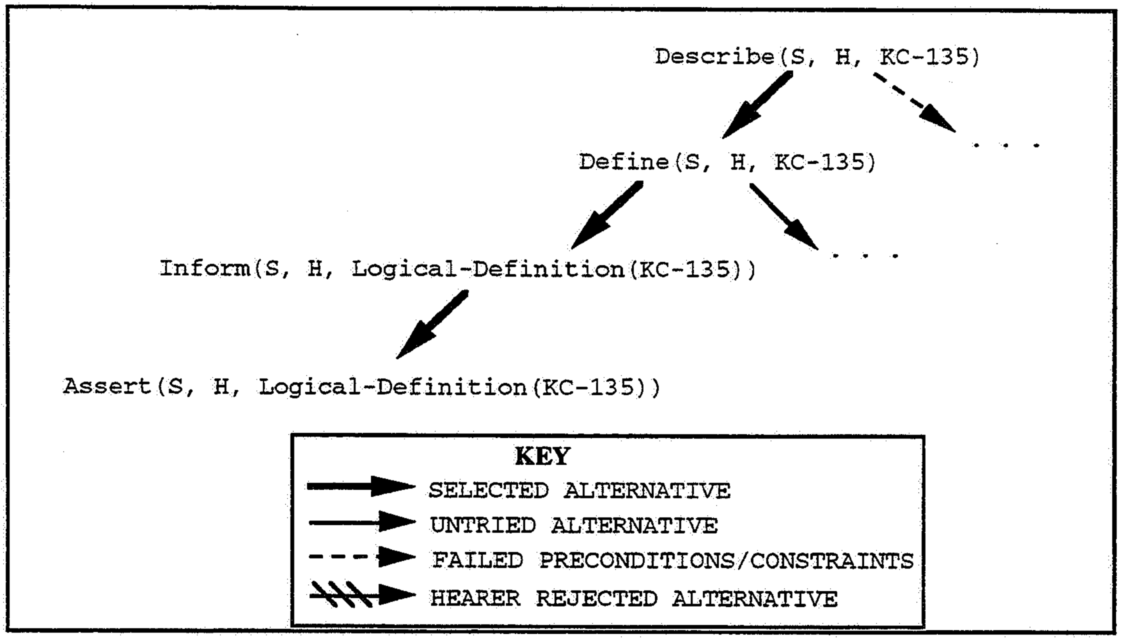



Figure 4.9 Hierarchical Text Plan for Logical Definition of a `KC-135`

### 4.5.2 Synonymic and Antonymic Definition

In the `KC-135` example, of all plan operators that could potentially achieve the goal `Define(S, H, KC-135)`, logical definition was chosen first based on the preference metric outlined at the beginning of the section. But if the constraints or preconditions of the logical definition plan operator had failed (e.g., if there is no superordinate of a `KC-135` in the knowledge base), then the planner would have attempted alternative actions. One of the alternative communicative acts, *synonymic definition*, is formalized as the `define-by-synonymic-definition` plan operator in Figure 4.10. Providing the knowledge base contains synonyms of the given entity (see the constraints in Figure 4.10) and the speaker knows at least one (see the essential preconditions), the decomposition of this plan operator informs the hearer of the synonymic definition of the entity. In our example, the header of the skeletal plan operator shown in Figure 4.10 matches the current goal `Define(S, H, KC-135)`. The plan operator in Figure 4.10 is then instantiated to the plan operator shown in Figure 4.11. Since a `KC-135` has a nickname (stratotanker) represented in the knowledge base and the hearer knows it, this plan operator is chosen.

```
NAME            define-by-synonymic-definition
HEADER          Define(S, H, entity)

CONSTRAINTS     ∃s Synonym(entity, s)
PRECONDITIONS
  ESSENTIAL     ∃s Synonym(entity, s) ∧KNOW(S, Synonym(entity, s))
  DESIRABLE     ¬KNOW-ABOUT(H, entity) ∧
                ∃s Synonym(entity, s) ∧KNOW-ABOUT(H, s)

EFFECTS         ∀x ∈ synonyms(entity)
                     KNOW(H, Synonym(entity, x))

DECOMPOSITION   Inform(S, H, Synonymic-Definition(entity))
```

Figure 4.10 Uninstantiated `define-by-synonymic-definition` Plan Operator

```
NAME            define-by-synonymic-definition
HEADER          Define(S, H, KC-135)

CONSTRAINTS     ∃s Synonym(KC-135, s)
PRECONDITIONS
  ESSENTIAL     ∃s Synonym(KC-135, s) ∧KNOW(S, Synonym(KC-135, s))
  DESIRABLE     ¬KNOW-ABOUT(H, KC-135) ∧
                ∃s Synonym(KC-135, s) ∧KNOW-ABOUT(H, s)

EFFECTS         ∀x ∈ synonyms(KC-135)
                     KNOW(H, Synonym(KC-135, x))

DECOMPOSITION   Inform(S, H, Synonymic-Definition(KC-135))
```

Figure 4.11 Instantiated `define-by-synonymic-definition` Plan Operator

The resulting text plan, similar to the logical definition description provided above, is shown in Figure 4.12. The linguistic realization component linearizes this tree and produces the utterance:

`A KC-135 is a stratotanker.`

where the synonym "stratotanker" was found in the "codename" attribute of the object in the knowledge base.

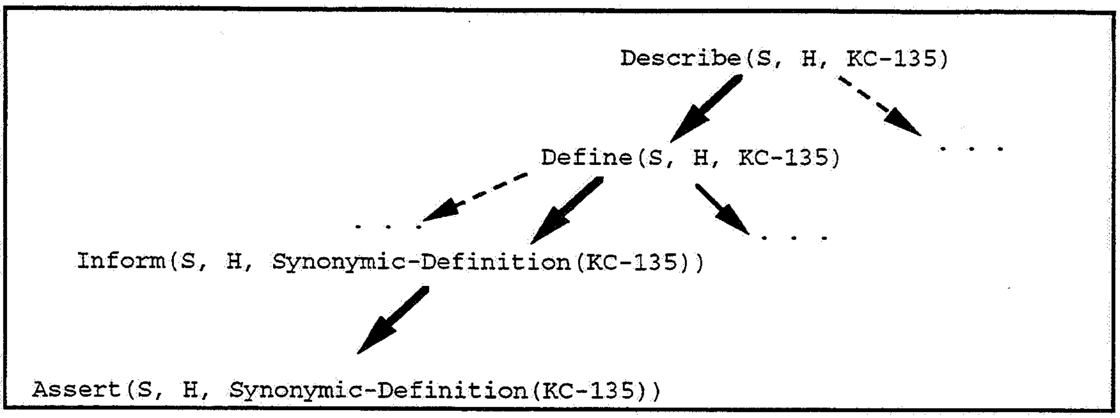



Figure 4.12 Synonymic Definition Text Plan

Unlike logical or synonymic definition, *antonymic definition* tells the reader what an object is not. The `define-by-antonymic-definition` plan operator, shown in Figure 4.13, requires that an entity have antonyms and prefers that the hearer is familiar with them. Since there are no antonyms of `KC-135` in the knowledge base, the constraints of the plan operator fail and it is not chosen.

```
NAME            define-by-antonymic-definition
HEADER          Define(S, H, entity)

CONSTRAINTS     ∃x Antonym(entity, x)
PRECONDITIONS
  ESSENTIAL     ∃s Antonym(entity, s) ∧ KNOW(S, Antonym(entity, s))
  DESIRABLE     ¬ KNOW-ABOUT(H, entity) ∧
                ∃x Antonym(entity, x) ∧ KNOW-ABOUT(H, x)

EFFECTS         ∀x ∈ antonyms(entity)
                     KNOW(H, Antonym(entity, x))

DECOMPOSITION   Inform(S, H, Antonymic-Definition(entity))
```

Figure 4.13 Uninstantiated `define-by-antonymic-definition` Plan Operator

### 4.5.3 Replanning a Definition

As the above generated definitions illustrate, in contrast to preplanned approaches to generation the planner can dynamically plan a text, selecting the appropriate communicative act (formalized as plan operators) for the current context and attempting alternate communicative actions when the constraints or preconditions of plan operators fail. In addition to this dynamic planning, the planner can recover if its planned strategy fails upon execution. That is, assume the `define-by-logical-definition` plan operator in Figure 4.9 is executed (i.e., linguistically realized or uttered) and fails (i.e., the hearer rejects it). At this stage the discourse controller passes the hierarchical text plan of Figure 4.9 to a replanner which attempts the next item in the ordered list of alternatives in the text plan, indicated as "untried alternatives" in Figure 4.9. This corresponds to the following interaction where user input (U) is italicized and system output (S) is in typewriter font. As TEXPLAN performs no query interpretation, the user's request is simulated by posting the corresponding goal, `KNOW-ABOUT(H, KC-135)`, to the text planner. Subsequent user interactions (e.g., "What?" and "ok") are simulated by a simple command language.

U1: *What is a KC-135?*
S1: `A KC-135 is a tanker for air-refueling and a transport vehicle for cargo transport.`
U2: *What?*
S2: `A KC-135 is a stratotanker.`
U3: *ok.*

So after U3 TEXPLAN assumes it has achieved the effect of S2 (i.e., `KNOW-ABOUT(H, KC-135)` and `KNOW(H, Synonym(KC-135, STRATOTANKER))`). Thus TEXPLAN builds a *user model* of the effects it believes it has achieved on the user's knowledge, beliefs, and desires. This allows the system to tailor its future responses by referring to this user model. As subsequent chapters discuss, it is necessary to model the cognitive (e.g., knowledge, beliefs, desires), physical (e.g., ability) and emotional (e.g., sad, frightened) state of the user to tailor different types of text (e.g., descriptions versus arguments). As user modelling is not the principal aim of this thesis, however, this user model is a placeholder for more sophisticated account (cf. Allen, 1988).

We have seen how the communicative acts of logical, synonymic, and antonymic definition can be planned and executed. The system can also replan a failed plan by executing alternative communicative acts recorded in the text plan, and by explicitly recording the effects associated with each executed action, it can gradually build a model of its expected effect on the knowledge, beliefs, and desires of the user. Now having characterized the three principal techniques of definition, the next section formalizes ways of detailing entities.

## 4.6 Details: Attribution, Purpose, and Illustration

Definition succinctly describes an entity. Oftentimes, however, it is unnecessary, difficult, or impossible to define something and so writers instead provide details. For instance a writer can make the reader know about an entity by detailing its characteristics. In a knowledge based system these include the attributes and values of an entity. In TEXPLAN, the plan operator that *details* the properties of an entity is shown in Figure 4.14. In *attribution*, distinguishing or salient characteristics (i.e., differentia) are used when an abundance of entity attributes and values exist. The "differentia formula" defined earlier is exploited in the `Attribution` rhetorical predicate in the decomposition. Following Levine (forthcoming), a `WHERE` field is used within plan operators to provide local variable definition. In Figure 4.14 the local variable *attributes* is defined as ("=") the set of attributes of the entity that the speaker knows (where "{}" is used to indicate a set and "|" means 'such that'). In the implementation this is achieved using a special operator, **set-var,** to bind local variables. This is similar to Moore's (1989) use of "setq" in plan operators except that in TEXPLAN the bound local variables are used not only in the decomposition of plan operators but also in the constraints, preconditions, and effects. During planning, local variables are instantiated in step 3 of Figure 4.2, just after the header of the plan operator is instantiated.

In addition to attribution, another way to provide details is to describe the purpose(s), function(s), or use(s) of an entity. For example, we can say that "A bicycle transports people" indicating the primary use or purpose of a bike. This technique is captured by the `describe-by-indicating-purpose` operator shown in Figure 4.15 where the `Purpose` rhetorical predicate retrieves from the knowledge base the function(s) or use(s) of an entity.

```
NAME            describe-by-attribution
HEADER          Describe(S, H, entity)

CONSTRAINTS     ∃a Attribute(entity, a) ∧(HASTE(S) ∨ HASTE(H))
PRECONDITIONS
  ESSENTIAL     WANT(S, KNOW-ABOUT(H, entity))
  DESIRABLE     ¬KNOW-ABOUT(H, entity) ∧
                ∃a ∈ attributes
                     ¬KNOW(H, Attribute(entity, a))

EFFECTS         KNOW-ABOUT(H, entity) ∧
                ∀a ∈ attributes
                     KNOW(H, Attribute(entity, a))

DECOMPOSITION   Inform(S, H, Attribution(entity, attributes))

WHERE           attributes = {a | Attribute(entity, a) ∧
                                  KNOW(S, Attribute(entity, a)) }
```

Figure 4.14 Uninstantiated `describe-by-attribution` Plan Operator

Another approach to short description is exemplification. This technique concretely describes an entity by furnishing a particular illustration of it. For example, consider the following passage from *Lessons in Physical Geography* (Dryer from Brooks and Hubbard, 1905):

> The lower portions of stream valleys which have sunk below sea level are called *drowned valleys*. The lower St. Lawrence is perhaps the greatest example of a drowned valley in the world, but many other rivers are in the same condition. The old channel of the Hudson River may be traced upon the sea bottom about 125 miles beyond its present mouth, and its valley is drowned as far up as Troy, 150 miles. The sea extends up the Delaware River to Trenton, and Chesapeake Bay with its many arms is the drowned valleys of the Susquehanna and its former tributaries. Many of the most famous harbors in the world, as

```
NAME            describe-by-indicating-purpose
HEADER          Describe(S, H, entity)

CONSTRAINTS     ∃p Purpose(entity, p) ∧(HASTE(S) ∨ HASTE(H))
PRECONDITIONS
  ESSENTIAL     WANT(S, KNOW-ABOUT(H, entity))
  DESIRABLE     ¬ KNOW-ABOUT(H, entity) ∧
                ∃p ∈ purposes
                     ¬ KNOW(H, Purpose(entity, p))

EFFECTS         KNOW-ABOUT(H, entity) ∧
                ∀p∈ purposes
                     KNOW(H, Purpose(entity, p))

DECOMPOSITION   ∀p ∈ purposes
                      Inform(S, H, Purpose(entity, p))

WHERE           purposes = {p | Purpose(entity, p) ∧
                                KNOW(S, Purpose(entity, p)) }
```

Figure 4.15 `describe-by-indicating-purpose` Plan Operator

> San Francisco Bay, Puget Sound, the estuaries of the Thames and the Mersey, and the Scottish firths, are drowned valleys.

Exemplification is captured in the `describe-by-illustration` plan operator shown in Figure 4.16. In TEXPLAN, an entity is considered an `Illustration` of another entity if it is either an instance of it or a subtype of it. In the WHERE portion of the plan operator, the variable *examples* is bound to those that the speaker knows. Exemplification enables TEXPLAN to produce text like that shown below from the vertebrate domain:

U1: *What's an invertebrate?*
S1: `Crustaceans, for example, are invertebrates such as lobsters, crabs, shrimp, and barnacles. Arachnids are invertebrates such as spiders, scorpions, and ticks. Myriapods are invertebrates such as centipedes and millipedes.`

As in the previous definition dialogue, the user query (U1) is italicized because no interpretation is performed: it is simulated by posting the corresponding goal, KNOW-ABOUT(H, INVERTEBRATE), to the planner. This example shows how illustration relates entities (preferably one of which the user KNOWS-ABOUT as indicated by the user model) to the more general concept of invertebrate. If the user does not react negatively (e.g., "what?"), or if the user explicitly acknowledges the text (e.g., "ok."), then TEXPLAN assumes it has achieved the effects of the above plan operator (recorded in the text plan) and so the discourse controller updates the user model to include the effects of the above plan operator. As a result,

```
NAME           describe-by-illustration
HEADER         Describe(S, H, entity)

CONSTRAINTS    ∃e Illustration(entity, e) ∧(HASTE(S) ∨HASTE(H))
PRECONDITIONS
  ESSENTIAL    WANT(S, KNOW-ABOUT(H, entity))
  DESIRABLE    ¬ KNOW-ABOUT(H, entity) ∧
               ∃e ∈ examples  KNOW-ABOUT(H, e)

EFFECTS        KNOW-ABOUT(H, entity) ∧
               ∀e ∈ examples
                    KNOW-ABOUT(H, e) ∧
                    KNOW(H, Illustration(entity, e))

DECOMPOSITION  ∀e ∈ examples
                     Inform(S, H, Illustration(entity, e))

WHERE          examples = { e | Illustration(entity, e) ∧
                                  KNOW(S, Illustration(entity, e)) }
```

Figure 4.16 describe-by-illustration Plan Operator

the user model now indicates that the hearer knows about the class invertebrate (i.e., KNOW-ABOUT(H, INVERTEBRATE)) as well as about any examples of it (e.g., KNOW-ABOUT(H, CRUSTACEANS), KNOW-ABOUT(H, Illustration(CRUSTACEANS, INVERTEBRATE)), and so on). This information can then be used to guide plan operator selection when producing future text.

## 4.7 Division: Classification and Constituency

A very common technique that allows the writer to rapidly give the hearer a picture of an entity is division. *Division* involves separating an entity into classes or parts. If an entity is described using its subclasses or subtypes this is termed *classification*. If it is divided into subparts this is termed *constituency*. The describe-by-classification plan operator shown in Figure 4.17 requires that the entity have subtypes or subclasses, and prefers that the hearer is familiar with at least one of these. Since this technique is more informative than exemplification, it is listed before it in the plan library. A constraint on using this plan operator is haste on the part of the speaker or hearer which distinguishes it from the extended description discussed in the next section, which divides the entity into subclasses or subparts and discusses each of these in turn. The dialogue below is from the Map Display System (Hilton, 1987).

U1: *What's a town?*
S1: A town is a carto-object with roads, a size, and a perimeter.
U2: *What's a carto-object?*
S2: There are 11 carto-objects: a town, an obstruction, a lake, an intersection, a heliport, a dam, a bridge, a block, an airstrip, a line segment object, and a segmented object.

```
NAME            describe-by-classification
HEADER          Describe(S, H, entity)

CONSTRAINTS     ∃x Subtype(entity, x) ∧(HASTE(S) ∨ HASTE(H))
PRECONDITIONS
  ESSENTIAL     WANT(S, KNOW-ABOUT(H, entity)) ∧
                ∀s ∈ subtypes(entity)
                      KNOW(S, Subtype(entity,s))
  DESIRABLE     ¬ KNOW-ABOUT(H, entity) ∧
                ∃x Subtype(entity, x) ∧KNOW-ABOUT(H, x)

EFFECTS         KNOW-ABOUT(H, entity) ∧
                ∀s ∈ subtypes(entity)
                      KNOW(H, Subtype(entity,s))

DECOMPOSITION   Inform(S, H, Classification(entity))
```

Figure 4.17 `describe-by-classification` Plan Operator

U3: *What's a segmented object?*
S3: `There are 5 segmented objects: a waterway, a road, a railroad, a powerline, and a border.`

As a result of the logical definition (S1), the hearer knows about the concept town, a subordinate of carto-object. Therefore TEXPLAN can select the describe-by-classification plan operator in response to U2. And if the user model indicates that the hearer knows about one of the segmented objects, the system responds to U3 with S3.

The cousin to classification, constituency (also called partition) discusses the subparts of an entity rather than its subtypes. For example, just as a writer can enumerate the different types of telephones (e.g., car phones, pay phones, cordless phones), s/he can also detail the subparts of it: the transmitter, receiver, dialing mechanism, housing, and cord. The `describe-by-constituency` plan operator is shown in Figure 4.18.

Using a small entity, event, and process knowledge base, TEXPLAN produces the responses shown below to indicate the constituents of a recipe (composed of ingredients), an event (composed of stages) and a process (composed of events, in this case transformations):

U1: *What is chili con carne?*
S1: `Chili con carne consists of meat, beans, onions, tomatoes, and chili powder.`

```
NAME           describe-by-constituency
HEADER         Describe(S, H, entity)

CONSTRAINTS    ∃x Subpart(entity, x) ∧(HASTE(S) ∨ HASTE(H))
PRECONDITIONS
  ESSENTIAL    WANT(S, KNOW-ABOUT(H, entity)) ∧
               ∀s ∈ subparts(entity)
                    KNOW(S, Subpart(entity,s))
  DESIRABLE    ¬ KNOW-ABOUT(H, entity) ∧
               ∃x Subpart(entity, x) ∧KNOW-ABOUT(H, x)

EFFECTS        KNOW-ABOUT(H, entity) ∧
               ∀s ∈ subparts(entity)
                    KNOW(H, Subpart(entity,s))

DECOMPOSITION  Inform(S, H, Constituency(entity))
```

Figure 4.18 `describe-by-constituency` Plan Operator

U2: *What is chili powder?*
S2: `Chili powder consists of dried chilies, oregano, cumin, and garlic.`

U1: *What is a day?*
S1: `A day consists of morning, afternoon, and night.`

U1: *What is metamorphosis?*
S1: `Metamorphosis consists of a transformation from egg to larva, a transformation from larva to pupa, and a transformation from pupa to adult.`

While other responses might be more effective (e.g., logical definitions), these examples are intended simply to illustrate constituency. In a description of this kind, the constituent parts can be ordered by size, importance, length, spatial relationship, and so on. This rhetorical technique is the core of McKeown's (1985) constituency schema, discussed in Chapter 2.

## 4.8 Putting things together: Extended Description

Oftentimes writers want or need to provide extended descriptions. Consider paragraph two from the foreword of the 1986 *Cambridge University Varsity Handbook*:

> The Varsity Handbook is different. It does not attempt to present a unified and neatly packaged version of the 'real' Cambridge. It is written and produced entirely by students and reflects a range of opinions. The 'University' section is an assortment of articles by students on aspects of University life. The 'Time Out' section is intended to suggest ideas about how to spend your spare time in and around Cambridge and includes an extensive restaurant and pub guide. The 'Information' section is a useful file of the many services and facilities available in the area.

The text first introduces the handbook by indicating some of its attributes, including what is not its purpose. The passage then describes each of the book's constituent parts. First the "university" section is characterized. Next the "time out" section's purpose and components (restaurant and pub guide) are indicated. Finally, the "information" chapter is described.

This and other texts like it suggest the `extended-description` plan operator shown in Figure 4.19. The boldfaced special operator, **optional**, relaxes the mandatory default in the decomposition so that parts of the decomposition are provided only when available. Like the Varsity Handbook text, this plan operator gets the hearer to know about an entity by defining it, detailing it, dividing it into its parts or classes and then describing them, and providing an example or analogy of the entity. Since definition was previously formalized, the following discussion defines subordinate plan operators for detail, division, exemplification, and analogy. The detail, division, and exemplification plan operators are related to those defined previously for terse descriptions, however, the ones here are imbedded in an extended description and therefore do not have the same constraints regarding haste on the part of the speaker or hearer.

```
NAME              extended-description
HEADER            Describe(S, H, entity)

CONSTRAINTS       Entity?(entity)
PRECONDITIONS
  ESSENTIAL       KNOW-ABOUT(S, entity) ∧
                  WANT(S, KNOW-ABOUT(H, entity))
  DESIRABLE       ¬ KNOW-ABOUT(H, entity)

EFFECTS           KNOW-ABOUT(H, entity)

DECOMPOSITION     Define(S, H, Entity)
                  optional(Detail(S, H, entity))
                  optional(Divide(S, H, entity))
                  optional(Illustrate(S, H, entity)) ∨
                            Give-Analogy(S, H, entity))
```

Figure 4.19 `extended-description` Plan Operator

One method of characterizing an entity is to *detail its attributes*. When the speaker conveys the attributes of an entity, the result is that the hearer knows about those properties. The natural constraints on this rhetorical act are that the entity must have attributes and that the speaker must know about at least one of them. This act is formalized as `detail-by-attribution` plan operator shown in Figure 4.20.

```
NAME            detail-by-attribution
HEADER          Detail(S, H, entity)

CONSTRAINTS     ∃a Attribute(entity, a)
PRECONDITIONS
  DESIRABLE     ∃a ∈ attributes
                     ¬ KNOW(H, Attribute(entity, a))

EFFECTS         ∀a ∈ attributes
                     KNOW(H, Attribute(entity, a))

DECOMPOSITION   Inform(S, H, Attribution(entity, attributes))

WHERE           attributes = { a | Attribute(entity, a) ∧
                                 KNOW(S, Attribute(entity, a)) }
```

Figure 4.20 `detail-by-attribution` Plan Operator

Similarly, when the speaker conveys the *purpose* of an entity, the result is that the hearer knows the purpose(s), function(s), or use(s) of the entity. The constraints on detailing the purpose of an entity are that it must have a purpose represented in the knowledge base and that the speaker must know about it. The `detail-by-indicating-purpose` plan operator capturing this is shown in Figure 4.21.

In an extended description, after an entity has been introduced, by defining it and (optionally) detailing it, it can be decomposed into its subparts or subtypes. This *division* is achieved by the rhetorical techniques of classification or constituency. To be as informative as possible, if there are more subclasses (subtypes) than components (subparts) then classification is chosen. As shown in Figure 4.22 , the

```
NAME            detail-by-indicating-purpose
HEADER          Detail(S, H, entity)

CONSTRAINTS     ∃p Purpose(entity, p)
PRECONDITIONS
  DESIRABLE     ∃p ∈ purposes
                     ¬ KNOW(H, Purpose(entity, p))

EFFECTS         ∀p ∈ purposes
                     KNOW(H, Purpose(entity, p))

DECOMPOSITION   ∀p ∈ purposes
                     Inform(S, H, Purpose(entity, p))

WHERE           purposes = { p | Purpose(entity, p) ∧
                               KNOW(S, Purpose(entity, p)) }
```

Figure 4.21 `detail-by-indicating-purpose` Plan Operator `divide-by-classification` plan operator informs the hearer of the subclasses, types, or groups of an entity and then optionally describes each of them in turn.

```
NAME             divide-by-classification
HEADER           Divide(S, H, entity)

CONSTRAINTS      ∃s Subtype(entity, s) ∧
                 length(subtypes(entity)) > length(subparts(entity))
PRECONDITIONS
  ESSENTIAL      ∀s ∈ subtypes(entity)
                      KNOW(S, Subtype(entity,s))
  DESIRABLE      ∃x Subtype(entity, x) ∧ KNOW-ABOUT(H, x)

EFFECTS          ∀s ∈ subtypes(entity)
                      KNOW(H, Subtype(entity,s))

DECOMPOSITION    Inform(S, H, Classification(entity))
                 ∀x ∈ subtypes(entity)
                      optional(Describe(S, H, x))
```

Figure 4.22 `divide-by-classification` Plan Operator

In contrast to this categorization technique, the `divide-by-constituency` plan operator shown in Figure 4.23 informs the hearer of the parts, segments, or elements of an entity and then optionally describes each component in turn. In both classification and constituency the special operator **optional** allows for variability.

```
NAME             divide-by-constituency
HEADER           Divide(S, H, entity)

CONSTRAINTS      ∃s Subpart(entity, s) ∧
                 length(subparts(entity)) > length(subtypes(entity))
PRECONDITIONS
  ESSENTIAL      ∀s ∈ subparts(entity)
                      KNOW(S, Subpart(entity,s))
  DESIRABLE      ∃s Subpart(entity, s) ∧ KNOW-ABOUT(H, s)

EFFECTS          ∀s ∈ subparts(entity)
                      KNOW(H, Subpart(entity,s))

DECOMPOSITION    Inform(S, H, Constituency(entity))
                 ∀s ∈ subparts(entity)
                      optional(Detail(S, H, s))
```

Figure 4.23 `divide-by-constituency` Plan Operator

While classification and constituency give the text generator structure with which to convey information associated with the underlying application, the available knowledge sometimes warrants using both forms of division. This dual division is illustrated by the following excerpt from the World Book Encyclopedia (1986):

> MICROSCOPE is an instrument that magnifies extremely small objects so they can be seen easily. It produces an image much larger than the original object. There are three basic kinds of microscopes: (1) optical, or light; (2) electronic; and (3) ion. The optical microscopes used in most schools and colleges for teaching have three parts: (1) the foot, (2) the tube, and (3) the body. The foot is the base on which the instrument stands. The tube contains the lenses, and the body is the upright support that holds the tube.

In this text the microscope is initially identified by its genus, purpose, and function. The passage then discusses the major classes of microscopes (classification) and partitions a microscope into components (constituency) and describes these components -- foot, tube, and body -- in turn. This is done by giving their function with respect to the whole. Both types of division, classification and constituency, thus complement each other to ensure a well-structured text. This strategy is captured in the `divide-by-classification-and-constituency` plan operator shown in Figure 4.24.

```
NAME             divide-by-classification-and-constituency
HEADER           Divide(S, H, entity)

CONSTRAINTS      ∃s Subpart(entity, s) ∧ ∃s Subtype(entity, s) ∧
                 length(subparts(entity)) = length(subtypes(entity)) ∧
                 Positive(length(subparts(entity)))
PRECONDITIONS
  ESSENTIAL      ∀s ∈ subparts(entity)
                      KNOW(S, Subpart(entity,s)) ∧
                 ∀s ∈ subtypes(entity)
                      KNOW(S, Subtype(entity,s))
  DESIRABLE      ∃x Subpart(entity, x) ∧ KNOW-ABOUT(H, x) ∧
                 ∃x Subtype(entity, x) ∧ KNOW-ABOUT(H, x)

EFFECTS          ∀s ∈ subparts(entity)
                       KNOW(H, Subpart(entity,s)) ∧
                 ∀s ∈ subtypes(entity)
                       KNOW(H, Subtype(entity,s))

DECOMPOSITION    Inform(S, H, Classification(entity))
                  ∀x ∈ subtype(entity, x)
                       optional(Describe(S, H, x))
                 Inform(S, H, Constituency(entity))
                  ∀x ∈ subparts(entity, x)
                      optional(Detail(S, H, x))
```

Figure 4.24 `divide-by-classification-and-constituency` Plan Operator

The last part of an extended description provides an illustration or analogy. The `illustrate` plan operator in Figure 4.25 defines exemplification. Providing that there are examples of the entity, and the speaker knows one, then the speaker informs the hearer of it and, optionally, describes it. Here the variable *`example`* is bound to a particular example of the entity so that is can be used consistently throughout the plan operator.

```
NAME              illustrate
HEADER            Illustrate(S, H, entity)

CONSTRAINTS       ∃e Illustration(entity, e)
PRECONDITIONS
  DESIRABLE       ¬ KNOW(H, Illustration(entity, example))

EFFECTS           KNOW(H, Illustration(entity, example))

DECOMPOSITION     Inform(S, H, Illustration(entity, example))
                  optional(Describe(S, H, example))

WHERE             example = e | Illustration(entity, e) ∧
                                KNOW-ABOUT(H, e) ∧
                                KNOW(S, Illustration(entity, e))
```

Figure 4.25 `illustrate` Plan Operator

Finally, Figure 4.26 shows TEXPLAN's plan operator for *analogy*. The constraint on the plan operator means it can only be exploited where an entity has an analogy (as defined in section 4.10 below). The essential preconditions first bind the variable *`analogue`* to an entity that the hearer knows which is analogous to the one the hearer is unfamiliar with. The speaker must know this analogy (or at least be able

```
NAME              give-analogy
HEADER            Give-Analogy(S, H, entity)

CONSTRAINTS       ∃x Analogous(entity, x)
PRECONDITIONS
  DESIRABLE       ∃x Analogous(entity, x) ∧ KNOW-ABOUT(H, x)

EFFECTS           KNOW(H, Analogous(entity, analogue))

DECOMPOSITION     Inform(S, H, Analogy(entity, analogue))

WHERE             analogue = x | Analogous(entity, x) ∧
                                 KNOW-ABOUT(H, x) ∧
                                 KNOW(S, Analogous(entity, x))
```

Figure 4.26 `give-analogy` Plan Operator

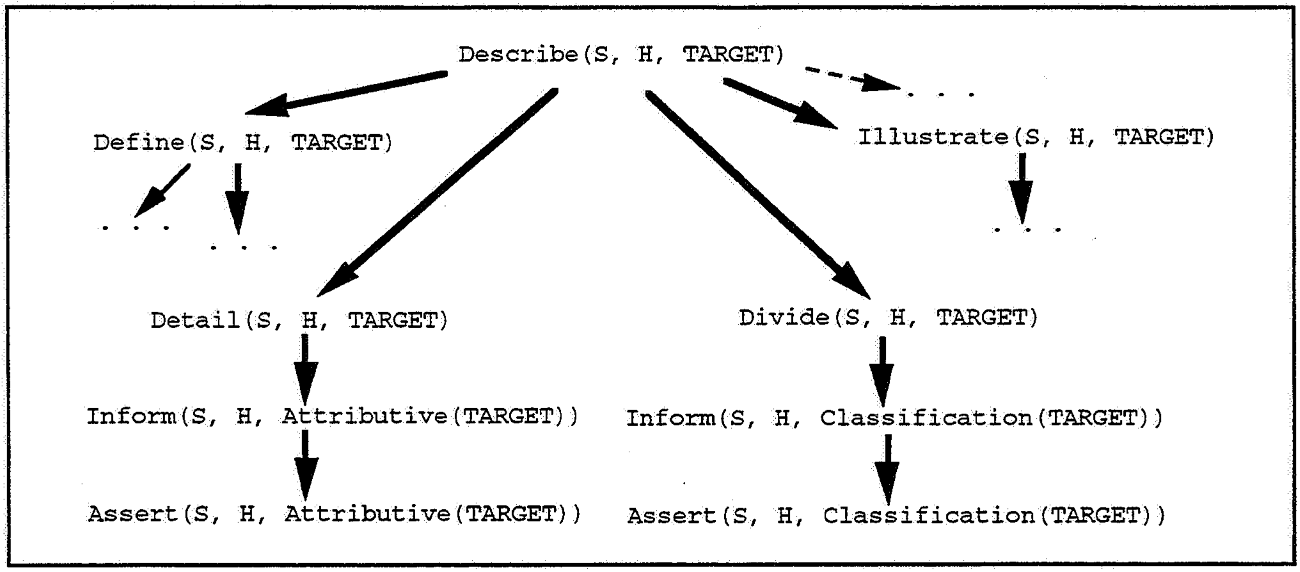



Figure 4.27 Hierarchical Text Plan for Extended Definition of a TARGET

to formulate it). As with previous plan operators, analogy can be very effective in discourse because it attempts to make contact with the hearer's knowledge.

We have now characterized all the constituents in the decomposition of the extended definition plan operator. To illustrate its use, consider what happens when the discourse goal KNOW-ABOUT(H, TARGET) is posted to the planner in a hybrid rule/frame based system for resource allocation (Dawson et al, 1987). If the assumed agent model indicates that neither the speaker nor hearer are rushed, then the extended definition plan operator is selected. Assuming that the agent model indicates that the user knows about weapons and the speaker knows that they are an example of a target, then the text plan shown in Figure 4.27 is produced. This corresponds to the text :

```
Targets are entities. They have a latitude/longitude, a cloud cover, a
cloud height, a visibility, and a weather condition. There are five
targets: passages, facilities, electronic hardware, weapons, and
vehicles. Weapons, for example, are targets such as anti-aircraft
missiles, surface-to-surface missiles, and enemy aircraft.
```

The structure of the text plan in Figure 4.27 is conveyed both by the content of the different parts of the text as well as by explicit cue words (e.g., "for example" in the final utterance). Cohesion is aided both by cue words as well as by tracking focus and using it to guide pronominalization (e.g., the use of "they" in the second utterance) and grammatical structure (e.g., there-insertion in the third utterance). (The relevant linguistic realization details are given in Chapter 8.) While the above text is a reasonable response to its user input, improvements can be made both in terms of content and presentation. First, the initial utterance is confusing because the concept of "target" is defined in terms of the vague concept, "entity". It would be useful to distinguish between the concept of "target" in general and its particular usage in the context of the domain application. Second, while the current implementation randomly selects examples, choice could be based on a model of what is known or not known by the user. One could argue that weapons is a sufficiently well-know concept and so does not need to be exemplified. Instead, passages, facilities, and electronic hardware could be detailed. Also, only a few of the most prototypical and best-known examples could be chosen (see Appendix A for a discussion of prototypicality). Finally, the presentation could be enhanced, for example, by exemplifying concepts parenthetically. The next section on comparison shows how parenthetical phrasing can be effective with the realization of the plan operator in Figure 4.29.

## 4.9 Comparison

Comparison focuses on the similarities and differences of two entities. Like extended definition, comparison is a complex text genre. Like division, it can be used independently or as part of another genre. Comparison is sometimes used to respond to an explicit query. Consider the following passage concerning the difference between an alligator and a crocodile (Pickett and Laster, 1988):

> *Alligator or Crocodile: What's the difference?*
>
> The alligator is a close relative of the crocodile. The alligator, however, has a broader head and blunter snout. Alligators are usually found in fresh water; crocodiles prefer salt water. The alligator's lower teeth, which fit inside the edge of the upper jaw, are not visible when the lipless mouth is closed. The crocodile's teeth are always visible.

This paragraph is organized point by point. First head and snouts are contrasted; then natural habitat; and, finally, the visibility of their teeth.

This type of comparison is reflected in the `compare-point-by-point` plan operator in Figure 4.28 which first states the resemblance of two entities and then supports this by comparing and contrasting them characteristic by characteristic. The `comparison_contrast` rhetorical predicate is based on Tversky's (1977) similarity metric as discussed and refined in Appendix A. As detailed there, the formula compares and contrasts the attributes of two entities and then, for all attributes common to them, it compares and contrasts their attribute-value pairs. This formula gives a measure of entity similarity on the range [0 1] which can be used in the `inference` rhetorical predicate at the end of the compare plan operator. Furthermore, the formula can be decomposed into the parts that identify common attributes and common attribute value pairs as well as the parts that contrast attributes and attribute value pairs. Note, however, that the comparison metric must also consider the equality of the superordinates of the two compared entities. This is necessary since differentia calculated for logical definition only considers entity attributes

```
NAME            compare-point-by-point
HEADER          Compare(entity1, entity2)

PRECONDITIONS
  DESIRABLE     KNOW-ABOUT(H, entity1) ∨ KNOW-ABOUT(H, entity2)

EFFECTS         KNOW-ABOUT(H, entity1) ∧ KNOW-ABOUT(H, entity2) ∧
                KNOW(H, Difference(entity1, entity2))

DECOMPOSITION
   optional(Inform(S, H, Inference(entity1, entity2)))
   ∀ attribute ∈ (differentia(entity1) ∧ differentia(entity2))
     Inform(S, H, Comparison-Contrast(entity1, entity2, attribute))
```

Figure 4.28 `compare-point-by-point` Plan Operator

and values, not their superordinates. The formula can also incorporate the refined notion of feature equality as defined in Appendix A to be more sensitive than binary equality tests.

The `compare-point-by-point` plan operator is used when the discourse goal `KNOW(H, Difference(FISH, BIRDS))` is posted to TEXPLAN in the domain of vertebrate classification. This results in the following English surface form:

```
Fish and birds are different entities.  Both fish and birds are
vertebrates.  However, fish swim whereas birds fly.  Fish have fins;
birds have wings. Fish are aquatic whereas birds are terrestrial. Fish
eat vegetation and fish whereas birds eat seeds. Fish have scales
whereas birds have feathers. Fish are cold-blooded; birds are warm-
blooded.
```

This arrangement analyzes `FISH` and `BIRDS` point by point, completing each point before going to the next.

In addition to this point by point organization, there are two other common approaches to comparison. The following text uses one method, first pointing out similarities, then differences.

```
Fish and birds are different entities.  Fish and birds have the same
superclass (vertebrates).  However, fish and birds have different
locomotion (swim versus fly), different propellors (fins versus wings),
different environments (aquatic versus terrestrial), different diets
(vegetation and fish versus seeds), different covering (scales versus
feathers), and different blood-temperatures (cold-blooded versus warm-
blooded).
```

The text addresses common features first; unique features last. This organization corresponds to the `compare-similarities/differences` plan operator shown in Figure 4.29. In part because there are many different attributes in the third utterance, I find this text less effective than the previous example.

```
NAME            compare-similarities/differences
HEADER          Compare(entity1, entity2)

PRECONDITIONS
  DESIRABLE     KNOW-ABOUT(H, entity1) ∨ KNOW-ABOUT(H, entity2)

EFFECTS         KNOW-ABOUT(H, entity1) ∧ KNOW-ABOUT(H, entity2) ∧
                KNOW(H, Difference(entity1, entity2))

DECOMPOSITION   optional(Inform(S, H, Inference(entity1, entity2)))
                Inform(S, H, Similarities(entity1, entity2))
                Inform(S, H, Differences(entity1, entity2))
```

Figure 4.29 `compare-similarities/differences` Plan Operator

Perhaps the most common form of comparison, however, first describes the two entities, then details their common and unique features, and optionally infers from this how close or far apart they are from each other. This corresponds to the `compare-describe-in-turn` plan operator shown in Figure 4.30. Figure 4.31 shows the topmost level of the hierarchical plan produced by TEXPLAN using this organization.

```
NAME            compare-describe-in-turn
HEADER          Compare(entity1, entity2)

PRECONDITIONS
  DESIRABLE     KNOW-ABOUT(H, entity1) ∨ KNOW-ABOUT(H, entity2)

EFFECTS         KNOW-ABOUT(H, entity1) ∧ KNOW-ABOUT(H, entity2) ∧
                KNOW(H, Difference(entity1, entity2))

DECOMPOSITION   Describe(S, H, entity1)
                Describe(S, H, entity2)
                Inform(S, H, Comparison-Contrast(entity1, entity2))
                optional(Inform(S, H, Inference(entity1, entity2)))
```

Figure 4.30 `compare-describe-in-turn` Plan Operator

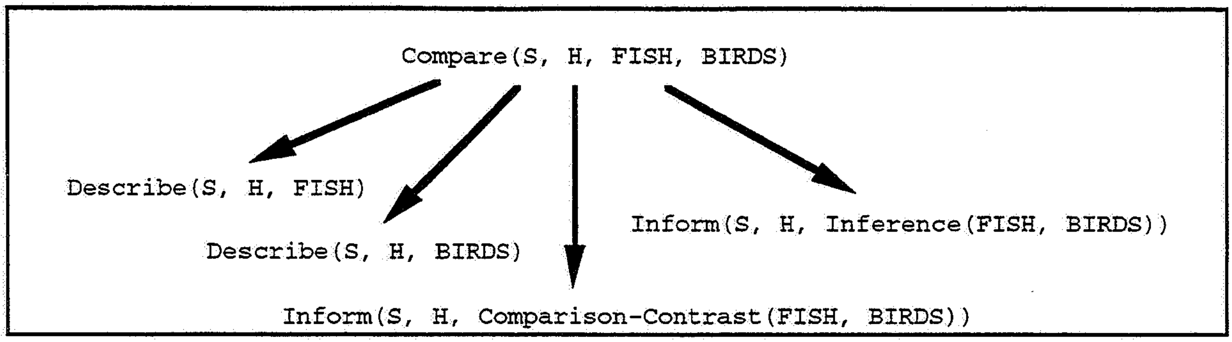



Figure 4.31 Top-Level of Hierarchical Text Plan for Comparison of `FISH` and `BIRD`

When linearized and realized onto English surface form, the text plan of Figure 4.31 yields:

```
Fish are vertebrates that swim, have fins, have gills, are aquatic, eat
vegetation and fish, have scales, and are cold-blooded.
Birds, on the other hand, are vertebrates that fly, have wings, are
terrestrial, eat seeds, have feathers, and are warm-blooded.
Fish and birds have the same superclass, different locomotion, different
propellors, different environments, different diets, different covering,
and different blood-temperatures.  Therefore, they are different entities.
```

While the three alternative arrangements of comparison have similar effects, the resulting surface form and emphasis is distinct for each.

## 4.10 Analogy

Just as comparison attempts to inform the hearer by making contact with familiar knowledge, figures of speech involve reference to known entities that are in some way, perhaps implicitly, related to the unknown entity. The most common figure of speech is *analogy* which compares two essentially different entities (e.g., a heart and a pump) that nevertheless have certain real or imagined similarities. To characterize a complex matter very broadly and crudely, there are two forms of analogy: simile and metaphor. A *simile* implies a comparison whereas a metaphor expresses it explicitly. That is, a simile asserts that one entity *is like* another whereas a metaphor says that one entity *is* another. Consider Robert Burns simile "My love is like a red, red rose." While women are not roses, both are delicate, fragrant, and beautiful. *Metaphor*, in contrast, *equates* two distinct entities as in "God is a mighty fortress."

TEXPLAN does not embody a very deep analysis of analogy, or attempt to capture the distinctions between simile or metaphor. Thus the `describe-by-analogy` plan operator shown in Figure 4.32 describes an entity using analogy if the hearer is familiar with an analogous entity, and for simplicity the `Analogy` rhetorical predicate is realized as a simile to make it explicit to the reader that analogy is being used.

```
NAME            describe-by-analogy
HEADER          Describe(S, H, entity)

CONSTRAINTS     Entity?(entity) ∧ ∃x Analogous(entity, x) ∧
                (HASTE(S) ∨ HASTE(H))
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, entity) ∧
                WANT(S, KNOW-ABOUT(H, entity)) ∧
                WANT(S, KNOW(H, Analogous(entity, analogue)))
  DESIRABLE     ¬ KNOW-ABOUT(H, entity)

EFFECTS         KNOW-ABOUT(H, entity) ∧
                KNOW(H, Analogous(entity, analogue))

DECOMPOSITION   Inform(S, H, Analogy(entity, analogue))

WHERE           analogue = x | Analogous(entity, x) ∧
                               KNOW-ABOUT(H, x) ∧
                               KNOW(S, Analogous(entity, x))
```

Figure 4.32 describe-by-analogy Plan Operator

As in the above comparison plan operators, the Analogous predicate in the plan operator uses formulas detailed in Appendix A to select and produce analogies. For example, to select an analogy, the feature-based entity similarity metric described in Appendix A measures the degree of similarity between the unknown entity and entities that the user knows about (i.e., in the user model). Given two entities, this function compares their attributes and attribute-value pairs and returns a value on the range [0, 1] indicating their similarity. A value above some predefined threshold indicates that two entities are analogous. Using this similarity measure, analogies are preferentially ordered. The search for analogous entities can be pruned by exploiting the structure of the knowledge base. For example, analogy selection can consider only entities in the user model that are "close" to the unknown entity where closeness is measured by the type, number, and length of links between the two entities in the knowledge base. Once an analogous entity has been selected, the content of the Analogy rhetorical predicate is produced by distinguishing the attributes and attribute-value pairs of the unknown entity from those of the known entity (see S4 in dialogue below). Again, this uses the differentia selection algorithm detailed in Appendix A.

The following dialogue illustrates the analogy plan operator in action. As before, user queries are simulated by posting the corresponding goal to the planner (e.g., KNOW-ABOUT(H, F-15) in U1). Similarly, feedback from the user is simulated by allowing the user to choose from a "reaction menu" with canned options such as 'accept' (i.e., "OK"), 'reject' (e.g., "huh?"), and "elaborate" ("Tell me more."). After the system defines an F-15 (S1), the user knows about it. Therefore, when the extended definition (S2) and synonymic definition (S3) of a MiG-29 fail, the system refers to the F-15 in describing it.

U1: *What's an F-15?*
S1: An F-15 is an American fighter for air-to-air-interdiction. It has

```
    several standard configuration loads (scl-d1 and scl-d2), a speed of
    350 miles per hour, a communications radius of 2,220,000, an empty
    weight of 12,500, a maximum weight of 24,500, a maximum fuel
    capacity of 10,600, a turn time of 3,600, a ferrying range of
    4,630,000, and a ferry package of 3,000.  There are five classes of
    F-15s:  the F-15A, the F-15B, the F-15C, the F-15D, and the F-15E.
```

U2: *OK. What's a MiG-29?*

S2:
```
A MiG-29 is a Soviet fighter for air-to-air-interdiction.
It has a speed of 600 miles per hour, and a communications radius of
1,200,000.
```

U3: *Huh?*

S3: `A MiG-29 is a Fulcrum.`

U4: *Huh?*

S4:
```
A MiG-29 is like an F-15 except that it is a Soviet fighter,
has a faster speed, and has a smaller communications radius.
```

U5: *OK.*

## 4.11 Further Work: Other Figures of Speech

In addition to analogy, there are several other special figures of speech that should be noted. These are not formalized, however, as they are used primarily for literary effect. (Nevertheless, the developed plan representation language should be suitable for at least part of their characterization). *Personification* is a special form of metaphor in which human qualities are attributed to inanimate entities (e.g., objects, animals, abstract ideas). *Apostrophe* is like personification but it pertains to the direct address of inanimate objects or the absent as if present as in Tennyson's:

Break, break, break
At the foot of thy crags, O Sea!

There are also several techniques whose effect it is to emphasize the content of an utterance. *Hyperbole* is an exaggerated expression used to increase the effectiveness of a statement as in "He has an iron fist." Similarly, *irony* emphasizes a point by saying just the opposite of what is intended. Finally, a *rhetorical question* (also called interrogation) is used not as a request but as an emphatic statement. An affirmative question denies ("Am I my brother's keeper?") and a negative question affirms ("Am I not free?").

Two other figures of speech involve substitution. *Metonymy* consists of substituting one entity for another, closely associated entity as in "The class is reading Shakespeare." Similarly, *synecdoche* consists of substituting a part of something for the whole or a whole for the part. Consider: "Two moons ago the eagle soared." or "Each household answered the survey."

There are also several rhetorical arrangement techniques that achieve certain effects on the hearer and can operate over longer stretches of text (i.e., at the clausal, sentential, or paragraph level). *Antithesis* involves contrasting statements as in (Brooks and Hubbard, 1905):

Look like the innocent flower,
But be the serpent under it.

-- Shakespeare

Unlike the contrastive order of antithesis, *climax* entails the ascendant arrangement of propositions. For example, "I came, I saw, I conquered." Each of these phenomena is beyond the scope of this thesis and the characterization of their specific constraints, preconditions, effects, and decomposition remains an exciting area for future research.

## 4.12 Conclusion

This chapter first defines text and then classifies the form and function of the four major genre of text: description, narration, exposition, and argument. It introduces a plan language that is used throughout the thesis to formalize communicative acts (rhetorical acts, speech acts, and surface speech acts). This is then used to define the principal techniques of description including definition, detail, division, extended description, comparison, and analogy. Each communicative act is first identified using naturally-occurring data, then formalized as a plan operator, and finally illustrated with implemented examples from TEXPLAN. These examples are only selected illustrations as literally hundreds of descriptive texts have actually been generated from multiple applications. In closing, figures of speech that were not formalized are briefly catalogued to provide a direction for future research. The chapters that follow detail how TEXPLAN composes the three remaining main types of text: narration, exposition, and argument.

# Chapter 5

# NARRATION

Show me a hero and I'll write you a tragedy.

*The Crackup*, 1936 Francis Scott Fitzgerald, 1896-1940

## 5.1 Introduction

This chapter begins by contrasting description with *narration*, a type of text which conveys sequences of events and states. Having defined narration, the chapter critiques previous systems that automatically generate narrative text. I then argue that what is needed is a formal ontology of events and states which, together with temporal knowledge and the notion of temporal focus (Webber, 1988), can be used to realize events in a more principled manner. In particular, I detail how TEXPLAN uses temporal information to select tense and aspect and to generate temporal adverbials. The remainder of the chapter then focuses on three particular types of narrative text that organize events: *reports* (temporally or topically sequenced events), *stories* (causally sequenced events), and *biographies* (event sequences concerning one agent). The communicative acts underlying these three types of narrative text are formalized as plan operators. I detail how TEXPLAN uses these plan operators to select and organize events with examples from a knowledge based simulation system. The chapter concludes by discussing more advanced narrative techniques such as surprise, suspense, and mystery.

## 5.2 Narration Defined

Whereas the primary purpose of description is to paint a verbal picture of some entity, narration attempts to produce a verbal *motion* picture that conveys a collection of related events. Simply put, narration tells a story of what happened. Narration comes in the form of anecdotes, incidents, short stories, tales, letters, novels, drama, history, biography, newspaper articles, travel writing, and even comic strips. Usually events are narrated by an omniscient story-teller in the third person, past tense. But events can also be conveyed in the present tense as they unfold or in the first person by one of the characters involved in the action. While description typically follows a spatial organization, narration is guided by temporal and causal orderings.

Narration can include descriptive passages which inform the reader of the static background for the dynamic foreground events. This *setting* includes the time, place, characters, or circumstances of the story. Description in narration tells the reader the who, what, when, and where of the story. It thus acts as a kind of backdrop for the action that aids the reader in interpreting why or how events took place.

Perhaps the most difficult aspect of narration concerns selecting and ordering events. The most common organization of narrative is simply a chronological presentation of the most salient happenings, which I shall term a *report*. A *story*, in contrast, follows some underlying plot or causally connected series of events. A plot line may concern a basic human theme, issue, or problem such as courage, cowardice, honesty, dishonesty, compassion, fortitude, maturity, immaturity, or magnanimity. In more sophisticated stories, the plot line is multi-level reflecting the actions of multiple, autonomous agents or simply the complex nature of life. This is an important issue because while in real life actions occur in parallel, prose is linear. Therefore, the narrator must deal with simultaneous and overlapping events by, for example, using connectives such as "while" and "in the meantime". Furthermore, s/he must reason about event persistence (i.e., duration). In short, the narrator often uses both temporal as well as causal relationships to link events in a story.

In some types of narrative text, however, non-chronological and non-causal orderings are more appropriate. Histories and biography can be ordered temporally or causally, or revolve around significant ideas or accomplishments of an individual (e.g., education, literature, discoveries). It is usually some combination of these. Furthermore, a narrator may intentionally leave out events or present events out of temporal or causal sequence in order to achieve specific effects on the hearer. For example, flashback stimulates interest by jumping backward in time in order to stimulate interest in how the situation has developed to its current state. In contrast, to create suspense, events can be communicated in increasing order of importance leading to a decisive point in the plot: the climax. A plan-based approach to language generation has the advantage that it can capture the structure, order, and effect of different narrative techniques.

It should be noted that many researchers have investigated the representation and/or recognition of event sequences. For example, SCRIPTS (Schank, 1975) attempt to capture event sequences underlying stereotypical situations independent of their order of presentation in a text. SCRIPTS included information about settings as well as agent roles. However, the presentation of events represented in SCRIPTS (e.g., event summaries) was based on fairly fixed conceptual templates associated with each script. Lehnert (1981) also characterized a number of event and state configurations that she claimed were the basic "plot units" in narrative. In contrast to this focus on event sequence recognition/representation, the next section examines work that focuses on generating narrative text. The remainder of this chapter then details how TEXPLAN captures narrative organizational techniques in terms of plan operators that are independent of the underlying event structure, i.e., tackles the presentation issue that SCRIPTS were intended to bypass.

## 5.3 Previous Attempts to Generate Narrative Text

For several decades researchers have attempted to develop mechanisms to generate narrative. The three principal avenues of work have been story simulation, domain-independent story grammars, and domain-specific text grammars.

### 5.3.1 Story Simulations

One of the first computational implementations to write stories was Meehan's (1976, 1977) TALE-SPIN (introduced in Chapter 3). TALE-SPIN simulated the rational behavior of agents as they made and executed plans to achieve their extralinguistic goals (e.g., fulfilling desires for food, drink, rest, or sex). The program used STRIPS-like plans and characterized inter-agent relationships (e.g., competition, dominance, familiarity, affection, trust, deceit, and indebtedness), personalities (e.g., degrees of kindness, vanity, honesty, and intelligence), and physical space (as an abstract representation -- a kind of "mental model" map). Below is a typical TALE-SPIN story, narrated in third person:

```
ONCE UPON A TIME GEORGE ANT LIVED NEAR A PATCH OF GROUND.  THERE WAS A NEST IN AN ASH
TREE.  WILMA BIRD LIVED IN THE NEST.  THERE WAS SOME WATER IN A RIVER.  WILMA KNEW THAT
THE WATER WAS IN THE RIVER.  GEORGE KNEW THAT THE WATER WAS IN THE RIVER.  ONE DAY
WILMA WAS VERY THIRSTY.  WILMA WANTED TO GET NEAR SOME WATER.  WILMA FLEW FROM HER NEST
ACROSS A MEADOW THROUGH A VALLEY TO THE RIVER.  WILMA DRANK THE WATER.  WILMA WASN'T
THIRSTY ANY MORE.

GEORGE WAS VERY THIRSTY.  GEORGE WANTED TO GET NEAR SOME WATER.  GEORGE WALKED FROM HIS
PATCH OF GROUND ACROSS THE MEADOW THROUGH THE VALLEY TO A RIVER BANK.  GEORGE FELL INTO
THE WATER.  GEORGE WANTED TO GET NEAR THE VALLEY.  GEORGE COULDN'T GET NEAR THE VALLEY.
GEORGE WANTED TO GET NEAR THE MEADOW.  GEORGE COULDN'T GET NEAR THE MEADOW.  WILMA
WANTED GEORGE TO GET NEAR THE MEADOW.  WILMA WANTED TO GET NEAR GEORGE.  WILMA GRABBED
GEORGE WITH HER CLAW.  WILMA TOOK GEORGE FROM THE RIVER THROUGH THE VALLEY TO THE
MEADOW.  GEORGE WAS DEVOTED TO WILMA.  GEORGE OWED EVERYTHING TO WILMA.  WILMA LET GO
OF GEORGE.  GEORGE FELL TO THE MEADOW.  THE END.
```

The user/programmer created George and Wilma, the river, and a problem to solve: "thirst". Both George and Wilma want to rescue other characters in danger, and both knew George would have drowned if he stayed in the water. Wilma rescues him and as a result earns his devotion.

The content of this story is plausible because the underlying actions and reactions are rational. Linguistic realization, however, is not based on a syntactic grammar: entities and events in the underlying simulation are mapped fairly directly onto surface form (although TALE-SPIN could, for example, perform some lexical choice such as choosing between "Joe Bear went to the cave" versus "Joe Bear returned to the cave" by examining the knowledge base to see if Joe Bear has been in that cave before). However, there are no mechanisms for coreference or connection (e.g., focus models for pronominalization or the use of clue words). A more significant problem is that there is no content-independent knowledge of narrative techniques and of the effects they have on the reader. In particular, the structure of the story arises from the `goal ⇒ subgoal/event` structure on the stack of the problem solver. (This also makes the strong assumption that hierarchical manipulation of the goal stack mirrors human behavior.) Thus, the narration is essentially a trace of events as character's goals are solved by the planner. While this technique ensures the coherence of short stories, it fails to prune events that are not interesting or can be inferred (e.g., most readers know that drinking water quenches thirst). More important, to narrate a story in the real world, in which millions of events occur (with complex temporal and causal interdependencies), selecting and ordering the most salient events is a complex task. This becomes apparent in more sophisticated, multiagent simulations. For example, later in this chapter an Air Force simulator, LACE, is discussed in which a ten-second run generates thousands of events, only the most salient of which are organized and presented to the hearer.

Inspired by the Aesop fables, Meehan also considered how stories with morals could be produced. TALE-SPIN could produce a story for a moral like "Never do X" by setting up the simulation so that whenever someone did X something bad happened. While this made the important connection between high-level author goals and underlying actions in the story, Meehan's "story simulator" had no knowledge of general organizational principles for presenting the simulated events.

Dehn (1981) recognized this limitation and discussed the need to produce stories from the narrator's perspective. Her program, AUTHOR, distinguishes between the goals of the characters in the story and the goals of the author. "The author's goals serve as a sort of scaffolding in constructing the story; they are no longer directly visible in the final story, but are reflected in the storyworld situations and resolutions they gave rise to." (Dehn, 1981, p. 17). Since an author's goals are constantly changing as s/he writes a tale, Dehn claims the story-generation process is a successive reformulation of the author's goals. These revisions can make the story more plausible, dramatic, or informative. Story content is often based on the author's own experiences -- of personal episodes and acquaintances. While AUTHOR makes the important distinction between narrator and character, the system, like Meehan's TALE-SPIN, does not reason about the abstract textual structure of stories. As we will discuss, even though narrative text may be motivated by the author's goals, it has its own formal properties.

### 5.3.2 Story Grammars

Abstract story structure is precisely what story grammarians have attempted to capture in story grammars. Modeled after narrative prose forms such as folk tales, these grammars codify content-independent scenarios in the same spirit that context free grammars capture regularities in syntactic structures. Story grammars were first proposed by Lakoff (1972) who reformulated Propp's (1968) theory of the structure of Russian folktales in terms of rewrite rules. Rumelhart (1975) proposed the first more general story grammar, illustrated in Figure 5.1, with both syntactic and semantic rules. The greatest weakness in the story grammar formalism is its lack of specificity: terminal categories (e.g., "internal response" in Figure 5.1) lack explicit definitions and semantic rules rely heavily on world-knowledge. Other failings of story grammars are well documented in the literature (e.g., Black and Wilensky, 1979; Frisch and Perlis, 1981; Garnham, 1983; Wilensky, 1983).

Story grammars have some utility, namely the classification of repetitive stories. For example, they are able to capture the repetitive style of the biblical story of Genesis which essentially follows the pattern:

*DayN -> Divine-suggestion + object-creation-event + object-naming +"Evening came and morning followed, the nth day."*

*The syntactic rules*

| | | | |
|---|---|---|---|
| 1 | Story | -> | Setting + Episode |
| 2 | Setting | -> | (State)* [i.e., an arbitrary number of states] |
| 3 | Episode | -> | Event + Reaction |
| 4 | Event | -> | { Episode / Change-of-state / Action / Event + Event } |
| 5 | Reaction | -> | Internal response + Overt response |
| 6 | Internal response | -> | { Emotion / Desire } |

*The semantic rules (corresponding to each syntactic rule)*

1 Setting ALLOWS episode, i.e., makes it possible.
2 State AND State AND ..., i.e., logical conjunction of the states.
3 Event INITIATES reaction, i.e., an external event causes a mental reaction.
4 Event CAUSES event, or event ALLOWS event.
(No semantic rule is require for the first three options in the syntactic rule.)
5 Internal response MOTIVATES overt response
i.e., the response is a result of the internal response.
6 No semantic rule required.

Figure 5.1 Rumelhart's Story Grammar (from Johnson-Laird, 1983 p. 363)

| ITALIAN | ENGLISH |
|---|---|
| Alla fiera dell'est<br>per due soldi<br>un topolino<br>mio padre comprò | At the Eastern fair<br>for 2 pieces of money<br>a little mouse<br>my father bought |
| E venne il gatto<br>che si mangiò il topo<br>che al mercato<br>mio padre comprò | And then came the cat<br>that ate the mouse<br>that at the market<br>my father bought |
| E venne il cane<br>che morse il gatto<br>che si mangiò il topo<br>che al mercato<br>mio padre comprò | And then came the dog<br>that bit the cat<br>that ate the mouse<br>that at the market<br>my father bought |
| ... | ... |
| E in fine il Signore<br>sull'angelo della morte<br>sul macellaio<br>che uccise il toro<br>che bevve l'acqua<br>che spense il fuoco<br>che bruciò il bastone<br>che picchiò il cane<br>che morse il gatto<br>che si mangiò il topo<br>che al mercato<br>mio padre comprò | And in the end God<br>on the angel of death<br>on the butcher<br>that killed the bull<br>that drank the water<br>that extinguished the fire<br>that burnt the stick<br>that beat up the dog<br>that bit the cat<br>that ate the mouse<br>that at the market<br>my father bought |

Figure 5.2 Angelo Branduardi's *Alla Fiera dell'est*

Similarly, Figure 5.2 shows an abbreviated form and translation of a popular Italian folk-song which can be interpreted by the story grammar because of its regular recursiveness. As this example illustrates, the power of a context free grammar is unmotivated since a finite state machine which allowed for, say, 100 repetitions of the rule `event -> event + reaction` (rule #3 in Figure 5.1) would suffice for all actual stories with this structure.

In summary, story grammars attempt to separate general story *form* from particular story *content*. Unfortunately, their rules lack descriptive precision. Furthermore, they are dependent on the particular type of text being generated: their primitive elements cannot be used in other forms of text such as description or exposition and their relationship to other types of prose has not been investigated. Finally, like schemas, story grammars are compiled text plans. They capture neither the necessary conditions for their application nor the intended effect their use has on the knowledge, beliefs, and desires of the hearer.

### 5.3.3 Text Grammars for Report Generation

One proposed solution to the descriptive inadequacies of story grammars is a domain-dependent representation of discourse: *text grammars*. For example, after studying a number of stroke reports handwritten by physicians, Li *et al.* (1986) and Collier (forthcoming) captured their organization in a context free grammar. Their text planner produces two types of reports from a Stroke Consultant expert system—a current status report and a discharge report—using two different text grammars. The selection and order of facts in the reports are guided by the text grammar. The top-level rule below illustrates how a stroke case report consists of a series of more specific reports:

```
Case_Report -> Initial_Information + Medical_History + Physical-Examination +
               Laboratory_Tests + Final_Diagnosis + Outcome
```

The first constituent expands to:

```
Initial_Information -> Patient_Information + Admission + Chief_Complaint + Defect_Evolution
```

The elements `Chief_Complaint` and `Defect_Evolution` on the right-hand side of this rewrite rule are already terminal leaves in the grammar. The first element rewrites as the terminal leaves:

```
Patient_Information -> Registration_Number + Age + Handedness + Race + Sex
```

When the system reaches a leaf node of the grammar, it accesses the database of facts and produces a surface form using the Linguistic String Parser (LSP) (Sager, 1981) and a stroke lexicon of about 3560 entries. In building phrases this surface generator may embed clauses to combine related propositions. For example, in giving the history of a patient, a number of facts can be combined to form the sentence:

```
PATIENT 137 IS A 70 YEAR OLD RIGHT-HANDED WHITE MAN ADMITTED FOR A STROKE ON AUGUST 10,
1983, WITH A MILD BILATERAL WEAKNESS.
```

While the output is indeed impressive, the text grammar approach limits the variability of the resulting text. This is evident when one examines corresponding paragraphs from two different reports. Consider:

```
PATIENT 127 IS A 48 YEAR OLD RIGHT-HANDED WHITE MAN ADMITTED FOR A STROKE WITH A MILD
LEFT-SIDED WEAKNESS.  THE DEFICIT CAME ON WHILE HE WAS CARRYING ON THE NORMAL
ACTIVITIES OF DAILY LIVING.  THE DEFICIT WAS MAXIMAL AT ONSET.  AT THE ONSET OF THE
DEFICIT THERE WAS NO HEADACHE, NO IMPAIRMENT OF CONSCIOUSNESS, NO SEIZURE ACTIVITY, AND
NO VOMITING.
```

These domain-dependent text grammars do not structurally vary the text, say by varying the order of sentences. This lack of variety is illustrated by comparing the above paragraph for patient 127 to the analogous one for patient 137 below:

```
PATIENT 137 IS A 70 YEAR OLD RIGHT-HANDED BLACK MAN ADMITTED FOR A STROKE WITH A MILD
BILATERAL WEAKNESS.  THE DEFICIT CAME ON WHEN HE WAS SLEEPING.  IT WAS MAXIMAL AT
ONSET.  AT THE ONSET THERE WAS NO HEADACHE, IMPAIRMENT OF CONSCIOUSNESS, SEIZURE
ACTIVITY, OR VOMITING.
```

This structural repetition may be desirable in domains where thoroughness is essential or where the consumers of reports prefer standardized output. Despite the relatively fixed paragraph structure, the linguistic string parser does produce syntactic variation. For example, notice how the same content is conveyed by distinct structures in the last sentences of the two examples. However, these examples do raise questions about the reference mechanisms. It is unclear, for instance, why the third sentence is pronominalized in this passage but not in the earlier one -- the attentional context is equivalent.

In contrast to story grammars, which have very general non-terminal categories and suffer from vaguely-defined terminal ones, text grammars are particular to one domain, that is, both their terminal and non-terminal categories are domain specific. Thus, text grammars do not explicitly represent different types of text: those that describe people or concepts, narrate events, or argue for a particular treatment. For example, the above medical report paragraphs mix description and narration. Furthermore, text grammars do not capture interclausal rhetorical relations (e.g., exemplification, purpose, elaboration) which are often signalled by connectives (e.g., "for example", "in order to", "in particular") and are used to increase text cohesion. Also, text grammars do not capture the effect of the text, as a whole or in its parts, on the knowledge, beliefs, or desires of the hearer, which may of course be sufficient in a non-interactive setting or where all users have similar characteristics. Even though multiparagraph reports were generated, the representation of the text is rather shallow because text grammars simply indicate preferred content sequencing much like McKeown's (1982) rhetorical schemas, but in a domain-dependent manner. Like subgrammars used in machine translation, text grammars can be quite effective in restricted domains (as here) but it is unclear if they will be generally applicable.

### 5.3.4 Sublanguages for Report Generation

Instead of using an explicit domain text grammar, other researchers have investigated the use of sublanguages and domain/task-specific text organizations for report generation. Kukich (1983, 1985b, 1988) analyzed human-produced stock reports (at least 20) in order to develop a prototype system for generating daily market summaries. Her system, ANA, produced the first three paragraphs of the daily stock market report using a set of half-hourly price and volume quotes from data from the Dow Jones News Service database. Figure 5.3 displays a typical output together with the Wall Street Journal's report for the same day written by Victory J. Hillery.

Just as the text grammar for the stroke report selects and orders domain-specific information, ANA examines a database of half-hourly quotes from one day of trading and collects semantically-related messages concerning 10 specific market issues/indicators including "closing market status", "volume of trading", "mixed market", and "interesting fluctuations". A relatively simple "discourse organizer" groups and prunes these semantic messages according to a standard ordering found in human-produced stock reports. The ordered messages are then linguistically realized using a slot-filler template associated with

**ANA'S Stock Report**

wall street's securities markets meandered upward through most of the morning, before being pushed downhill late in the day yesterday. the stock market closed out the day with a small loss and turned a mixed showing in moderate trading.

the Dow Jones average of 30 industrials declined slightly, finishing the day at 810.41, off 2.76 points, the transportation and utility indicators edged higher.

volume on the big board was 55860000 shares compared with 62710000 shares on Wednesday. advances were ahead by about 8 to 7 at the final bell.

**Wall Street Journal Stock Report**

The stock market finished with mixed results after the attempt to push its rebound into the fourth session faltered in continued active trading. Technology issues, Wednesday's star performers, were among yesterday's biggest losers. Some of the drug, oil and steel issues also were casualties.

The Dow Jones Industrial Average, which bounced back 24.55 points in the prior three sessions after plunging more than 80 points since early May, was up 5.04 points at 1:30 p.m. EDT yesterday. However, the index posted a 2.67 point loss an hour later and then closed at 810.41, down 2.76 points. The transportation and utility averages both moved higher.

New York Stock Exchange gainers led by better than two to one early in the day but at the final bell were ahead about seven to six.

[two paragraphs deleted -- commentary from financial and industry experts]

Big Board volume slowed to 55,860,00 shares from 62,710,00 Wednesday. Somewhat lower institutional activity was indicated by the decline in trades of 10,000 shares or more to 905 from 1,053 in the prior session.

[4 paragraphs extracted on performance of key stocks, AMEX, and Nasdaq.]

Figure 5.3 ANA and Wall Street Journal Stock Reports each message and a phrasal lexicon (just under 600 entries). Even though ANA produced only one type of paragraph typically found in stock market reports (regarding the trend and volume of the industrial average), the quality of the text makes it virtually indistinguishable from human text. The OPS5 production rule implementation required 5 minutes of real time on a lightly loaded VAX11780 processor to produce three relatively short paragraphs. Kukich's estimate of 25 minutes of processing for a complete stock report forced the program to be used in batch mode. Contant's (1986) FRANA system later replaced ANA's linguistic module with a French linguistic realization component.

An analogous system, RAREAS, was developed to generate Arctic marine weather forecasts in English from formatted data (Kittredge et al, 1986). After examining over 100,000 words of marine forecasts in English and French, a bilingual version, RAREAS-2 (Polguère, 1987), was developed and delivered in 1987 to Environment Canada for "testing, extensions, and implementation in regional weather offices." Like ANA, RAREAS used the sublanguage approach. In particular, RAREAS exploited:

- domain-specific lexical semantics
- syntactic patterns
- frequency preferences (e.g., among synonyms in marine bulletins)
- knowledge about the saliency of content
  - warnings preceded normal weather
  - sentence groupings follow the order:
    `WINDS > CLOUD-COVER > PRECIPITATION > FOG&MIST > VISIBILITY`
- causal or temporal connections between meteorological events.

This domain knowledge constrains the selection, order, and realization of content. First, a pre-linguistic

```
DATA:
            2200 mon 83/09/22 end.
            frob wind 220 30 &
                 nt 5 300 35 & nt 18 speed 40
                 wea rain cont heavy &
                    nt 15 nl n 65 rain per moderate
                 temp -3
                 end.
```

INTERPRETATION:
Line 1 of the above formatted data identifies the Greenwich time of report validity. The beginning of line 2 indicates the area concerned, Frobisher Bay (`frob`). The remainder of the data specifies initial values for each important weather parameter (e.g., `wind`, `rain`). Subsequent changes in the value of a parameter are preceded by the number of hours until the forecast changes. For example, line 3 says that 5 hours after the initial reading in line 2 (30 knot wind at 220 degrees), the wind will change direction (to 300 degrees) and speed (to 35 knots) and then 18 hours later winds will increase speed to 40 knots.

Figure 5.4 Weather Report Data (from Kittredge, 1988, p. 3-4).

module uses the formatted data shown in Figure 5.4 to compute additional weather parameters such as dangerous wind and freezing spray conditions. The domain-oriented text planner then uses this information to produce the English text:

```
marine forecasts for arctic waters issued by environment canada
at 3.00 pm mdt monday 22 september 1983
for tonight and tuesday.

frobisher-bay
gale warning issued ...
freezing spray warning issued ...
winds southwesterly 30 veering and strengthening to northwesterly
gales 35 late this evening then strengthening to northwesterly
gales 40 late tuesday afternoon.  cloudy with rain then showers
developing north of 65 n latitude tuesday.  visibility fair in
precipitation.
```

as well as the corresponding French text:

```
previsions maritimes pour l'arctique emises par environnement
canada a 10h00 har le lundi 22 septembre 1983
pour cette nuit et mardi.

frobisher-bay
avertissement de coup de vent en vigeur ...
avertissement d' embruns verglacants en vigeur ...
vents du sub-ouest a 30 virant et se renforcant a coups de vents
du nord-ouest a 35 tard ce soir puis se renforcant a coups de
vents du nort-ouest a 40 tard mardi apres-midi.  nuageux avec
pluie puis averses commencant au nord de la latitude 65 n mardi.
visibilite passable sous les precipitations.
```

After producing the bilingual weather report generator, Kittredge et al. (1988) focused on linguistic realization in the context of generating hardware utilization reports in their GOSSIP (Generation of Operating Systems Summaries in Prolog). Guided by fixed, domain-specific plans, GOSSIP produced the following report using facts from an operating system audit trail:

```
The system was used for 7 hours 32 minutes 12 seconds.  The users of the
system ran compilers and editors during this time.  The compilers were
run six times, for 47 % of the cpu-time.  The editors were run twelve
times, for 53 % of the cpu-time.  Two users, Jessie and Martin, logged
on to the system.  Jessie used the system for 63 % of the time in use.
Martin used the system for 40% of the time in use.
```

English surface form is produced using fairly direct mapping of the content of “messages” onto linguistically marked sentence fragments followed by domain-specific grammatical adjustments. This direct translation is possible only if the relationship between information structure and language structure is “relatively transparent”. Because the input to these report generators is constrained, there is less of a need to reason in general about content selection and organization that longer texts demand. Equally, the goal(s) and structure of the text are fixed and so there is no need to reason about speaker or hearer models to determine content.

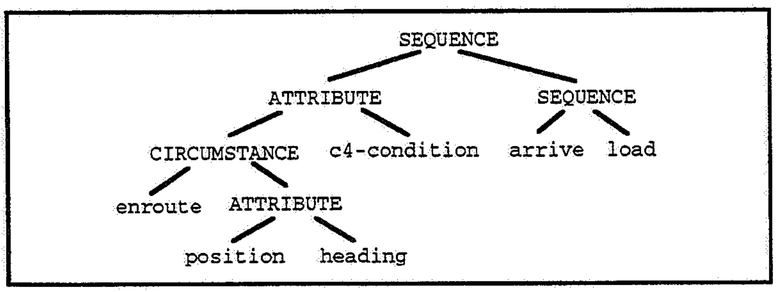
In summary, each of the above report generators -- for strokes, stocks (ANA), weather (RAREAS), or computer use (GOSSIP) -- simplifies the generation process by limiting the lexica and syntax to the sublanguage employed by domain experts. By using a text grammar appropriate to the specific application and domain sublanguage, ambiguous or other problematic linguistic structures can be avoided. Finally, these works neither focus on content-independent text structure nor are they concerned with the effects of different kinds of information of the hearer's knowledge, beliefs or desires. In short, these report generators are efficient, but restricted to the generation of domain-dependent reports.

### 5.3.5 RST Sequencing

Hovy's (1988) content "structurer" operationalizes part of Rhetorical Structure Theory (RST) (Mann and Thompson, 1987) and embodies a more general approach to narrative planning. As described in Chapter 3, Hovy uses his `sequence` plan operator (see Figure 3.3) to produce the following narration of events in a naval domain:

```
Knox, which is C4, is en route to Sasebo.
Knox, which is at 18N 79E, heads SSW.
It arrives on 4/24.
It loads for 4 days.
```

which corresponds to the text structure:




As illustrated in Chapter 3, the `sequence` operator first tests to make sure two given "actions" (not events) are (1) contiguous in some sequence, and (2) are a "main-topic". At any given point in the sequence, the nucleus of the `sequence` operator allows the text to "grow" and indicate the circumstances, attributes, and/or purpose of an action. Similarly, the satellite of the `sequence` operator (for subsequent actions) allows the text to indicate the attributes and/or details of the next action. It also allows the text to recursively call the `sequence` operator on the next action in the sequence. Hovy's operators, like text schemas, focus more on structuring information than on formalizing the effects various orderings or the addition of information at growth points should have on the hearer. (While not addressing narration, Moore (1989) investigated operationalizing the specific effects of using particular RST relations on the hearer's beliefs.) It is therefore unclear how Hovy's `sequence` plan operator could be modified to characterize the motivation for selecting among the different arrangements that narrative employs to achieve specific effects on the hearer (e.g., creating interest, suspense, or mystery). In addition, the `sequence` operator does not consider states as first-order objects in some causal chain (the fact "Knox is C4" is just an attribute extending off the "en route" event). This is important because states have complex relations (e.g., enablement, causation) to other states and events in the world, but these are not exploited to organize text in Hovy's system. Finally, Hovy treats sequences as contiguous, yet events are often simultaneous or overlapping in time.

As indicated in Chapter 3, this purely RST-based approach was improved upon by Hovy and McCoy (1989) by incorporating Focus Trees (McCoy and Cheng, 1991) which guide the ordering and interrelationships of sentence topics. This combined approach produced:

```
With readiness C4, Knox is en route to Sasebo.
It is at 79N 18E heading SSW.
It will arrive 4/24 and will load for four days.
```

Text coherence is improved here not only by regrouping content (a result of restrictions on the traversal of the Focus Tree) but also by using tensed verbs (e.g., future tense of "arrive" in the last utterance) to explicitly indicate the temporal relations among events. Unfortunately, no details of how this tense is generated are provided.

As Hovy and McCoy's work illustrates, more sophisticated representations of verb tense and aspect (cf. Allen, 1988) are critical to generating coherent narrative forms. This demands a more sophisticated representation of events, states, and their relationship to tense and aspect. The remainder of this chapter details how TEXPLAN characterizes events and temporal relations and how these are used in tandem with narrative plan operators to capture specific narrative techniques.

In summary, several organizational techniques have been suggested to present events and states. Conceptual templates associated with scripts (Schank, 1975) were used to present summaries of stereotypical event sequences, although these had a fixed order and were unable to handle non-standard event scenarios. Whereas story simulations like Meehan's (1976, 1977) conflated author and events, story grammars separated story form and story content, but were criticized for their overgenerality. Domain-dependent techniques such as text grammars and sublanguages overcome this limitation but, in consequence, lack general applicability. More recent work based on RST attempts to capture domain-independent organizational principles in a plan-based approach. Unfortunately, only one "action" sequence plan operator has been proposed (Hovy, 1988a), which does not address how differing presentational orderings potentially effect the reader's knowledge and is not based on a formal ontology of events and states, a topic addressed in the next section.

## 5.4 Narration in TEXPLAN

Unlike past attempts at narration, TEXPLAN reasons abstractly as an author would about the content, structure, and intended effect of the narrative it produces. Since events and states form the backbone of narratives, the first subsection of this section details the event and state ontology that defines the input to TEXPLAN's narrative plan operators. The second subsection then describes how temporal and aspectual information is conveyed grammatically by the verb and adverbials. A final subsection introduces the notion of temporal and spatial focus which guides the realization of the verb and adverbials.

After this discussion of events and states, tense and aspect, and temporal and spatial focus, the remaining sections in the chapter illustrate TEXPLAN's various narrative plan operators. These plan operators characterize three types of narrative text that organize events: reports (temporally or topically sequenced events), stories (causally sequenced events), and biographies (event sequences concerning one agent). The communicative acts underlying these three types of narrative text are formalized as plan operators. The chapter concludes by discussing how plan operators can characterize narrative techniques that can create more complex effects on the hearer, such as surprise, suspense, and mystery.

### 5.4.1 Event and State Ontology

Representing and linguistically realizing (articulating) events concerns issues of temporality, causality, and enablement as well as verb tense and aspect. Discussion of noninstantaneous events dates at least to Aristotle's distinction between process (*energia*) and state (*stasis*). Recently, these issues have been the focus of attention in philosophy (Dowty, 1986), linguistics (Tedeschi and Zaenen, 1981), and computational linguistics (Moens and Steedman, 1988; Nakhimovsky, 1988; Webber, 1988). Several researchers have begun building computational implementations (Hinrichs, 1988; Passonneau, 1988; Allen et al, 1990). While I make no claim of great depth of treatment of events and states, my research drew from this previous work to develop a taxonomy of and knowledge representation for events and states.

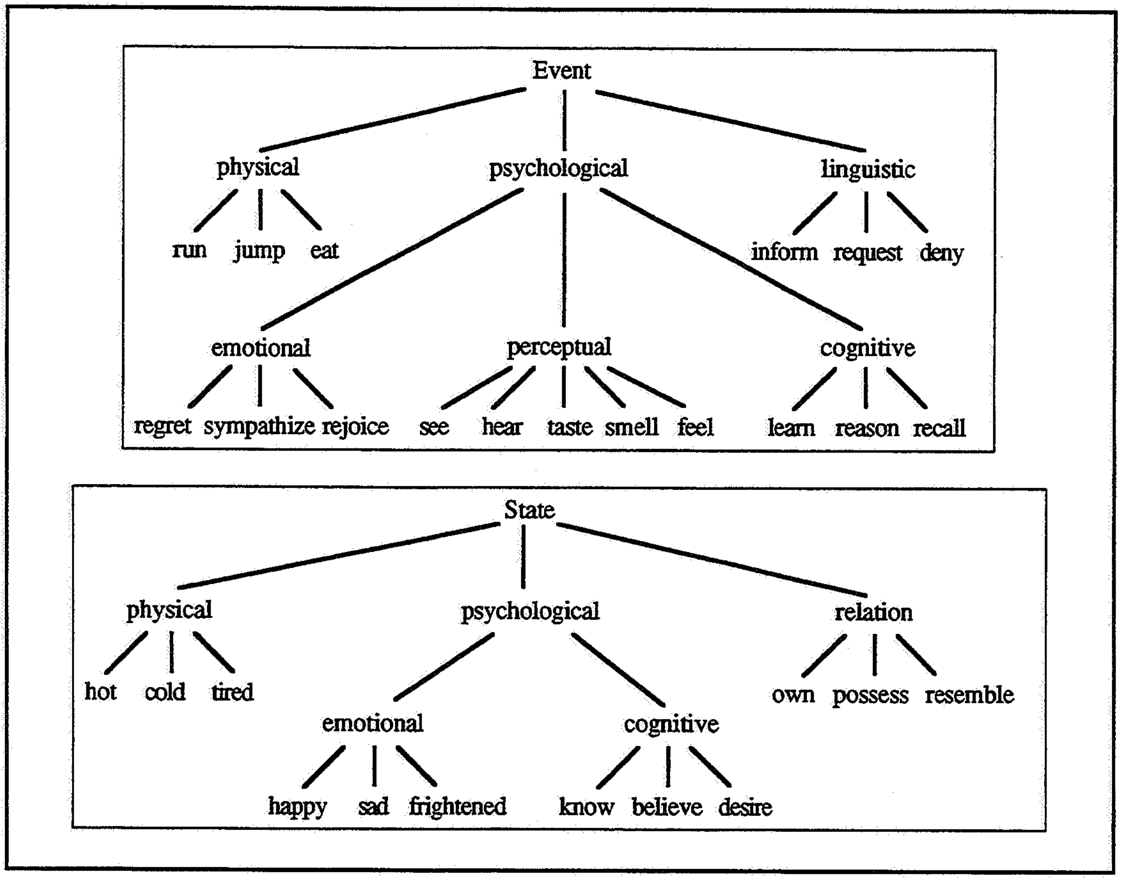



Figure 5.5 Taxonomy of Events and States

The input to TEXPLAN's narrative plan operators are events and states. Figure 5.5 illustrates the classification of events and states used in TEXPLAN. *Events* are physical, linguistic, or psychological happenings at some point(s) in space and time. *States*, in contrast, refer to temporally unbounded physical, psychological, or emotional properties of an entity or agent. A physical state, for example, can refer to the condition of an agent (e.g., hot, cold, or tired) as well as the structure, form, or phase of an entity (e.g., gaseous state or larval state). States also include relations that hold between agents or entities (e.g., possession, ownership) (Nakhimovsky, 1988). Some event and state classes are interrelated. For example, a perceptual event is an impression of an external stimulus that may involve cognitive processing. More complex, emotional states (e.g., happiness) result from emotional events (i.e., internal feelings). Emotional states may be accompanied by changes in physical state (e.g., changes in heart rate, respiration, perspiration) as well as overt physical manifestations or events (e.g., crying, shaking, and laughing). This general classification can be specialized. For example, physical events can be subclassified into events regarding motion, translocation, ingestion and so on. However, this is not the aim of this work. It is also important to note that this classification is but one (conceptual) classification of events and states. Ehrich (1986), for example, uses the features of duration, resultativity, and intentionality to produce an orthogonal categorization.

Each event or state is represented in a frame-like structure as illustrated in Figures 5.6 and 5.7. A collection of related events and states constitutes an *event/state structure* analogous to Webber's (1987) event/situation structure. This network of events and states serves as the basis for narrative generation. Events and states have associated attributes, roles, and relations. The term *attributes* refers to characteristics local to the event or state such as its time of occurrence (a point or interval), its type (e.g., physical, psychological), and any constituents (i.e., subevents or substates). *Roles* refer to the role an entity plays in the event or state (e.g., agent, beneficiary). Finally, causal *relations* refer to the associated enablement(s), cause(s), and effect(s) of an event or state. *Actions*, which are events that involve some agency (i.e., an agent with intentions), can also have associated motivations and purposes. A *motivation* refers to an inner urge (the consequence of a physical or psychological state) that moves an agent to act. This is distinct from a *reaction* (classified as a causation) which is a response, often involuntary, to a stimulus (e.g., an event or state of affairs). *Purpose* refers to the goal state or intended action of an agent.

**EVENT**

| | |
|---|---|
| Attributes | |
| time: | instantaneous (point) or duration (interval) |
| type: | e.g., physical, psychological, or linguistic |
| location: | e.g., in Rome, in the kitchen, 25° 5' 26'' longitude 30° latitude, etc. |
| constituents: | subevents |
| ... | |
| Roles | |
| agent: | |
| patient: | |
| beneficiary: | |
| ... | |
| Relations | |
| enablement: | event-or-state |
| motivation: | state |
| cause: | event-or-state |
| effect: | event-or-state |
| purpose: | event-or-state |

Figure 5.6 Representation of Events

**STATE**

| | |
|---|---|
| Attributes | |
| time: | instantaneous (point) or duration (interval) (e.g., ice is solid for ...) |
| type: | e.g., physical, psychological, or relation |
| location: | e.g., (it's cold) in Rome. |
| degree: | e.g., 10% complete, partially (cold, broken) |
| constituents: | substates |
| ... | |
| Roles | |
| agent: | |
| patient: | |
| beneficiary: | |
| ... | |
| Relations | |
| enablement: | event-or-state |
| motivates: | event |
| cause: | event-or-state |
| effect: | event-or-state |

Figure 5.7 Representation of States

States differ from events in that they are perpetual or unbounded in time (unless stated otherwise as in "Sally was sick until she took her medicine."). Processes, in contrast to states, involve changes or transformations over the interval for which they hold and often have some associated rate of progress toward a goal or a rate of consumption of resources (Nakhimovsky, 1988). Nakhimovsky (1988) makes a key distinction between events and processes:

> For a linguist, the distinction between **event-process** is one of aspectual perspective: "The term 'process' means a dynamic situation viewed imperfectively, and the term 'event' means a dynamic situation viewed perfectively" (Comrie, 1976: 51). The distinction **process-state** is one of aspectual class.

That is, an event is a completed process. Because event/process is principally a linguistic distinction, both utilize the common knowledge representation shown in Figure 5.6. The term *event* is used to refer both to instantaneous events (e.g., snap, click, wink) as well as to events with a duration, which can be viewed perfectively (event) or imperfectively (process). (See Nakhimovsky (1988) for a classification of instantaneous events (e.g., happening, transition, culmination, disturbance, activation, or switch).) Events can be a culmination of some goal or purpose (Moens and Steedman, 1988). A *telic event* is one with a definite endpoint in time such as goal-directed behavior (e.g., "I am running to school") or processes that consume finite resources (e.g., "The log is burning.") (Nakhimovsky, 1988).

### 5.4.2 Tense and Aspect

Because events and states incorporate rich references to time, attention must be paid to their proper verbalization. In English, tensed verbs (e.g., simple past, present, and future; and past, present, and future perfect) play a key role in event and state narration. Traditionally, English tense has been explained by (1) the absolute time of an event and (2) the time of the event relative to the time the tensed clause was uttered.[1] In 1947, Reichenbach instead proposed a tripartite interpretation of tense which includes: the point or time at which the utterance is spoken (S), the point at which the event happens, i.e., the absolute time (E), and the point of reference (R). This last time, R is the time "talked about" or "focused on" (i.e., the time the overall narration takes place). R is the same as S in present perfect (e.g., John has eaten. $E < R = S$, where "<" means temporally before). R also equals S in the simple present tense (e.g., John eats. $E = R = S$). In the simple past and simple future, R is the same as the event time, E. The distinction between simple past and simple future is what time the speaker is narrating from. We say "John ate the beans" if $E = R < S$, but we say "John will eat the beans" if $S < E = R$. Two final cases involve the past perfect and future perfect tense. In the former, $E < R < S$ as in "John had eaten the beans." In the latter case, $S < E < R$, as in "John will have eaten the beans."

Because TEXPLAN captures the absolute time of the event, i.e., the Reichenbachian event time (E) in the event structure, it can select the appropriate verb tense by reasoning about the time the speaker is narrating (S) and the time the overall narration focuses on (R). Table 5.1 indicates how TEXPLAN relates time, tense, and aspect (both perfectivity and progressiveness). While the current implementation incorporates a point-based time representation, this could be extended to consider time intervals. Allen (1984) suggests a temporal logic for relating temporal intervals (e.g., BEFORE, EQUAL, MEETS, OVERLAPS, DURING, STARTS, FINISHES).

Following Winograd (1983), TEXPLAN's sentence generator uses the prototypical verb sequence: Modal + Have + Be1 + Be2 + Main-verb. Individual verbs include both *modals* such as "will", "can", "could" (which have only one form), and ordinary verbs which have five basic forms in third person, singular: infinitive ("swim", "be", "walk"), simple present ("swims", "is", "walks"), simple past ("swam", "was", "walked"), present participle ("swimming", "being", "walking"), and past participle ("swam", "been", "walked"). Future tense does not have its own syntactic form, but it is implemented by the modals "will" or "shall". Tense is defined by the relationship of the time intervals of S to R and E as shown in Table 5.1.

---

[1]This description of Reichenbach's work is based on part of Bonnie Lynn Webber's foreword to "Special Issue on Tense and Aspect," *Computational Linguistics*, 14(2), June 1988.

| Example | Time | Tense | Verb Structure | Aspect | |
|---|---|---|---|---|---|
| | | | | Perfectivity | Perspective |
| Mary sings. | E=R=S | simple present | simple present | no | nonprogressive |
| Mary sang. | E=R<S | simple past | simple past | no | nonprogressive |
| Mary will sing. | S<E=R | simple future | WILL + infinitive | no | nonprogressive |
| Mary has sung. | E<R=S | present perfect (or present/past) | HAVE in present + past participle | yes | nonprogressive |
| Mary will have sung. | S<E<R | future perfect | WILL + HAVE in infinitive + past part. | yes | nonprogressive |
| Mary had sung. | E<R<S | past perfect or pluperfect (or past/past) | HAVE in past + past participle | yes | nonprogressive |
| Mary is singing. | E=R=S | present progressive | BE in present + present participle | no | progressive |
| Mary was singing. | E=R<S | past progressive | BE in past + present participle | no | progressive |
| Mary will be singing. | S<E=R | future progressive | WILL + BE in infinitive + present participle | no | progressive |
| Mary has been singing. | E<R=S | past perfect progressive | HAVE in present + BE in past participle + present participle | yes | progressive |
| Mary will have been singing. | S<E<R | future perfect progressive | WILL + HAVE in present + BE as past participle + present participle | yes | progressive |
| Mary had been singing. | E<R<S | pluperfect progressive | HAVE in past + BE in past participle + present participle | yes | progressive |

Table 5.1 Relationship of Time, Tense, and Aspect
(verb structure based on Allen (1987, p. 31-32))

In contrast to tense, aspect is a grammatical category of the verb implemented by affixes, auxiliaries, and so on (Nakhimovsky, 1988). This arises from both the temporal characteristics of the underlying event (i.e., point versus interval), as well as the relationship of the reference time (R) to event time (E) (e.g., $E < R$ indicates perfective; $E = R$ imperfective). Perfective sequences use Have (e.g., has taken), progressive use Be1 (e.g., was taking), and passive use Be2 (e.g., was taken).

### 5.4.3 Temporal Focus and Spatial Focus

While event time is explicit in the underlying event representation and speech time can be bound at the time of linguistic realization, TEXPLAN computes the reference time by following shifts in *temporal focus* (TF). Webber (1988) proposed TF as the event currently being attended to in time. She suggests that TF is used to integrate events into some evolving spatio-temporal event/situation structure. TF can shift depending on the relations that hold between events and their times of occurrence. Webber (1988) suggests three TF shifts: maintenance, forward, and backward. Nakhimovsky (1988) classifies TF shifts as forward, sideways, and backward "micromoves". Forward and backward shifts correspond to introducing the consequence or preparatory phases of events (Moens and Steedman, 1988). Backward shifts start a new discourse segment. In TEXPLAN, TF indicates the Reichenbachian (1947) reference time. Local TF shifts (micromoves) are implemented via the plan operators and are ordered as follows:

1. Maintain current TF (maintenance)
2. TF progresses "naturally" forward (progression)
3. Shift TF to a simultaneous event/state (lateral shift)

In addition, two other long distance temporal shifts are possible but are not addressed in the current implementation:

4. Shift TF to a prior event/state (flashback)
5. Shift TF to a future event/state (flash-forward)

Temporal shifts are conveyed to the reader in part by verb tense and aspect, as in the use of future tense in "John just arrived. He was in an accident yesterday and ..." Temporal shifts are also indicated by adverbials (e.g., "five minutes later"), explicit references to time ("at seven p.m."), and clue words (e.g., "simultaneously"). TEXPLAN tracks TF by recording pointers to events that appear in the propositional content selected by the text planner, just as it records DF from selected propositional content following McKeown (1982). As with DF, past, current, and potential temporal focus registers are updated after each utterance.

Just as discourse can be topically and temporally organized, psychologists have observed that humans utilize spatial organizations, for example when people describe their apartments (Linde and Labov, 1975). Shifts analogous to those of DF and TF can occur along the dimension not of discourse or time, but rather of space. I define *spatial focus* (SF) as the current entity or group of entities (and its/their associated spatial location) that the reader is attending to in space. The notion of spatial focus is related to but distinct from Conklin's (1983) notion of *visual saliency*. Visual saliency is the noteworthiness (from one perspective) of an entity in relation to a set of static objects. Spatial focus, in contrast, refers to a currently focused entity (a "moving target") that is spatially related to the other entities currently in the background (static entities) or foreground (dynamic entities). Just as DF and TF follow regular shifts, the following ordered legal shifts appear to govern SF:

1. Maintain the current SF
2. Shift SF to an entity spatially related to the current SF
3. Shift SF to some distant point or region

Shifts in rule 2 can be relational (e.g., behind, in-front-of, left-of, right-of, above, below, on-top-of) or in terms of distance (e.g., "five miles away"). Shifts in rule 3 signal a new discourse segment. Just as TF can refer to points or intervals of time, SF can refer to either a point in space ("At 23° latitude 5° longitude"), a region (e.g., "In Chesterville today, ...") or a set of points or regions (analogous to discourse focus spaces (Grosz, 1977)). After each utterance, by examining the underlying propositional content TEXPLAN updates global registers that encode the past, current, and potential spatial foci. SF is used to realize locative instructions, detailed in the next chapter. In the current implementation, the system has a preference ranking over different types of ordering: thus it prefers topical over causal over temporal over spatial orderings.

The remainder of this chapter illustrates how TEXPLAN uses the notions of Reichenbachian time and temporal focus to narrate events and states. By tracking TF and exploiting the temporal information in the underlying event/state model, TEXPLAN is able to select proper verb tense and aspect as well as indicate shifts in TF, for example, through the use of adverbials. The sections that follow focus on the generation of reports, stories, and biographies from event/state networks. While reports and biographies were tested using a knowledge based simulation system, LACE, stories were generated from a hand-encoded event/state network so that causal and motivational information could be explicitly indicated.

## 5.5 Reports in TEXPLAN: The LACE Application

Given the means of representing events and states, their temporal structure, and their relation to tense and aspect just detailed, this section turns to the task of planning and realizing narrative. The most basic form of narration recounts events in their temporal order of occurrence. This occurs in a report, journal, record, account, or chronicle, collectively termed a *report.* Reports typically convey the most important or salient events in some domain during one period of time (e.g., stock market report (Kukich, 1983, 1985b, 1988), weather report (Kittredge, 1988), news report, battle report). Sometimes reports focus on events involving one dimension of an agent as in a medical record, an educational record, or a political record. The `narrate-report-temporally` plan operator in Figure 5.8 gets the hearer to know about the events by presenting the most salient ones in their order of occurrence in the simulation. *Saliency* is determined by the frequency, uniqueness, importance, and so on of events in the domain.

Narrative plan operators were tested in LACE (Land Air Combat in ERIC), a knowledge based battle simulation system (Anken, 1989). LACE is coded in ERIC, an object-oriented simulation language (Hilton, 1987). In LACE, multiple autonomous agents interact simultaneously to achieve their individual goals. For example, attacking forces attempt to bomb targets, refuel aircraft, move cargo, and suppress ground forces with electronic countermeasures. In contrast, defending forces attempt to detect, track, and destroy intruders. To give a feel for the nature of the complexity of the simulation, there are over 150 classes of entities each with dozens of behaviors. In a typical run of the simulation, hundreds of instances of objects are generated. And if several agents (e.g., 10 or 15) are given goals to pursue at the start of a simulation run, their actions generate thousands of events per minute as agents react to their environment and the behavior of other agents in the simulation. For example, if a long-range radar detects an intruding aircraft it will order its associated mobile surface-to-air-missile sites to electronically track, pursue (i.e., along the ground), and fire at the incoming target. LACE is challenging because it is necessary to produce a coherent account of dense multiagent action. The generation task, then, is to produce a report of the

```
NAME              narrate-report-temporally
HEADER            Narrate(S, H, events)

CONSTRAINTS       Temporal-Sequence(events) ∧ ∀e ∈ events  Event(e)
PRECONDITIONS
  ESSENTIAL       ∀e ∈ events  KNOW-ABOUT(S, e)

EFFECTS           ∀e ∈ events  KNOW-ABOUT(H, e)

DECOMPOSITION     ∀e ∈ temporally-ordered-events
                        Inform(S, H, Event(e))

WHERE             temporally-ordered-events =
                        Select-and-order-temporally(events)
```

Figure 5.8 `narrate-report-temporally` Plan Operator

events after simulating conflicts between two opposing military forces. Over fifty texts where produced, a sample of which is discussed below.

The input to TEXPLAN's narrative plan operators is an event/state network such as that described in the previous section. A preprocessing module was built to construct this event/state network from the LACE simulation. Each machine second for which the simulation clock ticks, LACE records the events that occur at that moment. The simulation measures time using Common Lisp's universal time (i.e., as seconds since the year 1900). These snapshots (e.g., at time 34300023 #<SAM-291> began sweeping) must then be interpreted into the representation of events or states shown in Figure 5.6 and Figure 5.7, with their associated properties (i.e., in the case of an event, its attributes such as time, location, duration; its relations to other events such as causal and temporal connections; and any associated roles such as the agent, patient, and so on). Collectively these structures characterize the event/state network that represents an overall spatio-temporal picture of the simulation.

For example, the diagram in Figure 5.9 illustrates a portion of the network constructed after a typical run of the simulation. The diagram shows three LACE domain events (e.g., fire) and their associated attributes, roles, and temporal relations. This event/state network is processed to prune details. For example, persistent or uninteresting (e.g., frequent, non-unique, or unimportant) events can be deleted. Similarly, a number of abstractions can be made from this representation. For example, events and states can be grouped into time segments.

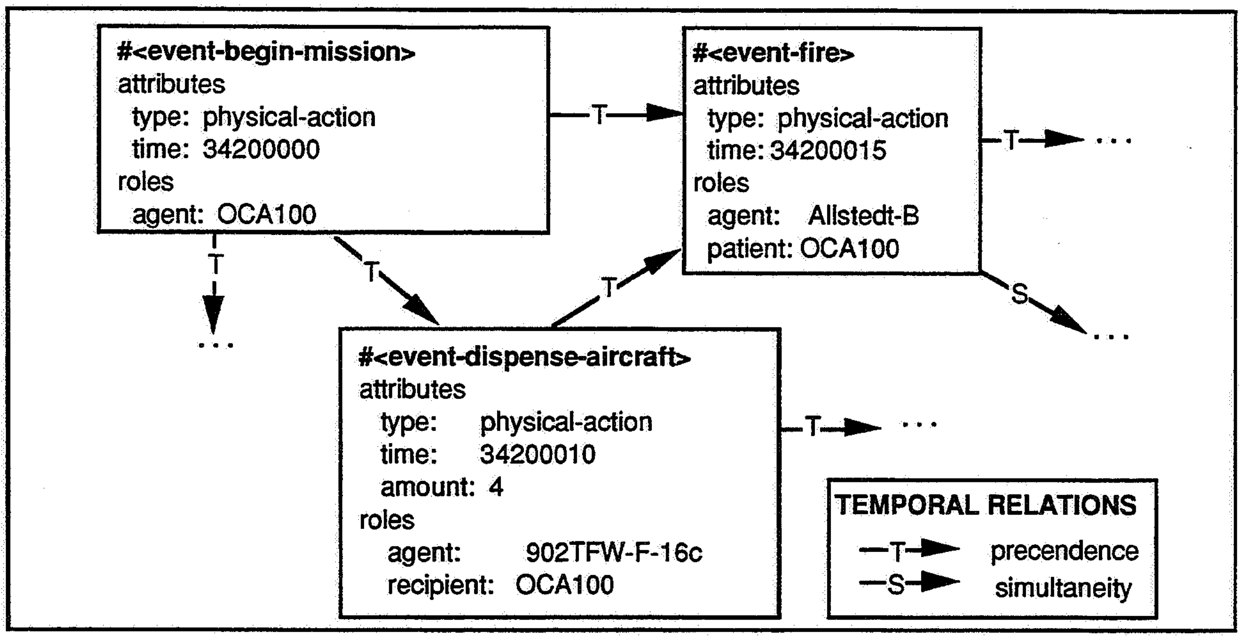



Figure 5.9 Portion of LACE Event/State Network

However, one of the problems is that some important information is not available when the event/state network is built. Information about motivations, enablements, causation, and purpose (i.e., agent intentions and goals), as well as state changes, is implicit in the snapshots of the simulation. This is because the narrator does not have a "glass" eye perspective on behaviors internal to agents. One possibility would be to enrich the event/state network by attempting plan recognition on these snapshots so that event/state representations include both temporal and causal information. Instead TEXPLAN limits itself to generating a temporally organized report of key events. Subsequent sections on stories will examine generation from event/state models embodying causal relations.

Accessing the event/state network (which could be instantiated with LACE events and states or those of another domain), TEXPLAN's narrative plan operators select, order, and realize events to compose a report. Event verbalization is based both on event type (e.g., physical, psychological, and linguistic) and on time (i.e., values of E, R, and S). Only physical events are represented in LACE, although there are a variety of these (e.g., begin, fire, dispense). When the generator operates in this post-simulation, reporting mode, $R = S$ = time of occurrence of the event, and S = time of narration, so $E = R < S$, and in this case simple past tense is used. But if TEXPLAN narrates events as they occur then $E = R = S$ which suggests simple present tense. However, if TEXPLAN were to narrate the future plans associated with the goal stack of a particular agent then $S < E = R$ which would dictate simple future tense. These straightforward cases assume no shift in TF.

Event selection from the event/state is guided by a *saliency metric* which measures the importance of an event relative to other events in the simulation. Saliency is a function of:

1. the kind and amount of links associated with an event or state in the event/state network
2. the frequency of occurrence in the event/state network
3. domain-specific knowledge of importance

The first item concerns issues such as does the event achieve a main goal of a key agent in the simulation, and does it motivate, enable, or cause a number of events or states to occur (i.e., how many and what type of relation links does it have in the event/state network). The second item is simply the observation that frequent or commonplace events are boring. For example in LACE long-range radar constantly sweep, mobile SAMs[2] regularly reposition themselves, and active aircraft are always flying point-based ingress/egress routes. An example of the third item in the saliency function is that mission types have an order of interestingness (e.g., offensive air attack > SAM suppression > refueling > transportation). This is analogous to Kittredge's et al. [1986] weather reports which indicate warnings first and then `WINDS` > `CLOUD-COVER` > `PRECIPITATION` > `FOG&MIST` > `VISIBILITY`. These three items yield a numeric measure which can be used by TEXPLAN either to select those events that supersede some defined threshold or to sequence events in order of salience. There are other issues involved in saliency that are beyond the scope of this dissertation such as the inferability of events and states (i.e., if a target is bombed, it is destroyed, unless told otherwise), event and state persistence, and the representation of perceptual saliency (Conklin, 1983).

For example after a typical run of the LACE simulation in which some blue forces are attacking some red forces, TEXPLAN first selects salient events from an event/state network (e.g., Figure 5.9) using the salience metric detailed above and then uses the `narrate-report-temporally` plan operator of Figure 5.8 to produce the rather shallow text plan shown in Figure 5.10. As in the previous and subsequent chapters, a text plan is a communicative action decomposition which incorporates the structure and order underlying an English text and is constructed by reasoning about communicative acts which are formalized in individual plan operators. The temporally-ordered English report corresponding to the text plan in Figure 5.10 is shown in Figure 5.11. Like other text plans in this dissertation, it is realized as English text using the linguistic realization component detailed in Chapter 8. In Figure 5.11 temporal connectives (e.g., "then" "and then") give the impression of passage of time. They exploit the temporal focus to locate the event in the temporal context. Adverbs are also central to conveying event information.

[2]Surface to Air Missile.

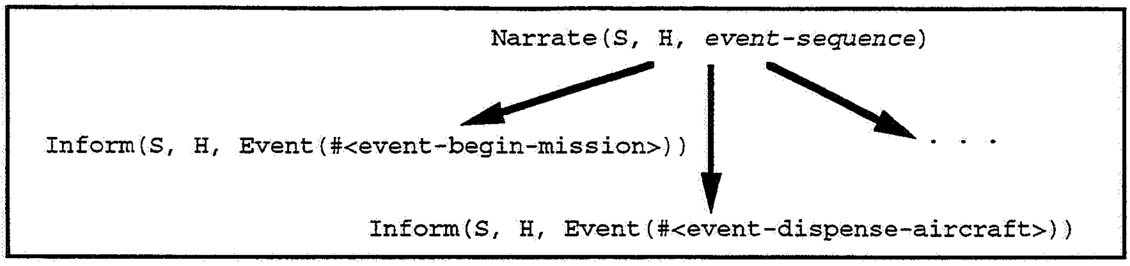



Figure 5.10 Text Plan for a Temporally Organized Report Narration

Offensive Counter Air Mission 100 began mission execution at 8:20::0 Tuesday December 2, 1987. The 902TFW-F-16c dispensed four aircraft for Counter Air Mission 100. Then eight minutes later Transportation Mission 250 began mission execution. 31MAC-C-130 dispensed one aircraft for Transportation Mission 250. Transportation Mission 250 began loading its cargo. Two minutes later Sam Suppression Mission 444 began mission execution. 126TFW-F-4g dispensed one aircraft for Sam Suppression Mission 444. One minute later Allstedt-B and Allstedt-C simultaneously fired a missile at Offensive Counter Air Mission 100. Six minutes later Transportation Mission 250 finished loading its cargo. Twenty seconds later Offensive Counter Air Mission 102 began mission execution. 901TFW-F-15e dispensed four aircraft for Offensive Counter Air Mission 102. And three minutes later Offensive Counter Air Mission 101 began mission execution. 900TFW-F-4c dispensed four aircraft for Offensive Counter Air Mission 101. And one minute later Sam Suppression Mission 444 aborted its mission. Then twenty seconds later Mobile-SAM1 fired a missile at Sam Suppression Mission 444. And three minutes later Haina-B fired a missile at Offensive Counter Air Mission 102. Then thirty seconds later Air Refueling Mission 100 began mission execution. 513TAW-SAC-Rotational-KC-135 dispensed one aircraft for Air Refueling Mission 100. Two minutes later Allstedt-B and Allstedt-C simultaneously fired a missile at Offensive Counter Air Mission 102. Then one minute later Offensive Counter Air Mission 102 aborted its mission. Two minutes later Allstedt-B again fired a missile at Offensive Counter Air Mission 102. One minute later Erfurt-A fired a missile at Offensive Counter Air Mission 102. One minute later Erfurt-D and Erfurt-F simultaneously fired a missile at Offensive Counter Air Mission 102. Twenty-Six seconds later Sam Suppression Mission 444 ended its mission. It generated its post-mission report. Then thirty-four seconds later Mobile-SAM2 fired a missile at Offensive Counter Air Mission 102. Two minutes later Offensive Counter Air Mission 101 bombed its target.

Figure 5.11 (Topically) Unfocused LACE report

TEXPLAN uses temporal adverbs (e.g., "three minutes later", "simultaneously", "before", "after", "when") to help convey a temporal model of the events. For example, events occurring at the same time are combined as in "One minute later Erfurt-A and Erfurt-D simultaneously fired a missile at Offensive Counter Air Mission 102." The adverb "again" is used when an event has already occurred, and thus functions as an anaphor. Other classes of adverbs can also enrich the event description. These include adverbs regarding manner (e.g., "deftly", "sadly"), rate ("slowly", "rapidly"), duration ("for twenty minutes"), location ("in the park"), frequency ("every ten minutes"), and numeration ("seventeen times"). Some adverbs (e.g., manner, rate, duration, locational) are *internal* to the event whereas others relate the current event to other events and therefore are *external* such as temporal adverbs (e.g., "simultaneously", "yesterday") or "anaphoric" adverbs (e.g., "again", "as before"). (see Winograd (1983, p. 540-2) for other adverb classes).

Despite these grammatical devices, the LACE report of Figure 5.11 remains difficult to comprehend. Part of the reason is that there is no static background or framework within which the events are to be interpreted. Also, comprehension is more difficult because the text follows a simple temporal sequence. This can be improved upon by organizing the propositions both temporally and topically. Because there are competing organizations, however, TEXPLAN must determine which organization of the events will be most effective. The `narrate-report-topically` plan operator of Figure 5.12 is selected because (1) relative to the number of events, there are few principal agents (i.e., missions), which enables topical grouping and (2) other plan operators are less appealing (for example, a top-level temporal organization would be confusing because there are many simultaneous events involving different agents). We distinguish here between the local focus of attention (Sidner, 1979) which is captured in the current discourse focus cache (introduced in the previous chapter and detailed in Section 8.4) and the topic or subject of multiple utterances, i.e., what they are "about". In our current example the various missions in the simulation are the topics. The decomposition of the `narrate-report-topically` plan operator in Figure 5.12 first sets the static background for the events using the `Introduce` plan operator shown in Figure 5.13 and described below. It then uses the previously detailed saliency metric (which considers the frequency, uniqueness, and importance of events) to order the various topics in decreasing order of importance (e.g., offensive air attack > SAM suppression > refueling > transportation). In LACE, given a list of events, the function `Topics` returns a list of the missions they describe in order of saliency.

The `introduce-setting` plan operator of Figure 5.13 sets the scene of the narrative by conveying the principal time, place, agents (i.e., characters), circumstances of the story, or any particularly significant or unusual entities that the hearer does not know about.

```
NAME            narrate-report-topically
HEADER          Narrate(S, H, events)

CONSTRAINTS     Topical-Sequence(events) ∧ ∀e ∈ events  Event(e)
PRECONDITIONS
  ESSENTIAL     ∀e ∈ events  KNOW-ABOUT(S, e)

EFFECTS         ∀topic ∈ topics(events) KNOW-ABOUT(H, topic)

DECOMPOSITION   Introduce(S, H, events)
                ∀topic ∈ order-According-to-Salience(Topics(events))
                 Narrate-Sequence(S, H, Events-with-Topic(events, topic))
```

Figure 5.12 `narrate-report-topically` Plan Operator

```
NAME              introduce-setting
HEADER            Introduce(S, H, entities)

CONSTRAINTS       ∀e ∈ entities  (Event(e) ∨ State(e)) ∧
                  ∃x | Main-Event(x, entities) ∨
                       Main-Time(x, entities) ∨
                       Main-Location(x, entities) ∨
                       Main-Agent(x, entities) ∨
                       Unknown-or-Unique-Entity(x, entities)
PRECONDITIONS
  ESSENTIAL       ∀e ∈ entities
                       KNOW-ABOUT(S, e) ∧ KNOW-ABOUT(S, x)
  DESIRABLE       ∃x | Main-Agent(x, entities) ∧ ¬ KNOW-ABOUT(H, x)

EFFECTS           ∀x | Main-Event(x, entities) ∨
                       Main-Time(x, entities) ∨
                       Main-Location(x, entities) ∨
                       Main-Agent(x, entities) ∨
                       Unknown-or-Unique-Entity(x, entities)
                  KNOW-ABOUT(H, x)

DECOMPOSITION     ∀x | Main-Event(x, entities) ∨
                       Main-Time(x, entities) ∨
                       Main-Location(x, entities) ∨
                       Main-Agent(x, entities) ∨
                       Unknown-or-Unique-Entity(x, entities)
                  optional(Describe(S, H, x))
```

Figure 5.13 `introduce-setting` Plan Operator

By referring to the description plans developed in the previous chapter, the decomposition of the setting plan of Figure 5.13 can describe the setting using a variety of means such as definition, characterization, division, comparison-contrast, and analogy. After events are grouped topically in order of salience, the `narrate-temporal-sequence` plan operator illustrated in Figure 5.14 exploits the temporal order of events to sequence them. In addition to the `narrate-temporal-sequence` plan operator, a `narrate-spatial-sequence` plan operator (not illustrated) performs a similar ordering function except in space rather than time. Similarly, a `narrate-topical-sequence` plan operator (not illustrated) orders events according to topic.

To contrast these plan operators with previous ones, we use these improved plan operators to structure the same events used to produce the simulation report of Figure 5.11. Thus using the above plan operators, TEXPLAN constructs the text plan shown in Figure 5.15 which topically and then, for each topic, temporally orders the events. The text plan of Figure 5.15 is a communicative action decomposition that achieves the top-level communicative act, `narrate`. The `Introduce` communicative act (arising from the use of the `introduce-setting` plan operator in the first part of the decomposition of the narrate-report

```
NAME              narrate-temporal-sequence
HEADER            Narrate-Sequence(S, H, events)

CONSTRAINTS       Temporal-Sequence(events) ∧ ∀e ∈ events  Event(e)
PRECONDITIONS
  ESSENTIAL       ∀e ∈ events  KNOW-ABOUT(S, e)

EFFECTS           ∀e ∈ events  KNOW-ABOUT(H, e)

DECOMPOSITION     ∀e ∈ select-and-order-temporally(events))
                     Inform(S, H, Event(e))
```

Figure 5.14 `narrate-temporal-sequence` Plan Operator

plan operator of Figure 5.12) sets the static background scene (i.e., the location, key characters, time, and so on) for the events that occur dynamically in the foreground. In LACE, information for the introduction is retrieved from the overall mission package, a frame-like structure which indicates the planned missions which are to be executed by LACE. This package contains the intended time of the missions, their location, their type, and so on. In this illustrative case the package (#<air-strike-10>) is the main event and is described using two rhetorical predicates: logical definition (which indicates the genus and differentia of the package) and constituency (which indicates the subparts, in this case missions). This descriptive structure is captured in the branch of the text plan in Figure 5.15 which introduces the air strike by logically defining it and then indicating its constituent missions.

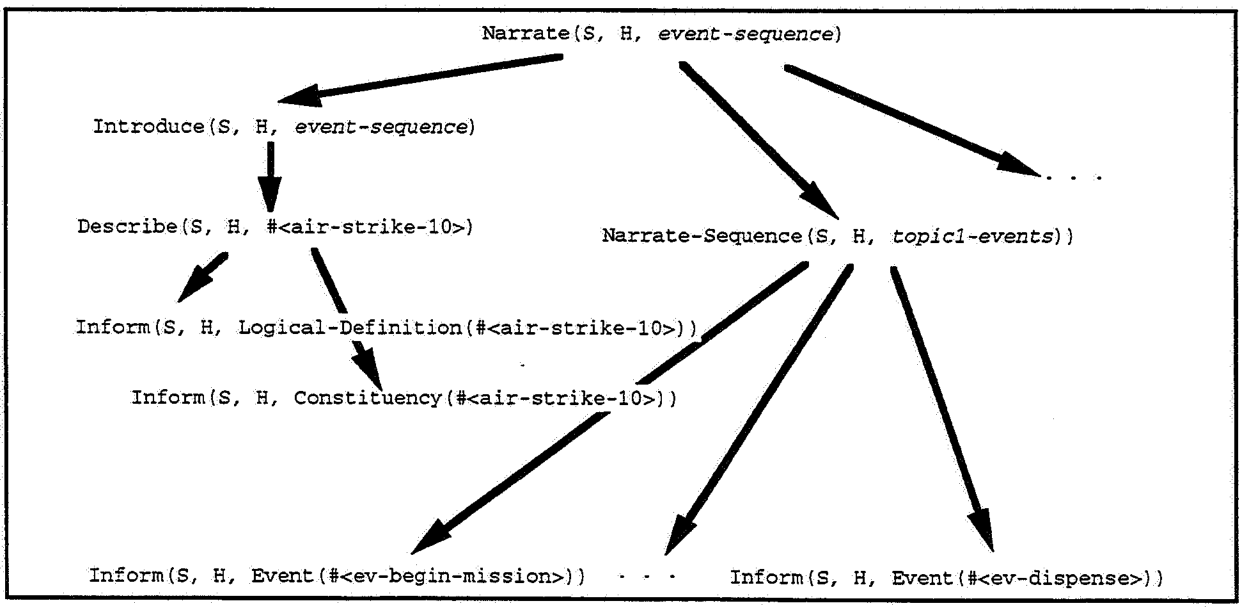



Figure 5.15 Text Plan for Topically/Temporally Ordered Report

```
Air-strike 10 was an attack against Delta airfield on Tuesday December 2, 1987. Air-strike 10
included three Offensive Counter Air Missions (OCA100, OCA101, and OCA102), one SAM Suppression
Mission (SSM444), one Transportation Mission (TRANS100), and one air refueling mission (RFL100).

Offensive Counter Air Mission 100 began mission execution at 8:20::0 Tuesday December 2, 1987.
902TFW-F-16c dispensed four aircraft for Offensive Counter Air Mission 100. Eight minutes later
Offensive Counter Air Mission 100 began flying its ingress route. Three minutes later Allstedt-B and
Allstedt-C simultaneously fired a missile at Offensive Counter Air Mission 100. And fifty-nine
seconds later Offensive Counter Air Mission 100 was ordered to abort its mission. One second later
Allstedt-C and Allstedt-B again simultaneously fired a missile at Offensive Counter Air Mission 100.
Two minutes later Allstedt-B again fired a missile at Offensive Counter Air Mission 100. Then one
minute later Erfurt-A fired a missile at Offensive Counter Air Mission 100. Then two minutes later
Haina-B fired a missile at Offensive Counter Air Mission 100. Seven minutes later Offensive Counter
Air Mission 100 ended its mission. It generated its post-mission report.

In the meantime SAM Suppression Mission 444 began mission execution at 8:30::0 Tuesday December 2,
1987. 126TFW-F-4g dispensed one aircraft for SAM Suppression Mission 444.
SAM Suppression Mission 444 began flying its ingress route. Thirteen minutes later Mobile-SAM1 fired
a missile at SAM Suppression Mission 444. Then fifty-nine seconds later SAM Suppression Mission 444
was ordered to abort its mission. And then one second later Mobile-SAM2 fired a missile at SAM
Suppression Mission 444. One minute later Mobile-SAM2 and Mobile-SAM1 simultaneously fired a missile
at SAM Suppression Mission 444.

In the meantime Offensive Counter Air Mission 101 began mission execution at 8:41::40 Tuesday
December 2, 1987. 900TFW-F-4c dispensed four aircraft for Offensive Counter Air Mission 101.
Then seven minutes later Offensive Counter Air Mission 101 began flying its ingress route.
Then ten minutes later it bombed its target. It began flying its egress route.
Thirty-Six minutes later it ended its mission. It generated its post-mission report.

Meanwhile Transportation Mission 250 began mission execution at 8:28::0 Tuesday December 2, 1987.
31MAC-C-130 dispensed one aircraft for Transportation Mission 250. Transportation Mission 250 began
loading its cargo. Ten minutes later it finished loading its cargo.
It began flying its ingress route. Then sixty-four minutes later it off-loaded its cargo.

Meanwhile Offensive Counter Air Mission 102 began mission execution at 8:38::20 Tuesday December 2,
1987. 901TFW-F-15e dispensed four aircraft for Offensive Counter Air Mission 102. Offensive Counter
Air Mission 102 began flying its ingress route. Six minutes later Haina-B fired a missile at
Offensive Counter Air Mission 102. Four minutes later Allstedt-C fired a missile at Offensive
Counter Air Mission 102. One minute later Allstedt-B and Allstedt-C simultaneously fired a missile
at Offensive Counter Air Mission 102.

Meanwhile air refueling mission 100 began mission execution at 8:46::40 Tuesday December 2, 1987.
513TAW-SAC-Rotational-KC-135 dispensed one aircraft for Air Refueling Mission 100. Air Refueling
Mission 100 began flying its ingress route. Eighteen minutes later it started its refueling orbit.
Twenty-Six minutes later it ended its refueling orbit. It began flying its egress route.
```

Figure 5.16 Temporally and Topically Focused LACE Report

Figure 5.16 illustrates the improved output report corresponding to the text plan of Figure 5.15, in which topics are ordered according to importance (i.e., offensive air > SAM suppression > refuel > transport) and events within each topic are ordered temporally. But despite the improvement obtained by the topical structure and temporal order of content, the report remains simply a recounting of salient events. It is not a story, even though it has a setting. It fails, for example, to indicate causal relations among events and states. In short, reports have no plot. In contrast, the next section discusses how stories revolve around some causal/temporal plot.

## 5.6 Stories in TEXPLAN

Like a report, *stories* typically recount events in time. They differ, however, in that stories have a plot, a series of causally connected events. In stories, events are narrated with respect to their cause and effect interdependencies. That is if John kisses Mary then this "kissing" event may cause a smiling event because Mary is in a state of elation because of John's action. For example, consider the passage below where Bill, a college student, tells the story about asking his father to borrow the car for the first time (Brown and Zoellner, 1968, p. 619):

> As I made my request, Dad's face assumed an expression of horrified astonishment; it was obvious that he regarded my asking for the car to be in the same category with asking for matches to burn down the house. He didn't say anything. He rattled his newspaper. He coughed. He uncrossed his legs and then recrossed them. He elaborately studied the toe of his right shoe. He looked out the window.
>
> Finally he said, "Let's see that license again."
>
> I got out my wallet, extracted the license, and handed it to him. He took it gingerly by the corner, as if it might contaminate him. With his head tilted back, he glared at it through his bifocals; he apparently thought it was counterfeit. With a gesture redolent of disapproval, he handed it back to me. He looked out the window some more, his fingers drumming on the arm of the chair.
>
> "Okay," he said. "Okay. And God help the insurance company."

The story concerns the son's desire to use his father's car, motivated by some as yet unexplained (and therefore mysterious) reason (e.g., an important date). This leads him to attempt to convince his father to give him the car. The narration is connected both by explicit dialogue between the characters as well as implicit emotions (e.g., the father's fear that son will wreck the car). This prose is sophisticated, even using humor (e.g., borrowing the car is compared to arson; the father treats the license as if contaminated or counterfeit). Notice the consistent use of past tense to describe events that occurred prior to speech time. Note also the inclusion of details that, while not central to the action, lend insight into the psychological state of the characters (e.g., "He didn't say anything." "He elaborately studied the toe of his right shoe."). Emotions of the characters are also expressed through word choice (e.g., horrified astonishment, glared) and syntactic choice (e.g., the simple, active sentences that describe the father at the end of the first paragraph reflect his agitated reaction).

The event and state relations of the above story are presented graphically in what I term a *story diagram*, shown in Figure 5.18 with a key in Figure 5.17. A story diagram is a network indicating the enablement, causation, motivation, purpose, and temporal relationships between events and states (in

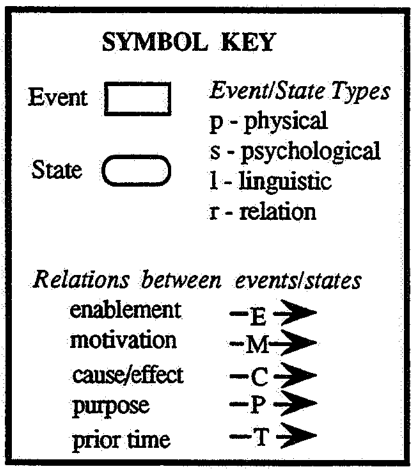



Figure 5.17 Key to Story Diagrams

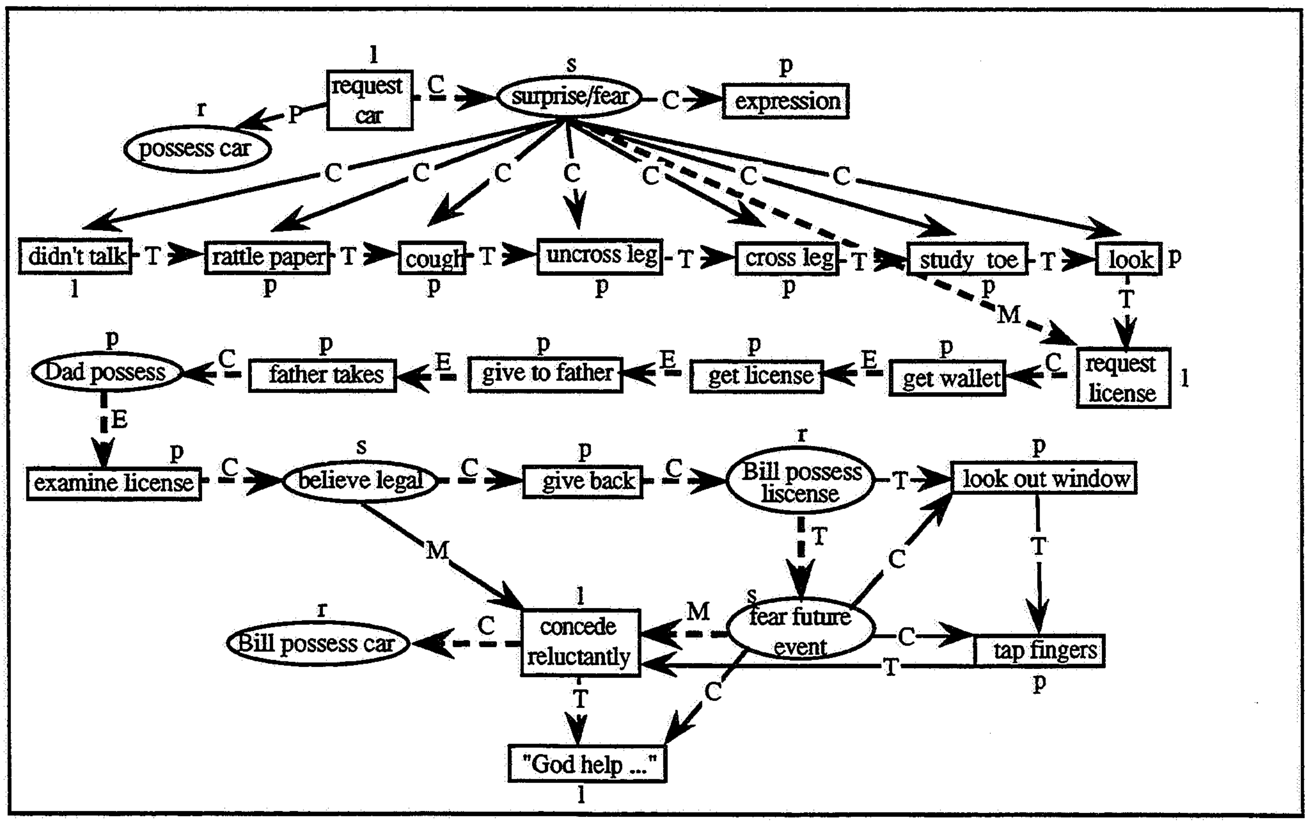



Figure 5.18 Story Diagram of Bill's Story

essence a graphical view of an event/state structure). This includes the plans and goals of individual agents. For expository purposes, only temporal precedence links are illustrated although other temporal relations (e.g., simultaneity, overlap, contiguousness, etc.) are represented in the event/state network underlying this story diagram.

From this network a critical path of events and states can be selected which forms the skeleton of the story. This path is indicated by dashed arrows in the story diagrams given in this chapter. Currently, the critical path is marked in the input to TEXPLAN because path selection has not yet been implemented. However, an algorithm has been designed that is not unlike that in Paris and McKeown (1986) and Paris (1987ab, 1988) (hereafter referred to as Paris). In Paris' so-called *process trace*, a *main path* is selected through causal relations in order to describe the process carried out by a physical object (e.g., a telephone). The system starts with a frame-based representation that indicates all the a priori relationships between entities and events, and then selects the path based on (1) the link type (either control (cause-effect, enablement, or interruptions), or analogical (equivalent, corresponds-to)[3]) and (2) the structure and length of links off the main path (e.g., side chains, isolated side links).

There are several differences between Paris' system, TAILOR, and TEXPLAN. First, TAILOR was concerned only with describing processes carried out by physical objects, as opposed to the situations and events found in stories. Not surprisingly, the intentionality-based nature of characters in stories leads to a richer set of relationships in TEXPLAN for events and states (e.g., it includes motivation and purpose links). Second, the event/state network used in TEXPLAN to represent narrative content incorporates a richer temporal representation which indicates not only temporal precedence but overlap, simultaneity, etc. Third, while TAILOR's process trace does use rhetorical predicates (identification, attributive, constituency, and cause-effect) as a level of representation independent of the application content (i.e., events, states, relationships), there is no explicit representation of hierarchical text structure above the utterance level as in say a story grammar. While McKeown's constituency schema encodes standard patterns found in object descriptions, Paris' process trace follows the underlying representation of individual processes to generate a description rather than reasoning about the hierarchical structure of the text. In addition, the process trace does not indicate what effect the selected content has on the reader's cognitive state (i.e., their knowledge, beliefs, or desires). As a consequence, it is not possible to use it as a general mechanism for narrative generation because it cannot capture how different effects such as suspense or mystery can be produced by restructuring or reordering the elements in a text. A further complication is that a start state and a goal state are crucial to selecting the main path in Paris's process trace, and yet stories may have multiple start and goal states (not to mention conflicting underlying goals) as a consequence of the interaction of multiple, autonomous agents. But although the process trace has these weaknesses, it should be noted that Paris's principal aim was to tailor a description to a user's level of expertise. She

[3] Another link type was temporal, although its use was not illustrated.

succeeded by generating functional descriptions using her process trace when users had little knowledge of an entity and by switching to structural descriptions using McKeown's (1985) constituency schema when they were familiar with it (because they could mentally "fill in" the causal links).

Because more types of link are represented in my story diagrams, the path selection algorithm is more complex. Path selection is a function of link types (e.g., enablement, motivation, causation, and purpose), temporal relations between events and states in the network (e.g., preceding, overlapping, succeeding, co-occurring, etc.), link topology (e.g., sides chains, isolated side links), entity importance (e.g., object, event, state), and the interestingness of entities. Interestingness has been a topic of controversy in the debate over story grammars (Wilensky, 1983), primarily because it is subjective and context dependent. While the computation of interestingness in TEXPLAN is in part domain and context dependent (e.g., mission types have a saliency ordering, in part determined by the current goal), it also relies on general principles. These include frequency of occurrence, prototypicality (i.e., does the event typically occur in the context of surrounding events as in a Schankian script), inferability (which relies on the model of the hearer), and the density of causal relations with other events and states (e.g., is the event or state intransitive or does it effect other agents or the world).

Because of these criteria, there is no one best path through the network, but rather a number of paths with varying degrees of goodness. Of course we would like to convey all important events. At any given node, the algorithm selecting the main path will prefer connections in the following order: causation, enablement, motivation, purpose, temporal precedence, temporal overlap, and temporal simultaneity (note that the first four imply the fifth). These preferences are tempered by the topology of the path (i.e., is it just a side link?), and interestingness can override these decisions as in the choice in Figure 5.18 of the event following the believe-legal state in the main path. Given any node (event or state) in the network, the path selection algorithm chooses the path through the remaining events and states in time which has the highest cumulative rating (using the above criteria). The highest rated path overall becomes the plot chain. The narrative plan operators later embellish this skeleton chain with related events and states.

Recall the story about Bill and his father. While Bill's account is rich, the underlying structure follows the temporal order and causal connection of states and events. After each event is introduced, a number of pieces of information can be added. For example the narrator can indicate the manner of the event (e.g., "took it gingerly"), describe any principal agents in the event (e.g., "My Dad's face"), the motivation of their actions, what enabled an event (taking out the wallet enables the boy to extract his license), the cause of an event (e.g., the boy's request horrifies the father), its purpose (e.g., the father asks to see the license to verify its authenticity), and any physical or psychological effects of certain events (e.g., horror). Furthermore, the narrator can provide his or her own interpretation of the events.

Several researchers have examined the causal chains underlying stories (e.g., Schank, 1975), focusing on the "real" order of situations and events, i.e., the content of a story as opposed to its discourse structure or form. In an attempt to abstract away from the underlying events and states of a story, Lehnert

(1981) suggested a number of "plot units" (e.g., problem resolution by intentional means, trade, and honored request) which were configurations of "positive" events, "negative" events, and "neutral" emotional states. Lehnert's aim, however, was narrative summarization as opposed to generation. Dyer (1981) argued that events need to be interpreted from multiple perspectives. For example, in one narrative processed by Dyer's system, BORIS, two characters agree to meet at lunch because they haven't seen each other in years and because one wants the other, a lawyer, to represent him in a divorce case. Their meeting can be viewed as a part of a restaurant script, in terms of the theme of a suspended friendship being renewed, in terms of the plans or goals the meeting satisfies, or in terms of the roles of the characters, i.e., a client meeting to discuss a legal case. After interpreting events, Dyer's system would answer questions about them.

In contrast to this previous work which focuses on the interpretation and representation of events underlying narratives, TEXPLAN reasons about story content (events and states), using narrative plan operators which are independent of that content, to generate the actual discourse structure used to present a narrative. The remainder of this section outlines the plan operators in TEXPLAN which narrate a story given an event/state structure. TEXPLAN's strategy for story narration mirrors the structure of Bill's story. To compose a story from a given event/state network, TEXPLAN first describes any key settings, key characters, significant events, and important changes of state that result from events. For example, if all the events take place on the same day and in the same location then this should be initially described. The key characters should also be introduced. A key character is one frequently involved in events, or one involved in the most important events in the sequence, of significant social importance, and so on. It is equally important to introduce particularly significant or unusual entities that the hearer does not know about so that they will be able to understand the event sequence. For example if the story is about a boy's favorite fly-cast fishing pole breaking on his first visit to his local pond, then its beauty and key features should be detailed. Once this background scene has been set, the story proceeds by expressing events in causal sequence, that is the events on the main path.

Figure 5.19 illustrates the `narrate-story` plan operator, the top-level operator used to present a story. The constraint on the use of the plan operator is that there must be a causal path through the event/state structure. The intended effect of its application is that the hearer will know about each of the events and states in the principal causal path (which can be used to summarize the story). The intent is also that the hearer will know their relation to one another.

```
NAME              narrate-story
HEADER            Narrate(S, H, events+states)

CONSTRAINTS       ∀x ∈ events+states (Event(x) ∨ State(x)) ∧
                  Causal-Path(events+states)
PRECONDITIONS
  ESSENTIAL       ∀x ∈ events+states KNOW-ABOUT(S, x)

EFFECTS           ∀z ∈ events+states KNOW-ABOUT(H, z)
                  ∀x,y ∈ events+states KNOW-ABOUT(H, Relation(x,y))

DECOMPOSITION     Introduce(S, H, events+states)
                  ∀x ∈ chain
                     Narrate-Event-or-State(S, H, x, chain)

WHERE             chain = select-causal-sequence(events+states)
```

Figure 5.19 `narrate-story` Plan Operator for Causal Sequence Narration

The decomposition of the main `narrate-story` plan operator of Figure 5.19 first sets the scene (principal time, location, characters, etc.) for the story using the `introduce-setting` plan operator defined in the previous section. The next step in the decomposition then selects a causally connected sequence of events. Each event or state is then detailed in turn using the `Narrate-Event-or-State` communicative action. There are two plan operators whose headers match the `Narrate-Event-or-State` communicative action. The first is for event narration (Figure 5.20) and the second for state narration (Figure 5.21).

The `narrate-event` plan operator is illustrated in Figure in 5.20. This plan operator reasons about the event and state network previously defined. The decomposition of the operator allows for a variety of extensions on the simple event description. The operator can indicate the enablement, cause, or motivation of an event. Similarly, it can indicate an event's consequences including physical effects, cognitive effects (e.g., changing the knowledge, beliefs, and desires of agents in a scene), and relational effects (e.g., ownership). If the event involves an agent unknown to the hearer, this can be indicated. Finally, the speaker (i.e., narrator) can interpret the event using a model of his knowledge, beliefs, desires, etc. (e.g., is this event positive/negative, important/trivial, surprising/expected, indicative of future events, and so on).

In contrast to event narration, the `narrate-state` plan operator is shown in Figure 5.21. It is analogous to that for events except states have neither purposes nor motivations because they lack intentions. States can, however, motivate agent's actions.

```
NAME              narrate-event
HEADER            Narrate-Event-or-State(S, H, e, chain)

CONSTRAINTS       Event(e)
PRECONDITIONS
  ESSENTIAL       KNOW-ABOUT(S, e)
  DESIRABLE       ¬ KNOW-ABOUT(H, e)

EFFECTS           KNOW-ABOUT(H, e) ∧
                  ∀x | Motivation(Agent(x), e)
                     KNOW(H, Motivation(Agent(x), e)) ∧
                  ∀a | (Agent(a,e) ∧¬ KNOW-ABOUT(H, a))
                      KNOW-ABOUT(H, a) ∧
                  ∀p | Purpose(e,p)
                      KNOW-ABOUT(H, Purpose(e,p)) ∧
                  ∀x | Narrator-view(e, x) ∧
                     KNOW(H, Narrator-view(e, x))

DECOMPOSITION

   optional(Tell-Enablement/Causation(S, H, e, chain))
   optional(Inform(S, H, Motivation(Agent(e),e)))

   Inform(S, H, Event(e))
   optional(∀a | Agent(a,e) ∧¬ KNOW-ABOUT(H, a)
                 Describe(S, H, Agent(e)))

   optional(Tell-Consequences(S, H, e, chain))

   optional(∃p Purpose(e,p) Inform(S, H, Purpose(e, p)))
   optional(Inform(S, H, Narrator-Interpretation(e)))
```

Figure 5.20 `narrate-event` Plan Operator

Figure 5.22 illustrates the `tell-enablement/causation` plan operator which informs the hearer of the enablement and/or causation of the event or state. There are no preconditions for the plan operator. The opposite of the enablement and causation of an event or state are its consequences. The `tell-consequences` plan operator in Figure 5.23 informs the hearer of the effects of the event or state provided it is not a member of the main path or chain, in which case it will already be narrated.

The plan operators illustrated in Figures 5.19 to 5.23 characterize the abstract story generation knowledge used by TEXPLAN to compose stories from a causally-connected structure of events and states. To tell a story, the plans first set the static background scene of time, place, and main characters. Next the plans narrate dynamic events and states in the foreground, detailing their causes/enablements as well as consequences were appropriate. Because the explicit effects of each communicative act are detailed in the plan operators, the system can build a model of what it expects its effects are on the user's knowledge of events, states, and their relations.

```
NAME              narrate-state
HEADER            Narrate-Event-or-State(S, H, state, chain)

CONSTRAINTS       State(state)
PRECONDITIONS
  ESSENTIAL       KNOW-ABOUT(S, state)
  DESIRABLE       ¬ KNOW-ABOUT(H, state)

EFFECTS           KNOW-ABOUT(H, state) ∧
                  ∀x | Motivation(Agent(state), x)
                     KNOW(H, Motivation(Agent(state), x))
                  ∀a | (Agent(a,state) ∧¬ KNOW-ABOUT(H, a))
                     KNOW-ABOUT(H, a)
                  ∀x KNOW(H, Narrator-Interpretation(state, x))
DECOMPOSITION

   optional(Tell-Enablement/Causation(S, H, state, chain))

   Inform(S, H, State(state))
   optional(∀a | Agent(a,state) ∧¬ KNOW-ABOUT(H, a)
                 Describe(S, H, Agent(state)))

   optional(Tell-Consequences(S, H, state))
   optional(Inform(S, H, Motivation(Agent(state), x)))

   ∀x optional(Inform(S, H, Narrator-Interpretation(state, x)))
```

Figure 5.21 `narrate-state` Plan Operator

```
NAME              tell-enablement/causation-of-event-or-state
HEADER            Tell-Enablement/Causation(S, H, e, chain)

CONSTRAINTS       (Event(e) ∨ State(e)) ∧
                  (∃x | (Enablement(x, e) ∧¬ Member(x, chain)) ∨
                   ∃x | (Cause(x, e) ∧¬ Member(x, chain))

EFFECTS           KNOW-ABOUT(H, e) ∧
                  ∀x | (Enablement(x, e) ∧¬ Member(x, chain))
                      KNOW(H, Enablement(x, e))
                  ∀x | (Cause(x,e) ∧¬ Member(x, chain))
                      KNOW(H, Cause(x,e)))

DECOMPOSITION ∀x | Enablement(x, e) ∧¬ Member(x, chain)
                   optional(Inform(S, H, Enablement(x, e)))
              ∀x | Cause(x,e) ∧¬ Member(x, chain)
                   optional(Inform(S, H, Cause(x, e)))
```

Figure 5.22 `tell-enablement/causation` Plan Operator

```
NAME              tell-consequences-of-event-or-state
HEADER            Tell-consequences(S, H, e, chain)

CONSTRAINTS       (Event(e) ∨ State(e)) ∧
                  ∃x | Effect(e, x) ∧ ¬ Member(x, chain)
PRECONDITIONS
  ESSENTIAL       KNOW-ABOUT(S, Effect(e,x))
  DESIRABLE       ¬ KNOW-ABOUT(H, Effect(e,x))

EFFECTS           ∀x | Effect(e, x) ∧ ¬ Member(x, chain)
                     KNOW(H, Effect(e, x))

DECOMPOSITION     ∀x | Effect(e, x) ∧ ¬ Member(x, chain)
                    optional(Inform(S, H, Effect(e, x)))
```

Figure 5.23 `tell-consequences` Plan Operator

### 5.6.1 LACE Domain Stories

Unfortunately, only physical events occur in LACE. Therefore, in order to test the above story plan operators, an event/state knowledge base for a battle scenario was developed that included physical, linguistic, and psychological events as well as physical, psychological, and relational states. This event/state structure is presented as a story diagram in Figure 5.24. The scenario concerns an insubordinate pilot who, after nearly being killed by an enemy surface-to-air (SAM) missile, defies orders and proceeds to accomplish his or her mission. Events and states are indicated using the same symbols (see key Figure 5.17) as the story about Bill borrowing his father's car.

Using the narrative story plan operators of Figures 5.19 to 5.23, TEXPLAN follows the main path indicated by the dotted lines in Figure 5.24 so as to organize the information in the story diagram (an event/state structure) to obtain a text plan that structures and orders the information for output presentation. Figure 5.25 shows the part of this text plan concerning the pilot's initial request to re-attack the target. While the text plan of Figure 5.25 has been produced and was in fact linearized as a list of surface speech acts with corresponding propositional content, it was not linguistically realized. Nevertheless, because of the structure of the text plan and the rich ontology underlying the propositional content (including, for example, the distinction between physical, psychological, and linguistic events), a significant amount of information is available to guide the realization component.

If the text plan of Figure 5.25 were realized as English, it would yield the story shown in Figure 5.26. The main path also includes the side link about the SAM firing on the aircraft which runs across the top of Figure 5.24 (this is indicated parenthetically in the English text of the story in Figure 5.25).

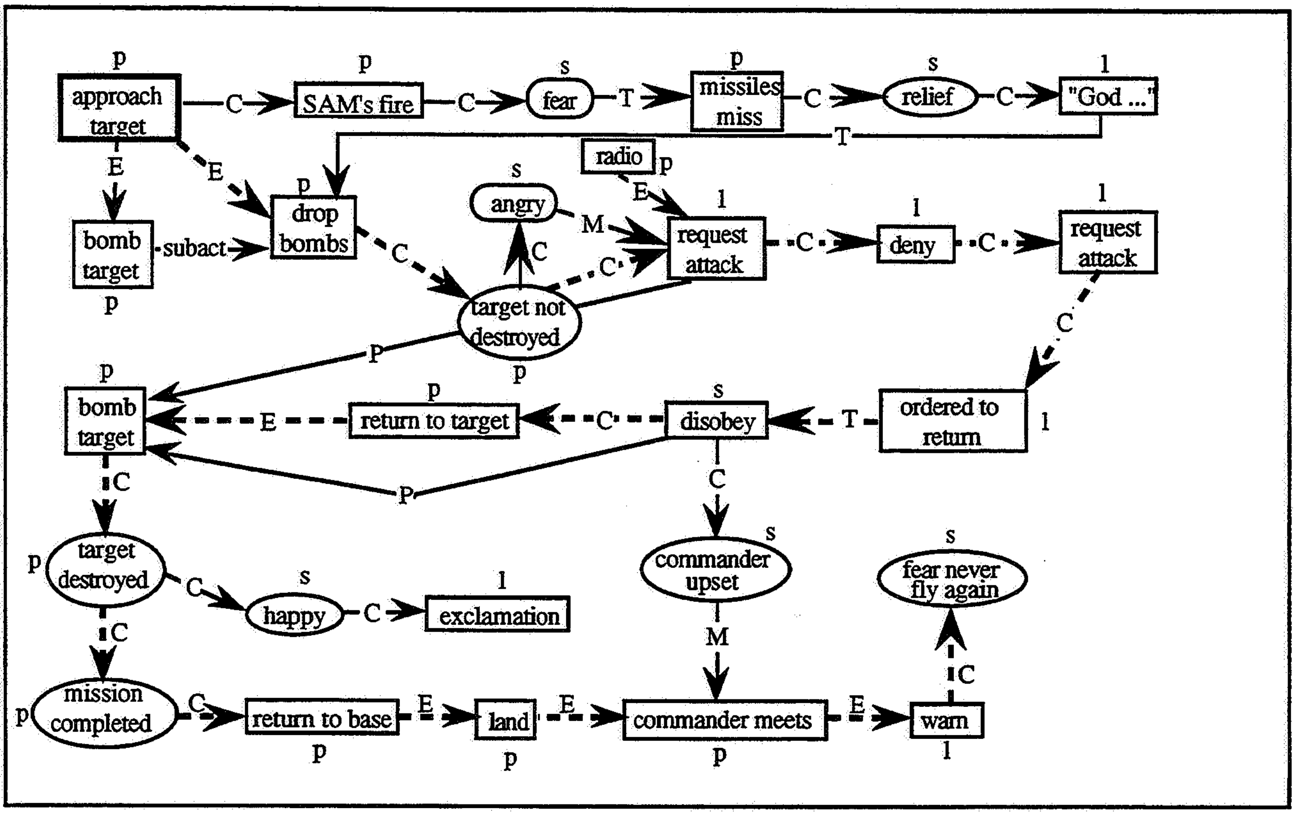



Figure 5.24 Story Diagram of Pilot's Story

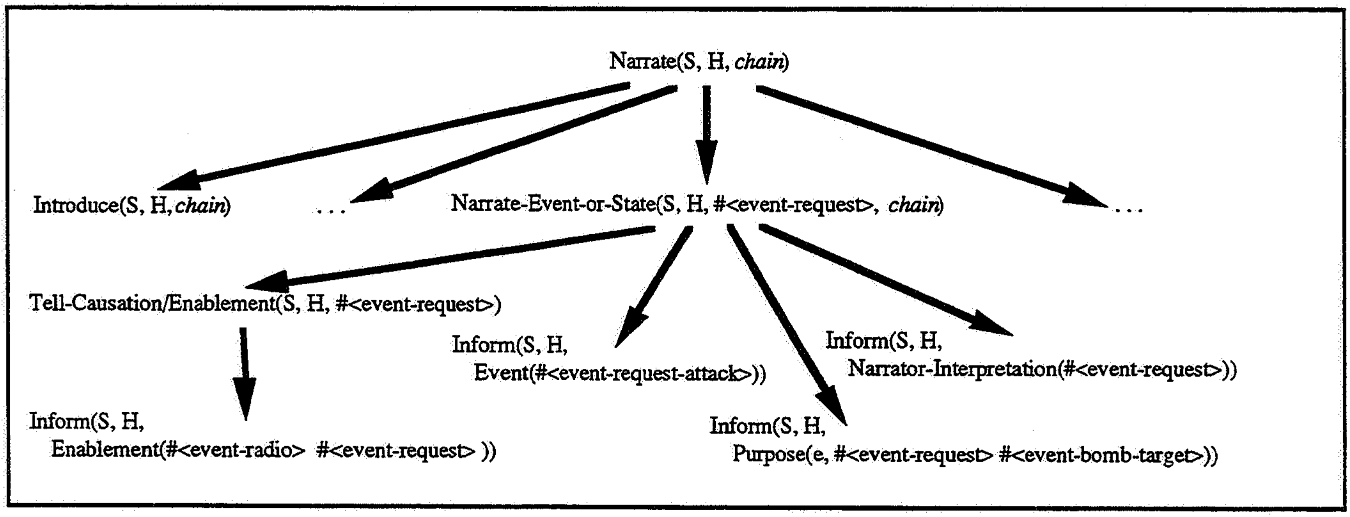



Figure 5.25 Portion of Text Plan of Pilot's Story for #<event-request>

My mission early Sunday morning was to bomb the airfield at Erfurt.

I approached the target in order to bomb it.

(side link)
Suddenly, two enemy surface-to-air missile sites fired missiles at me. I feared for my life. The missiles missed my jet and exploded 15 yards from my canopy. I was relieved. "God that was close" I said.

(continue main link)
I dropped my bombs. They didn't destroy the target. I was angry.

I radioed the control tower at my home base. "Can I try another attack?" I asked in order to bomb the target. This was expected of a brave pilot.

"No" shouted the air traffic controller. He sounded like he was under duress.

"I want to try again." I told him.

"Commander Willis orders you to return to base."

"I'm going to attack again." I said defiantly.

I returned to the target and bombed it. It was destroyed. I was happy. "Yahoo" I yelled.

My mission was accomplished. I returned to base. I landed.

Commander Willis was upset with me. He met me at the flight line. He didn't look happy.

"Never defy orders!" he shouted. I feared I would never fly again.

Figure 5.26 Pilot's Story

This story is distinct from previously generated stories in a number of ways. First, the generator reasons, as narrators appear to, at an abstract level about stories in general. The text plan indicates the relationships of events and states (e.g., cause, enablement, etc.), which in turn motives the order and structure (i.e., subordination) of elements in the text. Second, because text plans represent narrative structure independent of content, TEXPLAN's plan operators can characterize such phenomena as the narrator's interpretation of the unfolding events. For example the utterances "He sounded like he was under duress." and "This was expected of a brave pilot." in the story in Figure 5.26 are metacomments by the narrator that indicate his or her interpretation of the event or state. The objective view (which the underlying events and states represent), may simply indicate that the manner of the air traffic controller's reply was rapid or that the quality of his voice was high-pitched. In the implementation, the particular reaction of the narrator to an event or state is hand-encoded, although the narrator's interpretation could in principle be computed. For example, the narrator could interpret an event taking into account a variety of factors including his or her relationship to and affection for the agents in the story as well as his or her own goals (e.g., make the protagonist look good).

The richness of stories often arises because of the characteristics of the audience, author, or cultural context. For example a moralist author may highlight that a fictional wrongdoer pays for his/her sins whereas a cynical narrator might stress unethical actions that go unpunished. To do this one needs richer pragmatic models of the speaker and the hearer, their knowledge and beliefs, and their relationships to each other and to the content of the story. This in turn should effect the selection of content and its realization onto language (cf. Hovy, 1987).

Unlike the work described in the previous section on report generation in which many reports were actually produced, only one story was actually planned. However, the descriptive plan operators that introduce stories were illustrated in the previous section on report generation as well as in other domains in the previous chapter on description. Furthermore, the narrative plan operators used to produce stories (e.g., narrate-event-or-state) are domain independent: they refer only to the formal ontology of events, states, and their causal relations defined in previous sections.

## 5.7 Biographies in TEXPLAN

Whereas stories revolve around a casually-connected sequence of states and events and can involve multiple agents, *biographies* in TEXPLAN convey events and states over the lifetime of one individual. In general, biography includes eulogy and obituary as well as biography and autobiography. As with other forms of narration, only the most significant events and key people involved in the life of an individual are conveyed. Consider:

> Thomas Stearns (T. S.) Eliot (1888-1965) was born in St. Louis, Missouri. He was educated at Harvard University where he received a master's degree in philosophy in 1910, the same year in which he began writing "The Love Song of J. Alfred Prufrock." He finished the poem the following year during a visit to Germany, but it wasn't published until 1915 with the help of Ezra Pound [one of the creators of the Imagist Movement in poetry]. Eliot returned to Harvard to teach for a few years, but in 1914 he went back to Germany. World War I broke out and Eliot could not return to America. He went to England to work, married, and became a British subject in 1926.[4]

Like this biography, the `narrate-biography` plan operator in Figure 5.27 takes a number of situations (i.e., a number of events and states) that involve or concern some specified agent and conveys those in temporal sequence. As in a story, these events can be connected (e.g., by cause) to make the resulting narration flow. Note that the first portion of the decomposition optionally describes the agent. Often this is implicit in human biographies. Nevertheless, we can imagine the above text opening "T. S. Eliot (1888-1965) was a great American poet. He was born in ..."

[4] *The United States in Literature*, Glenview, Illinois: Scott, Foresman, and Company, 1976, p. 163.

```
NAME              narrate-biography
HEADER            Narrate(S, H, situations, agent)

CONSTRAINTS       ∃x | (Event(x) ∨ State(x)) ∧ Agent(agent, x)
PRECONDITIONS
  ESSENTIAL       KNOW-ABOUT(S, agent) ∧
                  ∃x | (Event(x) ∨ State(x)) ∧
                        Agent(agent, x) ∧ KNOW-ABOUT(S, x)
  DESIRABLE       KNOW-ABOUT(H, agent)

EFFECTS           ∀e ∈ situations
                      KNOW-ABOUT(H, e) ∧
                      KNOW-ABOUT(H, agent)

DECOMPOSITION     optional(Describe(S, H, agent))
                  ∀s | s ∈ ordered-situations ∧ Agent(agent, s)
                      Narrate-Event-or-State(S, H, s)

WHERE             ordered-situations = Select-and-order(situations)
```

Figure 5.27 `narrate-biography` Plan Operator

The biography plan operator can apply to real people (as in T. S. Eliot), simulated agents (e.g., the pilot in the previous story), non-human agents (e.g., a mission), or things (e.g., a biography of what happened to a ship). For example the plan operator in Figure 5.27 was tested by producing biographies of non-human agents (e.g., missions) in the LACE simulation because LACE does not model human agents. This was sufficient to illustrate the general principles of narration in TEXPLAN as agents in LACE are active, have effects on other agents and entities, and thus have an associated history. For example, if the discourse goal `KNOW-ABOUT(H, agent)` is posted to TEXPLAN after a typical run of the simulation, this matches the effect of the plan operator in Figure 5.27 and is selected by the text planner. The decomposition of the biography plan operator first describes *agent* and then describes events in which it played a role. For example, the discourse goal `KNOW-ABOUT(H, OCA101)` was posted to the text planner to produce the biography shown in Figure 5.28 in which the first utterance describes `OCA101` and the remaining ones indicate the significant events or states in which it was involved.

```
Offensive Counter Air Mission 101 was an air strike against Delta airfield.  It began
mission execution at 8:41::40 Tuesday December 2, 1987.  It received four aircraft from
the 900TFW-F-4c.  Seven minutes later it was flying its ingress route.  Then ten
minutes later it bombed its target.  It began flying its egress route.  Thirty-six
minutes later it ended its mission.  It generated its post-mission report.
```

Figure 5.28 Biography of Agent in LACE Simulation

Unlike report narration, biographies do not have to follow strict temporal sequencing. For example, they can follow some sequence of important issues or problems faced by the individual. Also, as the example in Figure 5.28 demonstrates, biographies do not have to involve animate or human objects. They simply must concern agents involved in events or states. Biography can be about inhuman, animate objects (e.g., animals) as well as inanimate objects (e.g., organizations, societies, or theaters).

## 5.8 Narrative Techniques: Surprise, Suspense, and Mystery

Because plan operators capture the intended effects of the text on the reader, they can seek to capture more sophisticated effects such as the creation of *surprise*, *suspense*, or *mystery* in a story. The author can surprise the reader by striking them with unexpected, unusual, or strange events, states, or characters. Suspense, in contrast, occurs when the hearer is placed in a state of expectation. The reader can fear some expected event (e.g., as in waiting for a killer to strike) or hope for it to happen (e.g., as in buying a lottery ticket and listening for the results). The author can build suspense by foreshadowing future events or putting characters in conflict. Finally, mystery occurs when significant events happen with no apparent justification (i.e., there is no known enablement, motivation, causation, or purpose for the event). By not telling the reader important information (as in a detective story) s/he is puzzled by the events because s/he cannot fit them into any plausible mental model. Consider:

> How was the prisoner able to escape?
> What motivated the butcher to leave his shop early?
> What caused the clock to stop ticking?
> What effect did the locked door have on the murderer?

Flashback and flash-forward can also induce mystery. When an author suddenly jumps to previous or future times, the reader does not know what sequence or chain of events connects the story and its characters (in the present time) to these past or future circumstances.

These techniques can be formalized as plan operators. For example, the mystery plan operator in Figure 5.29 intentionally does not convey some known enablement, motivation, cause, or purpose for some event. The constraints on the plan operator indicate that the hearer does not know any of these justifications for the event (this actually may be difficult for a user model to ascertain). The cognitive effect is simply that the hearer knows about the event. It is mysterious to the hearer because they do not know its motivation, enablement, cause, or purpose.

```
NAME            narrate-event-MYSTERY
HEADER          Narrate-Event-or-State(S, H, e, chain)

CONSTRAINTS     Event(e) ∧
                (∃x | Motivation(x, e) ∨ Enablement(x, e) ∨
                     Cause(x, e) ∨ Purpose(x, e)) ∧
                ∀x | Motivation(x, e)
                     ¬ KNOW-ABOUT(H, Motivation(x, e)) ∧
                ∀x | Enablement(x, e)
                     ¬ KNOW-ABOUT(H, Enablement(x, e)) ∧
                ∀x | Cause(x, e)      ¬ KNOW-ABOUT(H, Cause(x, e)) ∧
                ∀x | Purpose(x, e)    ¬ KNOW-ABOUT(H, Purpose(x, e))
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, e)
  DESIRABLE     ¬ KNOW-ABOUT(H, e) ∧
                ¬ KNOW(H, Motivation(e)) ∧
                ¬ KNOW(H, Narrator-view(e))

EFFECTS         KNOW-ABOUT(H, e)

DECOMPOSITION

   Do not:      Tell-Enablement/Causation(S, H, e, chain))
   Do not:      optional(Inform(S, H, Motivation(Agent(e), e)))

   Inform(S, H, Event(e))
   optional(∀a | Agent(a,e) ∧ ¬ KNOW-ABOUT(H, a)
                Describe(S, H, Agent(e)))

   optional(∃x | Consequences(e, x)
                Tell-Consequences(S, H, e, chain))

   Do not: optional(∃p Purpose(e,p)
                       Inform(S, H, Purpose(e, p)))

   optional(Inform(S, H, Narrator-Interpretation(e)))
```

Figure 5.29 Plan Operator that Attempts to Create Mystery

Just as an event is mysterious if the hearer does not know what caused or enabled it, an event is suspenseful if the hearer expects something to happen but they are not told about it. Finally, surprise occurs when the hearer does not expect an event to occur. The corresponding plan operator is the same as the regular `Narrate-Event` plan operator except that the constraints would indicate that the speaker has knowledge of the event but the hearer does not. Of course herein lies the operational difficulty: how does a user model determine that the hearer does not know about an event? Indeed, these proposed plan operators have not been implemented. States as well as events can be mysterious, surprising, or suspenseful. An analogous set of plan operators could formalize these effects.

To see how the plan operators eventually might function for a story, consider the canonical form of a subclass of mystery stories (Brewer, 1983 in Wilensky, 1983, p. 595):

> Butler hates his employer. Butler decides to murder him. Butler buys poison. Butler poisons food. Butler plants empty poison bottle in guest's purse. Employer eats food and dies. Detective is called. Detective eventually works out how and by whom murder was done. Butler is arrested.

The story diagram for this murder mystery is shown in Figure 5.30. To make the employer's death mysterious, the planner could fail to indicate the cause of the employer's death (eating the soup or its being poisoned) or withhold events prior in the causal chain (e.g., the butler's motivation). In general, the death of an agent is interesting, in part because of its infrequency and finality (depending upon your religion). It is particularly interesting in this story since it has multiple potential causes and multiple effects.

Similarly, the narrator can create suspense by calling the detective if the reader expects the detective to solve the crime. Finally, surprise occurs when the reader does not expect an event to happen. For example, if the detective arrests the butler before examining the evidence this would violate the reader's expectations and therefore be surprising.

The narrative techniques considered in this section correspond to modifying the order of, and deleting elements from, the underlying plot (i.e., the principal chain of events and states). Event and state inclusion and order affect the hearer's cognitive (e.g., knowledge, beliefs, and desires) and psychological state (e.g., fear, expectation). Narrative techniques make clear how story form (i.e., the discourse structure of the narrative) is independent of the underlying story content (the event/state structure). While plan operators for mystery, suspense, and surprise have not been tested computationally, it appears that the formal

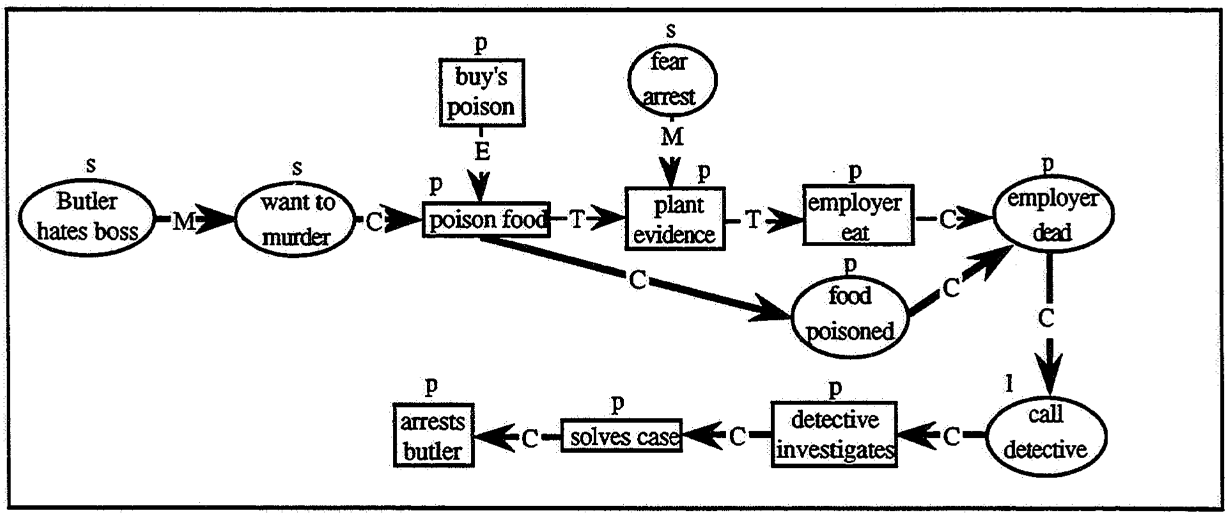



Figure 5.30 Story Diagram of Detective Story

properties of narrative techniques can be operationalized, notwithstanding the difficulty of some of the user modeling issues.

## 5.9 Summary

This chapter is concerned with a second type of text, narration. The chapter first analyzes previous attempts at narration including story grammars, text grammars, and, more recently, RST-based research. The limitations of this previous research led to the definition of the formal event/state ontology which underlies the narratives produced by TEXPLAN. The relationship of this ontology to tense and aspect was established by exploiting Reichenbachian tense models. To help guide the realization of verbs and adverbials, notions of temporal focus and spatial focus were introduced.

The remainder of the chapter formalizes narrative communicative acts as plan operators which are used to produced several forms of narration including reports, stories, and biographies. These forms are illustrated with implemented examples. While over fifty reports and over a dozen biographies have been produced from a mission planner, only one story was generated (although it was produced with domain independent plan operators). While the reports produced were multi-paragraph (and often multi-page), what remains to be investigated is how well these techniques operate on even longer stretches of text. For example, a close correlate of biography and story is *history*. History traces the causes and effects of events (including actions), states, and entities (e.g., objects, ideas) of some people, country, period, or person. History estimates, evaluates, and interprets these happenings, noting especially those that are important, unusual, and interesting -- especially those that can or will shape the course of future events and states (e.g., glasnost). History can be organized along a causal, temporal, or spatial (e.g., geographical) axis and therefore would be a rich vehicle for evaluating the interaction of these organizational threads. In the reports generated from LACE, for example, only topical and temporal sequences are illustrated. Despite the fact that the narrative plan operators of this chapter and the descriptive ones of the previous chapter were not tested on lengthy texts (other than reports), the plan operators contain many of the organizational principles the seem to underlie histories which are typically long texts. Finally, complex forms of narrative raise the issue of the more sophisticated narrative techniques such as surprise, suspense, and mystery which are discussed in the final section of this chapter.

# Chapter 6

# EXPOSITION

I do not know what I may appear to the world,
but to myself I seem to have been only like a boy playing on the seashore
and diverting myself in now and then finding a smoother pebble or a prettier shell than ordinary
whilst the great ocean of truth lay all undiscovered before me

*Isaac Newton*

## 6.1 Introduction

This chapter considers expository text, presents and illustrates the way expository texts are handled in TEXPLAN, and relates this to other attempts to model and generate it. As in the previous chapters, it is claimed that communicative acts underlie the production of text, and exposition and its associated communicative acts are formalized as plan operators.

The purpose of *exposition* is to enable the hearer to do things or to enable them to understand complex processes or ideas. This contrasts with the primary purpose of description (which is to get the hearer to know about an entity and can be viewed as producing a mental image or painting in the hearer's mind) and with the purpose of narration (which is to convey a sequence of events as it were by producing a stage play or motion picture in the hearer's mind). Just as narration employs description to set the background for events and situations, exposition uses description and narration to convey information about processes and propositions. As the purpose of expository prose as I am defining it is to elucidate methods, processes, or ideas, it is typically centered around a chain of events or ideas causally, temporally, spatially, or topically related, hence the need for embedded narrative plans. And in order to paint mental images of unknown entities, exposition may require the use of the rhetorical techniques found in descriptive texts including definition, detail (e.g., attribution, illustration, and purpose), division (e.g., classification and constituency), comparison/contrast, and analogy. As in previous chapters, by exposition we refer to types of text as opposed to a genre.

This chapter examines four types of expository text in turn: *operational instructions*, *locational instructions*, *process exposition*, and *proposition exposition*. Operational and locational instructions indicate how to do something or how to get someplace, respectively. In contrast, process exposition indicates how some mechanism works. Finally, proposition exposition elucidates a proposition, explaining what it means or how or why it is the case: this links into Chapter 7, which in part formalizes communicative acts that attempt to convince the hearer to believe a claimed proposition. Each of the four generic types of exposition produced by TEXPLAN—operational instructions, locational instructions, process exposition, and proposition exposition—is composed of communicative acts which are formalized as plan operators in this chapter. We begin by considering texts that instruct.

## 6.2 Operational Instructions

Perhaps the most common form of exposition, one with which we come in contact frequently, is *operational instruction.* Operational instructions tell us "how to", that is they enable us to perform a variety of tasks including: using things such as appliances or machinery (as in an owner's manual), fixing things (auto and home repair books), cooking (recipes), paying taxes (tax form instructions), and assembling products (assembly instructions). For example, part of a simple instruction found in a "fix-it" book is:

> First, unplug the appliance. Then with a Phillips screwdriver carefully remove the back plate. Now, ...

The text above indicates the two actions (unplug and remove), their temporal relation ("first", "then", "now"), the necessary preconditions of the second action ("with a Phillips screwdriver"), and its manner ("carefully"). The text as a whole enables the hearer to "fix" the appliance, a process which (hopefully) changes the appliance from a broken to a working state. Specific actions within the text are also enabled (e.g., indicating which tool is needed to remove the back plate). Unlike Appelt's KAMP, TEXPLAN does not address the issue of producing referring expressions appropriate to the situation and the user's knowledge (e.g., saying "the screwdriver" instead of "a Phillips screwdriver" if there is only one screwdriver (a Phillips) visible). However, the system does pronominalize when this is appropriate, using focus information (see Chapter 9).

As in the above example, instructions are normally given in second person (often with "you" implied), present tense (i.e., Reichenbachian speech time equals event time), and imperative mood. Sometimes the purpose of the instruction is explicit in the text (e.g., "To get telephone information call 555-1212." or "Captain Ahab's: For reservations, 446-3272"). Often, however, the purpose is implied because it can be inferred from context (as in simply listing a telephone number on a business card).

In the telephone examples just mentioned, it is assumed the hearer knows how to perform the required subtasks (e.g., how to telephone) and the necessary precondition for doing them (e.g., having a phone). In contrast, household products often include explicit instructions for product use. Consider the instructions on the label of Tilex™ Instant Mildew Stain Remover:

> **HOW TO USE:** Use in well ventilated areas. Open windows and turn on fans before use. Turn sprayer nozzle counter-clockwise to open.
> **To remove stains:** spray, wait until stains disappear, and rinse.
> **To clean soap scum:** spray, wait a minute or two, and wipe with a sponge.

The instructions indicate not only how to use the product and its purposes, but also the preconditions of its use and where to use it.

After examining a variety of instructional texts (e.g., product use labels, fix-it books, assembly instructions), the `enable-to-do` operator shown in Figure 6.1 was developed. The constraints on the plan operator require that (1) the speaker knows how to perform the operation (indicated by the intensional operator KNOW-HOW), (2) the speaker knows the subtasks of the operation, and (3) the speaker wants to convey all of this to the hearer. The plan operator prefers (desirable preconditions) that the hearer does not know how to perform the overall task but is able to perform the task (while a physically handicapped individual may in fact KNOW-HOW to perform a task, they may not be physically ABLE to do it). The `enable-to-do` plan operator also prefers that the hearer knows and is able to perform any subtasks (e.g., the hearer may not know how to wax the floor but they are able to do it because they KNOW-HOW and are ABLE to perform subtasks such as sweeping and mopping). Thus the intensional operator ABLE refers to the physical or mental capability of an agent to perform a physical or mental task, which is distinct from their knowledge of that task captured by the KNOW-HOW operator. The assumption made throughout this dissertation is that a user modeling component will be able to provide information about the user's knowledge, beliefs, abilities and so on (indeed, for testing we assume a given user model). However, even if no information were available on the user, it is possible to relax the constraints and preconditions in the plan operators that refer to the cognitive state of the user. We can then still utilize the decomposition and effect portions of plan operators which pair communicative acts with their expected effects. With little or no information about the addressee, the generator would of course produce text that was less tailored to them.

The effect of the `enable-to-do` plan operator is that the hearer knows how to perform some task or action (physical or mental) and knows its subactions. The plan operator does not affect the physical or mental capability of the hearer to perform the task, it only gives them the prerequisite knowledge of how to perform the task. The plan operator accomplishes this by conveying the various constraints, preconditions, and subactions necessary to perform the action (see decomposition in Figure 6.1). Thus, in my initial example about fixing an appliance, the indication that a Phillips screwdriver is to be used is a precondition to actually taking off the back plate. The plan operator also warns the hearer of any dangers of performing the tasks. Finally, if the (user model indicates that the) hearer does not know how to do any of the subactions in the operation, the decomposition of the `enable-to-do` plan operator allows for recursion on that subaction.

```
NAME            enable-to-do
HEADER          Enable(S, H, Do(H, action))

CONSTRAINTS     Action?(action)

PRECONDITIONS
  ESSENTIAL     KNOW-HOW(S, action) ∧
                WANT(S, KNOW-HOW(H, action)) ∧
                ∀x ∈ subacts(action)
                   KNOW(S, Subaction(action, x))
  DESIRABLE     ABLE(H, action) ∧
                ¬ KNOW-HOW(H, action) ∧
                ∀x ∈ subacts(action)
                   ¬ KNOW(H, Subaction(action, x)) ∧
                   ABLE(H, x) ∧
                   KNOW-HOW(H, x)

EFFECTS         KNOW-HOW(H, action) ∧
                KNOW(H, Constraints(action)) ∧
                ∀x ∈ preconditions(action)
                    KNOW(H, Enablement(x, action)) ∧
                ∀x ∈ subacts(action)
                   KNOW(H, Subaction(action, x))

DECOMPOSITION   Inform(S, H, Constraints(action))
                ∀x ∈ preconditions(action)
                    Request(S, H, Do(H, x))
                Warn(S, H, Danger(action))
                ∀subact ∈ subacts(action)
                 optional(Inform(S, H, Constraints(subact)))
                 ∀p ∈ preconditions(subact)
                    optional(Request(S, H, Do(H, p)))
                 Request(S, H, Do(H, subact))
                 ∀y ∈ { y |(Subaction(y, x) ∧ ¬ KNOW-HOW(H, y)) }
                    Enable(S, H, Do(H, y))
```

Figure 6.1 enable-to-do Plan Operator

In addition to telling the hearer how to perform a task, instructions often indicate any necessary preparations, skills, equipment (e.g., tools), or amounts and types of materials (e.g., recipe ingredients or product parts). For example, consider the fish recipe in Figure 6.2 (Street, 1986, p. 228). Apart from linguistic complexities such as coreference or issues of graphical layout, the basic strategy with recipes is to indicate constituents and then narrate key events in the process. Recipes normally assume the hearer has access to and knows how to use key tools. Instruction books sometimes define basic terms, tools, and operations in appendices (Verdon, 1985). Special terms, tools, novel uses of tools or uncommon tasks should be explicated. The strategy used in the above recipe is analogous to that used for assembly instructions with new products or repair instructions in "fix-it" manuals, although the

| |
|---|
| **Smoked Herring Fillets on Garlic Bread**<br>preparation time: 15 minutes<br>3/4 lb smoked herring<br>3/4 lb brown bread<br>2 tbsp butter<br>2 garlic cloves<br>(optional) chili sauce or pickles<br><br>1. Cut each herring into half and divide into fillets, removing the bone.<br>2. Slice the bread. Mix crushed garlic with butter, spread on the slices of bread, lay the herring fillets on top with skin and roe.<br>3. Serve cold, accompanied by chili sauce or pickle. |

Figure 6.2 Fish Recipe

latter do not always explicitly list tools or necessary materials. The strategy is reflected in the `instruct` communicative act, formalized as a plan operator in Figure 6.3. The decomposition of the `instruct` plan operator first indicates the purpose, motivation, or effect of some given task or action and then details the constituents of the object of the action (e.g., ingredients or parts list). The last item in the decomposition calls the `Enable` communicative act in Figure 6.1 which gets the hearer to know the individual steps required to perform the action, for example, the steps in fixing an object or cooking an entree. The `Enable` act also indicates any constraints, preconditions, or warnings associated with the principal action (e.g., "unplug the appliance", "preheat the oven"), and recurses on subactions as required.

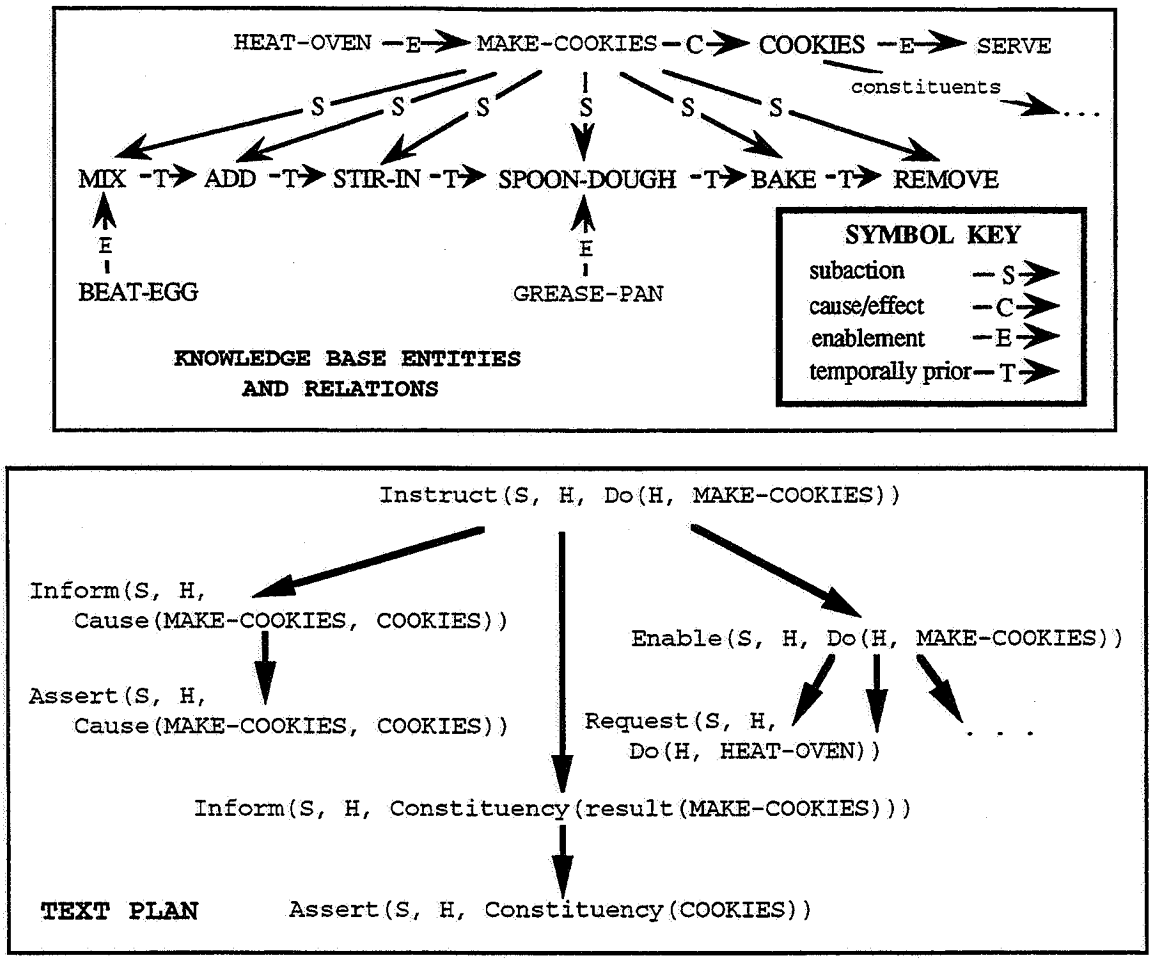
The plan operators of Figures 6.1 and 6.3 were tested with a small knowledge base of cookie recipe/instructions. This was motivated by Dale's (1989) generation of cooking recipes. While Dale's system, EPICURE, focused only on recipe generation (and not on other forms of text such as description or narration), his underlying representation included a rich ontology that even represented the changing nature of ingredients. In contrast to Dale's work, the plan operators in Figure 6.1 and Figure 6.3 assume well-structured data which simplifies the problem and allows the text planner to focus solely on the presentation of a plan. Figure 6.4 represents the three principal elements of the recipe test: the knowledge base, a generated hierarchical text plan, and the corresponding English text. The example was implemented using a FRL (Roberts and Goldstein, 1977) knowledge base which represented actions, subactions, constraints, preconditions, effects, temporal relations, as well as other attributes and values of entities. In Figure 6.4 all knowledge base entities are events (processes or actions) except for COOKIES, which is an object that results from making cookies (this can also be thought of as the state of there being two-dozen cookies.)

```
NAME              instruct
HEADER            Instruct(S, H, Do(H, action))

CONSTRAINTS       Action?(action)

PRECONDITIONS
  ESSENTIAL       KNOW-HOW(S, action) ∧
                  WANT(S, KNOW-HOW(H, action))
  DESIRABLE       ABLE(H, action) ∧
                  ¬ KNOW-HOW(H, action)

EFFECTS           KNOW-HOW(H, action) ∧
                  ∀x (KNOW(H, Purpose(action, x)) ∨
                      KNOW(H, Motivation(action, x)) ∨
                       KNOW(H, Cause(action, x))) ∧
                  ∀z KNOW(H, Constituent(object(action), z))
DECOMPOSITION
  optional(∀x Inform(S, H, Purpose(action, x)) ∨
           ∀y Inform(S, H, Motivation(action, y)) ∨
           ∀z Inform(S, H, Cause(action, z)))
  optional(Inform(S, H, Constituency(result(action))))
  Enable(S, H, Do(H, action))
```

Figure 6.3 `instruct` Plan Operator for Operational Instruction

The text plan in Figure 6.4 is produced in response to the discourse goal KNOW-HOW(H, MAKE-COOKIES) using the `instruct` and `enable-to-do` plan operators. The corresponding English text first indicates the result and constituents of the recipe, then walks through the key actions in the task. This includes indicating any constraints or preconditions on individual actions and the overall task: for example, before the cookies can be baked the oven must be preheated.

While the structure and function of the cookie text plan accurately reflects, I believe, human produced text, a number of improvements can be made. A presentational enhancement would be to lay out the ingredients in tabular format as in the "Herring" recipe above. Similarly, the output text can be compressed by deleting propositions which the addressee can infer or, as in some instructions, informing the addressee about the amounts of constituents in parallel with instructing them on individual steps in the procedure (e.g., "First add two pints of dry gas to your tank. Next ..."). Furthermore, while the content of the recipe seems to reflect those produced by humans, its verbalization is less natural. For example, in Figure 6.4, the second utterance in the first paragraph (based on the constituency rhetorical predicate) would probably more naturally read "For two dozen chocolate chip cookies you will need ..."




**ENGLISH TEXT:**

This recipe makes two dozen chocolate chip cookies. Two dozen chocolate chip cookies contain one half cup of shortening, one cup of sugar, two tablespoons of milk, one tablespoon of vanilla, one egg, one-cup of flour, one half teaspoon of baking soda, and one cup of semi-sweet chocolate pieces.

Heat the oven to 350°F. Beat the egg. Mix the shortening, sugar, milk, vanilla, and egg in a bowl. Add the flour and the baking soda. Stir in the chocolate pieces. Grease a cookie sheet. Spoon the cookie dough onto the cookie sheet. Bake it at 350°F for 9 to 13 minutes. Remove the cookies from the oven. Cool them. Serve them.

Figure 6.4 Chocolate Chip Cookies Knowledge Base, Text Plan, and English Text Structured Using `instruct` and `enable-to-do` Plan Operators

In addition, there are several complexities in generating operational instructions like those exemplified by recipes that are not addressed by, nor are the aim of, this work. Thus, in recipes ingredients or parts may change in shape, form, or texture during the process—so too should the expressions that refer to them (Dale, 1989).[1] Other linguistic complexities include elliptical phrases (e.g., "loosen with spatula" instead of the full form "loosen the cookies with a spatula"), quantification (e.g., "fasten all the red and blue wires"), connectives (e.g., "for 9-13 minutes or until golden brown"), and complex adverbials (e.g., "Heat until wine starts to boil", "stir until smooth"). Another problem not addressed by this work is the simplification and composition of underlying plan components to produce more fluent surface forms (Mellish and Evans, 1990).

In addition, some instructions involve complex temporal, causal, and spatial relations which need to be properly indicated. For example, in many instances it is necessary to execute actions simultaneously:

> To release top of clothes dryer, insert putty knife under it, push knife against clip, and pull on top. It is held by a pair of hidden clips two inches from each end.

These instructions will not have the proper effect if followed literally. That is, the pushing and pulling must occur simultaneously for the top to come off, just as two people lifting a piano from either side must lift simultaneously (cf. Allen, 1984). Assuming information about the time of these actions is represented in the underlying event/state structure, the generator can explicitly indicate this simultaneity by generating temporal adverbials using the notion of temporal focus introduced in the previous chapter. Nevertheless, all of these syntactic, and semantic, and temporal issues require further research.

## 6.3 Locational Instructions

Just as operational instructions tell how to perform tasks to achieve some goal, *locational instructions* tell how to get places. Locational instructions are those given when we ask someone how to drive to our hotel from the airport, how to walk to a new restaurant downtown, which paths to follow to get to the top of a mountain, or, more technically, how to navigate a complex channel. All of these cases entail routes or paths. The term "locational instructions" is used to avoid the ambiguity of the word "directions" which implies both directions that detail how to perform some task (operational instructions) and directions that tell how to get somewhere (locational instructions). To simplify the discussion we will consider only point-to-point routes and not more complex situations such as visiting multiple points along the way (e.g., travelling salesman type problems).

TEXPLAN uses two plan operators to give locational instructions. The first simply geographically identifies a given entity with respect to its well-known or conspicuous (i.e., easily locatable) neighbors.

---

[1] In actual recipes human writers do not always choose correct referring expressions, although what they mean can usually be inferred. Consider: "Mash the potatoes. Put them in the pan." There are no longer several potatoes at the time of the second utterance, just a mush and so the pronoun "it" is more accurate.

The second plan operator actually details the substeps of getting from one place to another, just as operational instructions detail substeps in a process. This section considers these two strategies in turn.

### 6.3.1 Location Identification

In the first case, a speaker can enable the hearer to get to some desired place by simply identifying it, assuming the hearer is familiar with the general area (or can find out about it). The simplest form of *location identification* is an address (e.g., "109 Stanwix Street, Rome, NY") or some other absolute reference (e.g., "29° 15' latitude, 67° 57' longitude"). Absolute locations are particularly useful when devices can pinpoint locations (e.g., LORAN, satellites) as, for example, in marine navigation and rescue and recovery missions. Locations can also be relative as in "fifteen miles Northeast of Calcutta", "15 fathoms under the sea" (depth), "30,000 feet above sea level" (height), or "at the Mall" (collocation). Absolute and relative locations are illustrated by the following examples from the Syracuse, NY yellow pages:

> Lorenzo's Restaurant, in Western Lights Plaza on the Onondaga Blvd side.
>
> Top O' the Hill, minutes from downtown, 5633 W Genesee St., Camillus, located 3/4 mile west of Camillus Plaza, opposite St Joseph's Church.

In the first example, the restaurant is collocated with a large entity and a well-known boulevard. In the second example, not only is an absolute location given (the address)[2], but the restaurant is related to downtown, a plaza, and a church. These examples correspond to the `identify-location` plan operator in Figure 6.5 which identifies the location of an entity assuming the hearer knows the general area of the unknown locale and is able and knows how to get there. The intended effect of the plan operator is that the hearer knows where the place is and knows how to get there.

This plan operator was used with the Map Display System (MDS) (Hilton, 1987) to locate a given object on the map, assuming the hearer knows about the general area of the entity. To locate a town, a lake, an airbase, and so on, the `Location` predicate uses absolute spatial locations (e.g., "in", "on", or "at" some point or entity) as well as relative spatial relations (e.g., "across from", "by", or "North-East of" some entity).

[2]Addresses can of course be ambiguous, but in this example context (i.e., downtown Camillus) enables resolution.

```
NAME            identify-location
HEADER          Enable(S, H, Go(from-entity, to-entity))

CONSTRAINTS     Entity?(from-entity) ∧ Entity?(to-entity) ∧
                path(from-entity, to-entity)
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, to-entity) ∧
                WANT(S, KNOW-ABOUT(H, to-entity)) ∧
                KNOW-HOW(S, Go(from-entity, to-entity)) ∧
                KNOW-ABOUT(H, area(to-entity)) ∧
                KNOW-HOW(H, Go(from-entity, area(to-entity)))
  DESIRABLE     ABLE(H, Go(from-entity, to-entity)) ∧
                KNOW-ABOUT(H, to-entity) ∧
                ∀p ∈ path(from-entity, to-entity)
                    KNOW-ABOUT(H, p) ∧
                    KNOW(H, Subpath(from-entity, to-entity, p))

EFFECTS         KNOW-HOW(H, Go(from-entity, to-entity)) ∧
                KNOW-ABOUT(H, area(to-entity))

DECOMPOSITION   Inform(S, H, Location(to-entity))
```

Figure 6.5 `identify-location` Plan Operator

The Map Display System represents locations at three levels of spatial abstraction: Cartesian (x,y,z) coordinates measured in kilometers, longitude/latitude pairs, and Universal Transverse Mercator (UTM) coordinates. While the first two are self-explanatory, the last is the system used by the United States Defense Mapping Agency (DMA) based on the Military Grid Reference System (MGRS), a global mapping system which represents two-dimensional coordinates in a single value called a Universal Transverse Mercator (UTM) coordinate. UTM coordinates have the form ZZSBBEENN[3] which encodes the zone (z), strip (s), alphabetic code for each 100 kilometer block (b), easting (e), and northing (n). In the Map Display System a spatial database represents hierarchies of blocks of 100 kilometers, 10 kilometers, and 2 kilometers. As the system is knowledge based, each of these blocks has properties (e.g., terrain type, elevation, etc.) and indicates the entities therein (e.g., powerlines, dams, bridges, etc.). Because of this structure, it is possible not only to provide the absolute location of any given object (in UTM, longitude/latitude, or x-y-z coordinates), but also to relate any entity to other entities within or near its block (e.g., relating a smaller town to a larger, more recognizable city). Using the kilometer-based Cartesian coordinate system, simple Euclidian geometric functions were developed to calculate relative distances and directions (e.g., N, S, E, W) between objects.

[3] There actually can be up to five easting and northing codes.

For example, assuming the user's current location is the town of Eisenberg and they ask "Where is Karl-Marx-Stadt?" the system uses the plan operator in Figure 6.5 to produce the text plan:[4]

```
Enable(S, H, Go(#<Eisenberg>, #<Karl-Marx-Stadt>))
                    ↓
Inform(S, H, Location(#<Karl-Marx-Stadt>))
                    ↓
Assert(S, H Location(#<Karl-Marx-Stadt>))
```

which is realized as:

```
Karl-Marx-Stadt is a town located in block 33UUS5030 at 33UUS5135.
```

Since it is possible to convert from UTM coordinates to longitude/latitude, the following response can also be produced by TEXPLAN:

```
Karl-Marx-Stadt is a town located in block 33UUS5030 at 50.82° latitude
12.88° longitude.
```

This presentational difference could be signaled perhaps by a user type, where military or expert users might be assumed to prefer UTM coordinates and civilians or novices might prefer the more common longitude/latitude coordinates. This is analogous to the situation where the user asks "How do I get from here to Oberlungwitz?" (simulated by posting the goal KNOW-HOW(H, Go(#<town Eisenberg>, #<town Oberlungwitz>))), and the user model signals that the addressee is familiar with the general location of Oberlungwitz, i.e., knows about its block. In this situation TEXPLAN replies:

```
Oberlungwitz is a town located in block 33UUS3020 at 50.75° latitude
12.70° longitude nine kilometers South-West of Lichtenstein-Sachsen and
two kilometers South of Gersdorf.
```

If the user instead asks, "Where is Granetalsperre?", and they know the region, TEXPLAN says:

```
Granetalsperre is a dam located in block 32UNC9050 at 51.89° latitude and
10.37° longitude two kilometers North-West of Langelsheim and three
kilometers Northeast of Goslar.
```

[4] In the actual implementation of the Map Display System, the printed representation of entities includes their type. For example #<Karl-Marx-Stadt> actually appears as #<town Karl-Marx-Stadt>. This is necessary to discern between multiple entities with the same name (i.e., the town Karl-Marx-Stadt versus the identically named airbase).

Relative locations are based on nearby towns in the same block, which corresponds roughly to a local cluster or region of entities. Each block and subblock records the towns, roads, intersections, railroads, borders, airstrips, heliports, obstructions, lakes, powerlines, dams waterways, and bridges. While entities are related to other towns in the block, a more psychologically plausible approach might be to incorporate some measure of the perceptual saliency (Conklin, 1983) of related entities, e.g., weighting entities according to their utility as anchor points. For example, well-known lakes or very large industrial cities serve as better anchor points than smaller, remote towns. Some measure could be devised based on location, size, frequency of occurrence (e.g., there are few dams but lots of towns) and so on. In this manner TEXPLAN could select reference points based on a dynamic measure of saliency.

In the `identify-location` plan operator of Figure 6.5, the function `area` used in the essential precondition of the plan operator returns the sector or block in which the entity appears. An assumed user model of what sectors users do or do not know is used to guide plan operator selection. As a default, the user model assumes the user is unfamiliar with all blocks,[5] although after text is produced this model is updated. The plan operator could easily be adapted to new domains by redefining the `area` function along other sectors such as city limits, county lines, or some "virtual" grouping. While the `identify-location` plan operator is effective at locating unknown locales in areas the user is familiar with, it does not address how people given lengthier instructions on how to travel from one location to another. The next subsection considers this case.

### 6.3.2 Locational Instruction

The location identification strategy is only valid if the hearer is familiar with the general area of the unknown entity. If the hearer is unfamiliar with the area, then it is necessary to give explicit directions, i.e., *locational instructions*, starting from their current or some known location. For example, consider the following directions given to a person in one city, Rome, NY, who wants to travel to a restaurant in another city, Syracuse, NY. (The writer assumes the hearer is familiar with the interstate highway system.)

> To get to the Country Inn from Rome take the New York State Thruway (Interstate 90) to Exit 39. Take a left onto Interstate 690. Travel a quarter mile to the Farrell Road exit. The Country Inn is located at 1615 State Fair Blvd.

The directions appear in the order of the path from start to finish. Distance adverbials like "a quarter mile" are relative to the current location in the path (i.e., the spatial focus). Destination adverbials like "to Farrell Road exit" indicate the local destination or goal of an individual action in the overall itinerary. The final utterance, "The Country Inn ...", simply identifies the location as in the above `identify-location` plan operator. This is typical of most spatial directions: once you are physically near the desired location, its distinguishing geographic characteristics (i.e., relationships to other conspicuous entities) are identified.

[5] While this could be enhanced, user modeling is not the principal aim of this work.

Landmarks or other distinctive points can also be used to anchor individual instructions as in the adverbial "take a left at the large Exxon sign".

This type of strategy is represented in the `enable-to-get-to` plan operator shown in Figure 6.6, which enables the hearer to go from one place or entity (e.g., home, city, state, country) to another. The decomposition of the plan operator first finds and then describes a path between the two entities. It concludes by identifying the absolute and relative location of the entity. The plan operator requires that the speaker knows such a path and wants to convey it to the hearer. It also prefers that the hearer is in fact physically able to go from one place to the other and knows about the area they want to go to, but does not know how to get to the desired locale (i.e., does not know the subpaths to it). The effect of the plan operator is that they will know how to get there.

The `enable-to-get-to` plan operator in Figure 6.6, like the `identify-location` plan operator, was tested using the Map Display System (Hilton, 1987). Figure 6.7 gives a simplified visual perspective of the Map Display System which represents over 600 towns, over 200 airstrips, and over 4,600 road segments. The map includes a road network which represents 233 roads (divided up into 4,607 road segments) and 889 intersections of roads. The map also represents 605 towns, 227 airbases, 40 lakes, 14

```
NAME            enable-to-get-to
HEADER          Enable(S, H, Go(from-entity, to-entity))

CONSTRAINTS     Entity?(from-entity) ∧ Entity?(to-entity) ∧
                path(from-entity, to-entity)
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, to-entity) ∧
                WANT(S, KNOW-ABOUT(H, to-entity)) ∧
                KNOW-HOW(S, Go(from-entity, to-entity)) ∧
                WANT(S, KNOW-HOW(H, Go(from-entity, to-entity)))
  DESIRABLE     ¬ KNOW-ABOUT(H, area(to-entity)) ∧
                ABLE(H, Go(from-entity, to-entity)) ∧
                KNOW-ABOUT(H, to-entity) ∧
                ¬ KNOW-HOW(S, Go(from-entity, to-entity)) ∧
                ∀p ∈ path(from-entity, to-entity)
                   KNOW-ABOUT(H, p)

EFFECTS         KNOW-ABOUT(H, area(to-entity)) ∧
                KNOW-HOW(H, Go(from-entity, to-entity)) ∧
                ∀p ∈ path(from-entity, to-entity)
                   KNOW(H, Subpath(from-entity, to-entity, p))

DECOMPOSITION   ∀p ∈ Path(from-entity, to-entity)
                  Request(S, H, Do(H, Go(p, next-segment(p))))
                  optional(Inform(S, H, Location(p)))
                Inform(S, H, Location(to-entity))
```

Figure 6.6 `enable-to-get-to` Plan Operator

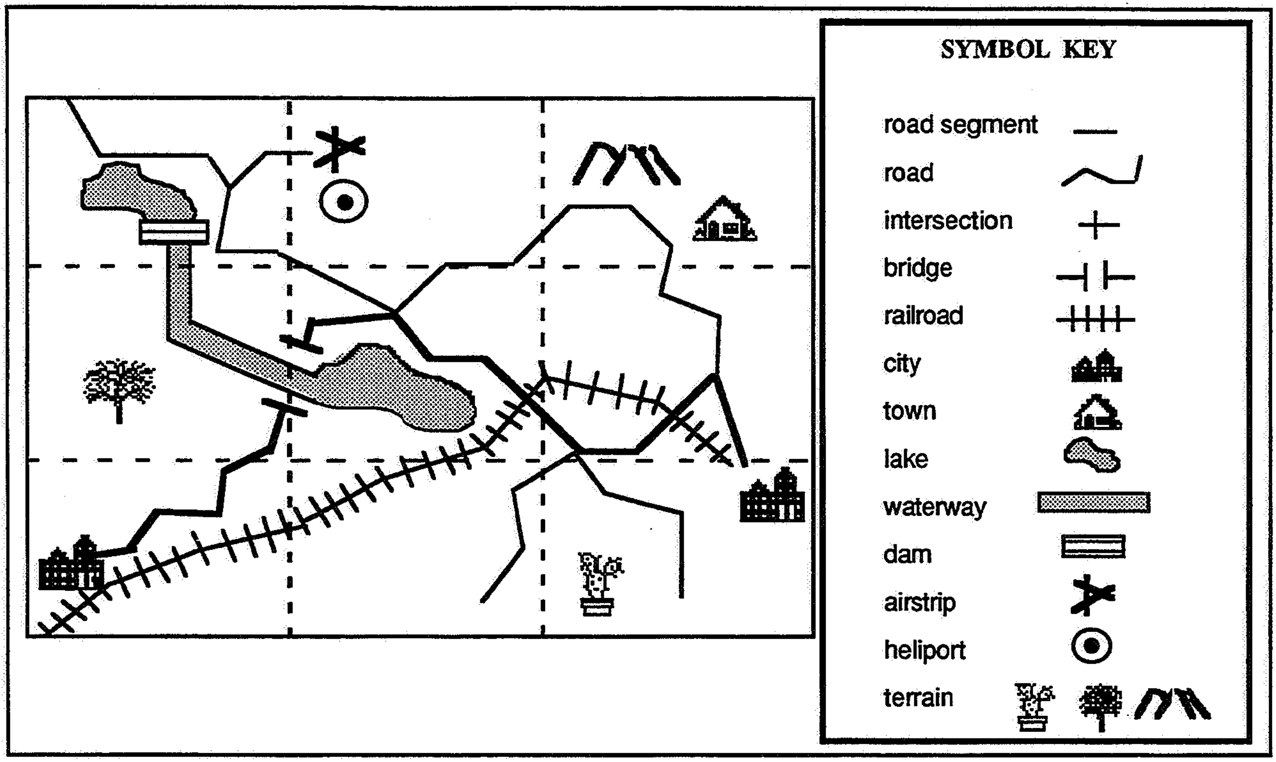



Figure 6.7 Map Display System: Schema of Blocks (dotted lines) and Selected Path (bold)

dams, and other objects that are located on or at the end of roads (see Figure 6.7). The function `path` used in the plan operator in Figure 6.6 takes as arguments two objects from the cartographic knowledge base and, using a branch and bound search strategy, explores the road network to return the "best" route between the two points (if one exists). The path returned by the function is an ordered list of roads, intersections, and towns indicating the preferred route from one entity to another, as defined by the rewrite rules:

```
path      -> segment + (path)
segment   -> road-segment | intersection | town
```

where "()" indicates optionality and "|" indicates logical disjunction (i.e., or).

For example, assume the user asks TEXPLAN (interfaced to the Map Display System) how to get from Wiesbaden to Frankfurt. This is simulated by posting to TEXPLAN the discourse goal `KNOW-HOW(H, Go(#<Wiesbaden>, #<Frankfurt-am-Main>))`. Assuming that the user model indicates that the user is not familiar with the area, the generator then attempts to achieve this goal by producing the text plan in Figure 6.8 using the `enable-to-get-to` plan operator in Figure 6.6. The text plan of Figure 6.8 is then realized as:

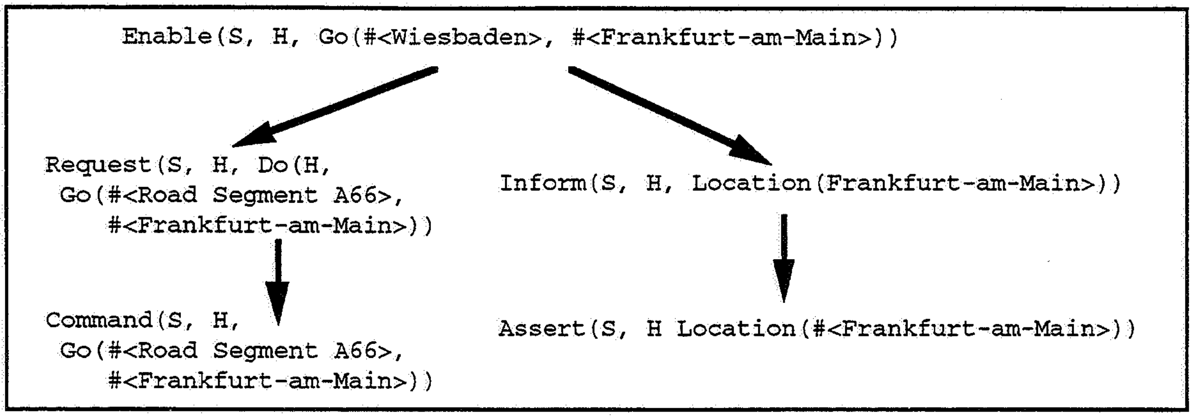



Figure 6.8 Text Plan for Locational Instructions

From Wiesbaden take Autobahn A66 Northeast for thirty-one kilometers to Frankfurt-am-Main. Frankfurt-am-Main is located in block 32UMA7050 at 50.11° latitude and 8.66° longitude.

A slightly more complex locational instruction results if the user asks how to get from Mannheim to Heidelberg, initiated by posting the discourse goal KNOW-HOW(H, Go(#<Mannheim>, #<Heidelberg>)).

From Mannheim take Route 38 Southeast for four kilometers to the intersection of Route 38 and Autobahn A5. From there take Autobahn A5 Southeast for seven kilometers to Heidelberg. Heidelberg is located in block 32umv7070 at 49.39° latitude and 6.68° longitude, 4 kilometers Northwest of Dossenheim, six kilometers Northwest of Edingen, and five kilometers Southwest of Eppelheim.

These texts are produced by reasoning about the path between the two entities. A pointer is maintained to the current spatial focus (SF - introduced in Chapter 5, in locational directions the most recently traversed segment of the path) in order to generate relative spatial adverbials such as headings (e.g., "Northwest (from here)") and distances (e.g., "three miles (from there)"). In the current implementation, a temporal adverbial (e.g., "travel West for 6 minutes") can be substituted for a spatial adverbial (e.g., "travel West for 12 kilometers") in an ad hoc manner, by converting distance to time assuming some velocity (e.g., 120 kilometers per hour). Distance and direction adverbials are based on the relation of the next segment in the path to the current SF. Straightforward Euclidian algorithms were developed to calculate distance and direction in relation to the current SF by using the absolute locations associated with each entity. As in operational instructions, because the speaker is mentally travelling down the path recounting events as they occur, speech time is equal to event time which motivates the use of present tense. When text like those above are produced, if there is no negative user feedback then the user model is the updated by the discourse controller to indicate that the user knows how to get to the desired location.

The `enable-to-get-to` plan operator in Figure 6.6 can be extended to enhance locational instructions with other illocutionary actions. For example just as you might warn a ship captain about dangerous obstacles when giving directions for navigating down a channel, you would equally tell someone to avoid the dangerous parts of a big city, or perhaps to be careful about notorious speed traps along an interstate highway. Similarly, you would inform them of any constraints or preconditions on their travelling (e.g., special clothing, materials, weather conditions, etc.).

The examples in this section are based on a road network and other related text types may have their own particular characteristics. For example, while air routes would likely follow the same strategy, they might use different "anchor" points (e.g., landmarks as opposed to intersections and buildings). Furthermore, it would be desirable to make the selection of individual path segments sensitive to a model of the user. For example, in the context of giving locational instructions about a child's map, Shadbolt (1984) discusses how depending on the "communicative posture" of the participant, certain aspects of a discourse are conveyed explicitly whereas others are left to the hearer to infer. Refining these ideas, Carletta (1990ab) describes a planning architecture which allows interruptions, checking moves, and repair and replanning strategies to recover from miscommunications involving navigational instructions around Shadbolt's (1984) map. Others have considered path selection and pruning to give better locational instructions (McCalla and Schneider, 1979).

The first half of this chapter has detailed plan operators which enable the user to perform operational and locational tasks. In contrast to this focus on the enablement of actions, the two remaining forms of exposition that TEXPLAN produces focus on enabling the hearer simply to understand a process or proposition. That is, these two expository forms explicate processes and propositions, respectively. We first consider text that enables the user to understand complex processes.

## 6.4 Process Exposition

In contrast to operational and locational instructions which enable the user to get to some location or perform some action, the purpose of *process exposition* is to make the user understand the sequence of events or states that occur in some process. Unlike event narration, on the other hand, which usually describes past events and states, process exposition usually describes what happens in the third person, present tense, as in the following exposition of how the heart works.

> During the heart's relaxed stage (diastole), oxygen-depleted blood from the body flows into the right atrium and oxygenated blood from the lungs flows into the left atrium. Then the natural pacemaker, or sinoatrial node, fires electrical impulses causing the atrial to contract. This causes the valves to open and blood fills the ventricles. During the pumping stage (systole), the electrical signal, relayed through the atrioventricular node, causes the

ventricles to contract. This forces oxygen-poor blood to the lungs and oxygen-rich blood to the body.

This text is organized around the three principal stages of the heart's pumping: diastole, electrical impulse, and systole. Process exposition typically follows the underlying causal/temporal/spatial organization of the mechanism being described. TEXPLAN's plan operator for a process exposition, `explain-process`, is shown in Figure 6.9. The plan operator gets the hearer to know about some entity (e.g., the heart) and how the process associated with it (e.g., blood circulation) works. That is, the intended effect is not that the hearer will perform the process themselves (in contrast to operational instructions), but rather that they will understand how the process works. The plan operator's decomposition first defines the entity and indicates its purpose, and then divides the entity into its main subparts or subtypes using plan operators defined in Chapter 4. The plan operator then retrieves the events and states of the process associated with the entity being explained. The narration plan operators defined in Chapter 5 attempt to recognize a path through this event/state network to organize the resulting text causally, temporally, spatially, or topically, so narrative plan operators are subplans within the whole process exposition.

To illustrate the process exposition plan operator, an event/state network for the heart diastole/impulse/systole process was developed and represented in a small FRL (Roberts and Goldstein, 1977) knowledge base along with information about a heart (e.g., subparts, attributes, etc.). Figure 6.10 gives a sketch of the knowledge base where all the items are events except for the state `heart-relaxed`.

```
NAME            explain-process
HEADER          Explain(S, H, entity)

CONSTRAINTS     Has-Process?(entity)
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, entity) ∧
                WANT(S, KNOW-ABOUT(H, entity)) ∧
                KNOW-HOW(S, process(entity)) ∧
                WANT(S, KNOW-HOW(H, process(entity)))
  DESIRABLE     ¬ KNOW-ABOUT(H, entity)

EFFECTS         KNOW-ABOUT(H, entity) ∧
                KNOW-HOW(H, process(entity)) ∧
                ∀x KNOW(H, Purpose(entity, x))

DECOMPOSITION   Define(S, H, entity)
                ∀x optional(Inform(S, H, Purpose(entity, x)))
                Divide(S, H, entity)
                Narrate(S, H, event-and-states(process(entity)))
```

Figure 6.9 `explain-process` Plan Operator

As in the narrative event/state networks of the previous chapter, the main path of events in the process is indicated by a dashed line. Because the information in the knowledge base contains only causal and temporal relations among entities, it is necessary to reason about the types of information and their communicative function in the text in order to produce an effective text.

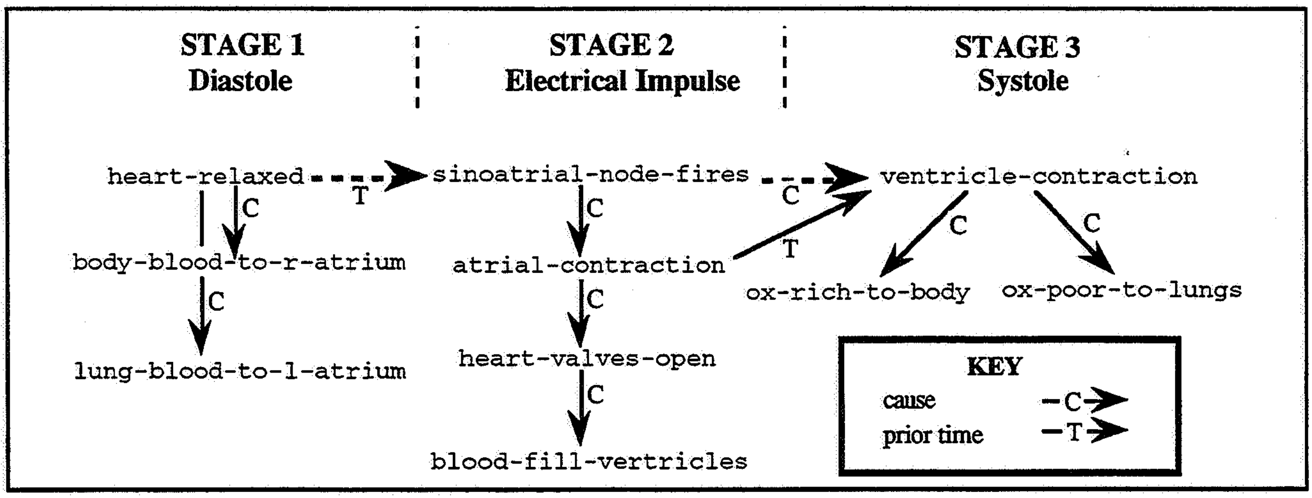



Figure 6.10 Pumping Heart: Stages and Event/State Network

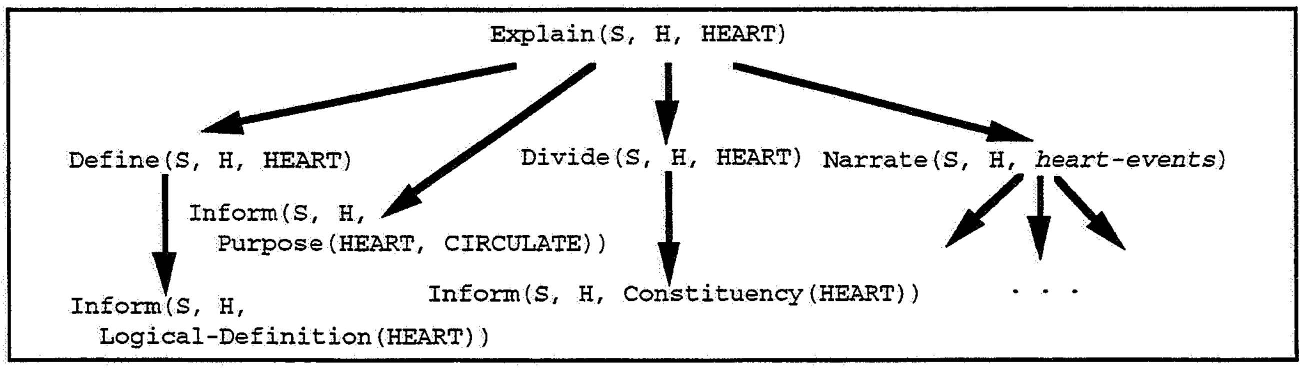



Figure 6.11 Text Plan for Pumping Heart Exposition

When the discourse goal `KNOW-HOW(H, process(HEART))` is posted to TEXPLAN, the plan operator in Figure 6.9 produces the (top-level) text plan shown in Figure 6.11 for an exposition of a pumping heart. The text plan in Figure 6.11 corresponds to the following surface form where the paragraph break is signalled by the structure in the text plan (i.e., the call to the `Narrate` plan operator).

```
The heart is an organ located in the chest.  The purpose of the heart is
to circulate blood.  The heart contains four parts:  the left atrium,
the right atrium, the left ventricle and the right ventricle.

First the heart is relaxed.  The relaxed heart causes oxygen-depleted
```

```
blood from the body to flow into the right atrium and oxygenated blood
from the lungs to flow into the left atrium.  Next the sinoatrial node
fires electrical impulses.  The electrical impulses cause atrial
contraction.  Atrial contraction causes the heart valves to open.  The
open heart valves causes blood to fill the ventricles.  The electrical
impulses also cause ventricle contraction.  Finally, the ventricle
contracts.  Ventricle contraction causes oxygen-poor blood to flow to
the lungs and oxygen-rich blood to flow to the body.
```

The first three sentences in this text describe the heart using rhetorical acts of logical definition, purpose, and constituency, respectively. Then using the narrative plan operators defined in Chapter 5, the remainder of the text follows the causal connections shown in Figure 6.7 to order the key events and states involved in pumping blood. But unlike the narrative texts detailed in the previous chapter which were realized in past tense, process exposition uses present tense. This is because while in report or story narration the Reichenbachian speech time is assumed to be after the event time, in process exposition speech time is assumed to be equivalent to event time which results in the use of present tense. Furthermore, because there are no explicit times in the underlying event/state model as there were in the narrative examples of Chapter 5, the linguistic realization component cannot make reference to specific times (e.g., "ten minutes later"). It instead indicates relative times (e.g., "then") of events in distinct temporal time chunks (indicated by explicit temporal links in the knowledge base). Finally, note the use of the adverbial "also" in the penultimate sentence of the example text. This anaphoric reference to a repeated event (causation) is analogous to the use of "again" in Chapter 5 to indicate repeated events in narrative reports from LACE. Producing the "also" adverbial is accomplished by preprocessing the event/state network to identify which events in the main path cause multiple events to occur. The manner slot of all but the first of the resulting events is marked with this information which drives the realization of "also".

In addition to this anaphoric use of the adverbial "also", the clue words "first", "next," and "finally" in the above example are used to signal the structure of the underlying text plan. Because the narrative portion of the heart exposition is structured around the causal main-path of events, this gives rise to a narrative text structure that is centered around the three main processes of diastole, electrical impulse, and systole. Just before the text plan is linearized and realized, a slot in the rhetorical proposition associated with each event in the main path is marked with a connective (e.g., "first", "next") in order to signal the hierarchical structure of the text plan. Other connectives may be subsequently added during linguistic realization in order to indicate the type of rhetorical predicate used (e.g., illustration -> "for example"; See Table 8.1).

As detailed in Chapter 2, Paris' (1987ab) TAILOR system also generated a process exposition (of a telephone). This used a "process trace" strategy represented in an ATN to trace underlying causal, temporal, and equivalence connections of entities in a frame knowledge base. As indicated in the previous chapter, the path selection algorithm underlying TEXPLAN's narrative plan operators is inspired by that developed for TAILOR. There are, however, several differences between TAILOR and TEXPLAN's production of exposition. First, the text produced by TAILOR's process trace strategy includes only a trace of the underlying process, whereas the TEXPLAN process exposition plan operator in Figure 6.6 defines, characterizes, and divides the heart into constituent parts (or subtypes if they exist) before it narrates the process (i.e., event/state) sequence. On the basis of an assumed user model, TAILOR could choose between a constituency schema (for experts) and a process trace (for novices) for particular components of a device, thus tailoring output to the user (detailed in Chapter 2). But as the above heart exposition illustrates, both descriptive and expository techniques can be used concurrently. A more important difference is that TEXPLAN's strategy is represented declaratively as plan operators which are used by a general hierarchical planner to construct executable text plans. (The advantages of plan-based models of communication are detailed in the final section of Chapter 3.) Perhaps the most significant difference regarding process exposition in TAILOR and TEXPLAN concerns the content of TEXPLAN's plan operators. Not only do they distinguish rhetorical, illocutionary, and surface speech actions, but they explicitly represent what expected effect(s) their use will have on the hearer and so, unlike TAILOR, TEXPLAN can build a model of the expected effects of its utterances on the hearer.

In addition to explaining how to do things, how to get places, and how things work, authors often find the need to explain the how or why of propositions. This requires a final form of expository text: proposition exposition.

## 6.5 Proposition Exposition

It is often necessary to explain general propositions such as "Politicians are ambitious" or specific ones like "Napoleon was ambitious". Propositions are either true or false. Propositions attribute properties or states to an entity (e.g., "John is small."), indicate relations between entities (e.g., "John's wife is Mary."), or indicate events (e.g., "John hit the ball yesterday."). These natural language expressions of propositions can be represented more formally (for discussion purposes here very simply) as predicate-argument structures like `small(John)`, `wife(John, Mary)` and `hit(John, ball, yesterday)`. In each of these cases it is possible that the hearer is not familiar with the *terms* or arguments[6] of the proposition or with its *predicate*. For example, the statement "Noriega is a dictator" corresponds to the proposition `dictator(Noriega)` where the predicate is `dictator` and the term is `Noriega`. The hearer could know about Noriega and know about dictators, but not that he was one, and still understand the proposition. If, however, the hearer either does not know what a dictator is or does not know about Noriega, they cannot fully appreciate the statement.

If the statement of a proposition confuses the hearer, it is possible that they do not understand the predicate or the term(s) of the proposition and so speakers often describe both predicate and term(s). This

[6]Because "argument" can refer to both the terms of a logical proposition and the general form of prose that aims to convince or persuade, the argument(s) of a proposition are called "term(s)."

```
NAME            explain-proposition-by-description
HEADER          Explain(S, H, proposition)

CONSTRAINTS     Proposition?(proposition)
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, proposition) ∧
                WANT(S, KNOW-ABOUT(H, proposition))
  DESIRABLE     ¬ KNOW-ABOUT(H, proposition) ∧
                ¬ KNOW-ABOUT(H, predicate(proposition)) ∧
                ∀x ∈ terms(proposition)
                     ¬ KNOW-ABOUT(H, x)

EFFECTS         KNOW-ABOUT(H, proposition) ∧
                KNOW-ABOUT(H, predicate(proposition)) ∧
                ∀x ∈ terms(proposition)
                     KNOW-ABOUT(H, x)

DECOMPOSITION   Describe(S, H, predicate(proposition))
                ∀x ∈ terms(proposition)
                    Describe(S, H, x)
```

Figure 6.12 `explain-proposition-by-description` Plan Operator

is a natural strategy since there is a certain mutual dependency between predicate and argument. Thus to explain the statement "Noriega is a dictator", a speaker might say "Dictators wield absolute power. Noriega is a South-American politician and a drug-runner." were the first utterance defines the predicate, `dictator`, and the second defines its term, `Noriega`.

This type of proposition exposition corresponds to the `explain-proposition-by-description` plan operator shown in Figure 6.12 which has the effect that the hearer knows the proposition and knows about its predicate and all of its terms. The decomposition of the plan operator first describes the predicate of the proposition and then describes all of its terms. As the plan operator has access to all of the different types of descriptive operators formalized in Chapter 4, it can not only define the predicate and terms of the proposition, but also it can give examples of them, compare and contrast them to other entities the system believes the user knows, give analogies, and so on.

Sometimes describing the predicate and terms of a proposition is not sufficient to get the hearer to understand it. Just as a variety of descriptive plan operators are necessary to recover from miscommunications, we require alternative plan operators to elucidate propositions. For example, consider the dialogue in Figure 6.13 between an adult (Michelle) and a three-year old friend (Kelly). In order to explicate the dog's actions to Kelly, Michelle uses a variety of strategies. The conversation moves from description of the specific actions of the dog to the purpose behind the dog's actions, the cause/effect of the action, and ultimately to the overall motivation or intent (instinct) of his action. In the final utterance, Michelle resorts to an analogy to justify the events unfolding before the hearer.

*Visual context: My dog Fritzie is burying his bone underneath some pillows on the couch.*

| | |
|---|---|
| Kelly: | What is Fritzie doing? |
| Michelle: | He's burying his bone. |
| Kelly: | Why? |
| Michelle: | Because he wants to hide it. |
| Kelly: | Why? |
| Michelle | Because that's what dogs do. |
| Kelly: | (perplexed facial expression) |
| Michelle: | Fritzie is burying his bone so nobody finds it. |
| Kelly: | Why? |
| Michelle: | So that he can eat it later on when he's hungry. |
| Kelly: | Why? |
| Michelle: | Fritzie buries his bones just like a squirrel buries nuts so that he can eat them later on. |
| Kelly: | Oh. |

Figure 6.13 Michelle and Kelly's Conversation

TEXPLAN similarly uses a range of strategies for proposition exposition. For example, Figure 6.14 shows another strategy which exemplifies a general proposition. This is distinct from the `describe-by-illustration` communicative act defined in Chapter 4 because description has been applied so far only to the predicate and/or term(s) of some proposition. The `explain-proposition-by-illustration` plan operator in Figure 6.14, in contrast, provides examples of the entire proposition.

While a user may understand a proposition, they may not understand how or why it is true. This may be because they do not know, for example, what enabled or caused the proposition to be true. Propositions indicate either events or states (defined in the previous chapter) which have relationships to other events and states such as enablement (i.e., precondition), causation, motivation, and purpose. Therefore, if TEXPLAN believes that the user already knows about or understands the proposition, it uses the `explain-reason-for-proposition` plan operator shown in Figure 6.15 to indicate the enablement, motivation, or cause of the proposition.

```
NAME            explain-proposition-by-illustration
HEADER          Explain(S, H, proposition)

CONSTRAINTS     Proposition?(proposition)
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, proposition) ∧
                WANT(S, KNOW-ABOUT(H, proposition))
  DESIRABLE     ¬ KNOW-ABOUT(H, proposition) ∧
                KNOW-ABOUT(H, predicate(proposition)) ∧
                ∀x | Argument(proposition, x)
                    KNOW-ABOUT(H, x)

EFFECTS         KNOW-ABOUT(H, proposition) ∧
                ∀x ∈ examples(proposition)
                    KNOW(H, Illustration(proposition, x))

DECOMPOSITION   ∀x ∈ examples(proposition)
                    Inform(S, H, Illustration(proposition, x))
```

Figure 6.14 `explain-proposition-by-illustration` Plan Operator

```
NAME            explain-reason-for-proposition
HEADER          Explain-How(S, H, proposition)

CONSTRAINTS     Proposition?(proposition)
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, proposition) ∧
                WANT(S, KNOW-ABOUT(H, proposition)) ∧
                KNOW-HOW(S, proposition) ∧
                WANT(S, KNOW-HOW(H, proposition)
  DESIRABLE     KNOW-ABOUT(H, proposition) ∧
                KNOW-ABOUT(H, predicate(proposition)) ∧
                ∀x ∈ Argument(proposition, x)
                    KNOW-ABOUT(H, x)

EFFECTS         KNOW-HOW(H, proposition) ∧
                ∀x ∈ preconditions(proposition)
                    KNOW(H, Enablement(x, proposition)) ∧
                ∀x ∈ motivations(proposition)
                    KNOW(H, Motivation(x, proposition)) ∧
                ∀x ∈ causes(proposition)
                    KNOW(H, Cause(x, proposition))

DECOMPOSITION   ∀x ∈ preconditions(proposition)
                    Inform(S, H, Enablement(x, proposition))
                ∀x ∈ motivations(proposition)
                    Inform(S, H, Motivation(x, proposition))
                ∀x ∈ causes(proposition)
                    Inform(S, H, Cause(x, proposition))
```

Figure 6.15 `explain-reason-for-proposition` Plan Operator

In addition to the plan operators in Figures 6.14 and 6.15, it may make sense to explain the purpose implicit in the content of an action (e.g., "Fritzie barked.") which is executed by some intentional agent. For example, in the discussion in Figure 6.13 about the dog Fritzie, Michelle explains the dog's actions in terms of the goal(s) or purpose(s) he is trying to achieve. He buries his bone to hide it so that he can eat it later when he is hungry. This strategy is reflected in the `explain-purpose-for-proposition` plan operator in Figure 6.16 where the speaker indicates the purpose(s) of the proposition (which the constraints dictate must be an action).

```
NAME             explain-purpose-for-proposition
HEADER           Explain-Why(S, H, proposition)

CONSTRAINTS      Proposition?(proposition) ∧
                 Action?(predicate(proposition))
PRECONDITIONS
  ESSENTIAL      KNOW-ABOUT(S, proposition) ∧
                 WANT(S, KNOW-ABOUT(H, proposition))
  DESIRABLE      KNOW-ABOUT(H, proposition) ∧
                 KNOW-ABOUT(H, predicate(proposition)) ∧
                 ∀x  | Argument(proposition, x)
                      KNOW-ABOUT(H, x)

EFFECTS          ∀x ∈ purposes(proposition)
                     KNOW(H, Purpose(proposition, x))

DECOMPOSITION    ∀x ∈ purposes(proposition)
                     Inform(S, H, Purpose(proposition, x))
```

Figure 6.16 `explain-purpose-for-proposition` Plan Operator

Finally, if the proposition does not detail an action (which can have a purpose), then it might be understood if the hearer knows about its consequences. The `explain-consequence-of-proposition` plan operator in Figure 6.17 does precisely this. Given a proposition, if it is not an action then it informs the hearer of what the state or event causes.

Just as the descriptive operators in Chapter 4 could be combine to provided an extended description, TEXPLAN has a plan operator that combines the various proposition exposition techniques into an extended exposition plan operator. This plan operator first describes the predicates and terms of the proposition, then details what enabled, motivated, or caused it, and finally indicates what its purpose was (e.g., as in the purpose of an action). Furthermore, a recursive call in the decomposition of the above plan operators could be added to deal with compound propositions.

To illustrate these proposition exposition plan operators, consider the heart knowledge base used previously for process exposition. Assume that the user has just read the exposition of the heart pumping but is confused by the statement concerning atrial contraction. Consider the following dialogue where user

```
NAME            explain-consequence-of-proposition
HEADER          Explain-Consequence(S, H, proposition)

CONSTRAINTS     Proposition?(proposition) ∧
                ¬ Action?(predicate(proposition))
PRECONDITIONS
  ESSENTIAL     KNOW-ABOUT(S, proposition) ∧
                WANT(S, KNOW-ABOUT(H, proposition))
  DESIRABLE     KNOW-ABOUT(H, proposition) ∧
                KNOW-ABOUT(H, predicate(proposition)) ∧
                ∀x |Argument(proposition, x)
                    KNOW-ABOUT(H, x)

EFFECTS         ∀x |Cause(proposition, x)
                    KNOW(H, Cause(proposition, x))

DECOMPOSITION   ∀x |Cause(proposition, x)
                    Inform(S, H, Cause(proposition, x))
```

Figure 6.17 `explain-consequence-of-proposition` Plan Operator

queries are simulated by posting corresponding discourse goals to the generator. For example, U1 is simulated by posting the discourse goal KNOW-ABOUT(H, CONTRACT(ATRIA)) to TEXPLAN.

U1: *What does "The atrial contracts" mean?*
S1: `Contraction is a restriction of the muscles.`
`The atria are chambers located at the top of the heart.`
U2: *Why does the atrial contract?*
S2: `Electrical impulses cause atrial contraction.`
U3: *What does atrial contraction cause?*
S3: `Atrial contraction causes the heart valves to open.`

The `explain-proposition` plan operator in Figure 6.12 is used to produce S1 which first describes the predicate, CONTRACT, and then its term, ATRIA using a logical definition. In particular, the event, contraction, is defined in terms of a more general event, restriction, and the features which distinguish contraction from other forms of restriction (in this domain contraction deals specifically with the restriction or pulling together of muscles). Similarly, the entity, ATRIA, is then defined with respect to its superclass, chamber, and its distinguishing feature, its location.

While the response in S1 may get the hearer to know about the predicate and its term which may enable them to understand the proposition, they still may be curious as to its cause. This is addressed by the second response, S2, which uses the plan operator in Figure 6.15 to indicate the cause of the event. If this were an action executed by an agent with a purpose, then the plan operator in Figure 6.16 could be used to indicate its purpose. The final response, S3, conveys the consequences of the event using the plan operator in Figure 6.17.

This example and section illustrate a range of communicative acts (formalized as plan operators) that can attempt to get the hearer to understand a proposition. This mirrors the strategy used in the human dialogue in Figure 6.13. Despite this range of techniques, there are several limitations. One problem is that if there are multiple causes, enablements, or motivations for an event or state (or equally if an event or state has multiple consequences) then some metric of saliency must be used to select among a variety of potential explications so that it is tailored to the user. TEXPLAN currently informs the hearer of all of them. Also, while the above short explications may be sufficient in some cases, in other situations (e.g., lectures, legal documents, etc.), longer, more complete explications may be necessary.

## 6.6 Summary

This chapter has examined expository text. In doing it has identified four principal expository forms: operational instruction, locational instruction, process exposition, and proposition exposition. Each of these expository forms is characterized as a series of communicative acts which are formalized as plan operators with associated constraints, preconditions, effects (on the cognitive state of the addressee), and decompositions. In some instances these decompositions include plan operators defined in previous chapters (e.g., entity description and event and state narration). These plan operators are used to produce hierarchical text plans which are realized as English text. These expository plan operators are carried forward into the subsequent chapter on argument which attempts to influence the beliefs or actions of the addressee, and which at times defines plan operators for argument in terms of descriptive, narrative, and expository plan operators.

The range and depth of testing varied among the different expository forms. Location identification and locational instructions were tested in the context of a large knowledge based cartographic system and over a hundred texts were produced. In contrast, testing of operational instructions, process exposition, and proposition exposition was rather limited, typically relying on small, hand-encoded knowledge bases which were used to produce only a few texts. Nevertheless, the definitions of the plan operators which produced these expository forms were based on analysis of naturally occurring texts and were encoded using domain independent relations (e.g., enablement, cause, purpose).

There are several issues which require further investigation with regard to exposition. Regarding locational instructions, the semantics of knowing about locations are rather complex. For example, you can know about a locale, say by its reputation, but have no knowledge of its location. On the other hand you may know a generic distinguishing attribute of a locale (e.g., the shape of a Holiday Inn sign) and this enable you to find it even though you may never have been to the particular one your are seeking. The location plan operators in TEXPLAN do distinguish between knowing about an entity and knowing a particular attribute of that entity (e.g., its location). The semantics for knowing about were defined in Chapter 4 where an agent knows about an entity if they know its superordinate, attributes, subparts, subtypes, or purpose. However, a more formal account of know and know about are beyond the scope of this dissertation and require more sophisticated modeling of the user's knowledge (including, for example, default or stereotypical knowledge).

Another issue raised by locational instructions concerns spatial focus. In particular, this constraint holds promise for resolving or generating deictic references (e.g., choosing between the demonstratives "this" and "that"). The selection of deixis can been explained by relating the entity that is the current spatial focus (CSF) to the spatial focus of the previous utterance or by relating the CSF to the speaker's location (e.g., "here" versus "there"; "this" versus "that"). A final issue was raised when an attempt was made to produce lengthy (i.e., page-length) locational instructions. It became clear that additional mechanisms are required to produced extended instructions. For example, because of human attentional limitations it becomes necessary to abstract and/or summarize as well as repeat and remind the reader over longer stretches of prose. One method of redundancy or repetition is to combine text and graphics, an issue which is explored in Chapter 9.

Yet another area for further work concerns proposition exposition. A natural but large step from proposition exposition is the notion of idea exposition. This can be accomplished, in part, by using the descriptive plan operators defined in Chapter 4. For example, we can get the hearer to know about a concept by using definition, detail, division, comparison/contrast, and analogy. But "idea exposition" actually implies some more sophisticated analysis or synthesis of an idea. One strategy would be to present major assumptions, principal consequences, and the relationship to other ideas. For example, the concept of "democracy" can be related to "freedom", "liberty", "self-determination", "equality", "inherent/unalienable rights", "majority rule". A systematic analysis of idea expositions needs to be performed to undercover organizational principles and rhetorical techniques that underlie idea exposition.

Having detailed how TEXPLAN elucidates operations, processes, and propositions, the next chapter turns to techniques that convince the user of a proposition or persuade them to act, i.e., argument. Argument has strong ties to exposition because a precondition of the user believing something is that they understand it (excluding counter examples such as "blind faith"). Similarly, even if a speaker succeeds in persuading someone to act, they must know how to execute the task to be successful, and hence a reliance on operational and locational instruction. Therefore, the next chapter considers argument.

# Chapter 7

# ARGUMENT

In a republican nation, whose citizens are to be led by reason and persuasion and not by force, the art of reasoning becomes of first importance.

*Thomas Jefferson*

## 7.1 Introduction

The previous three chapters have characterized several types of descriptive, narrative, and expository text as a series of communicative acts which were then formalized as plan operators. This chapter examines a final type of text, *argument*. In contrast to description, narration, and exposition, the purpose of argument is either to convince the hearer of a proposition or to persuade the hearer to act. Argument may employ the previous text types, for example to define terms (i.e., entities) or to explain propositions.

Aristotle claimed effective argument relies on three distinct constituents: ethos, pathos, and logos. Ethos refers to the moral character or values of the speaker which motivate the argument. Pathos is the emotional appeal or passion of the argument, in particular its emotional impact on the audience. Finally, logos is the logical basis of the argument, for example its use of *enthymeme* (a truncated syllogism), *exemplum* (example), and *sententia* (maxim). An Aristotelian example of the latter is "No man who is sensible ought to have his children taught to be excessively clever". Ethos, pathos, and logos are intertwined, for example the choice of maxims will disclose the ethos of the speaker. Classical logicians (e.g., Baum, 1981) and rhetoricians (e.g., Brooks and Hubbard, 1905; Brown and Zoellner, 1968) similarly enumerate a number of general techniques which can be used to convince or persuade the hearer (e.g., tell advantages, then disadvantages). I should emphasize that while the logical rules of deduction like those underlying syllogism are a generic apparatus defining legitimate reasoning which may be exploited for argument, what I am concerned with here is argument in a broader sense than this as illustrated, for example, by the matters addressed in classical discussions of rhetoric. In addition to discussing general argument forms (e.g., deduction and induction), they also indicate presentational strategies such as give the argument which will attract attention first and the most persuasive one last. While these ideas are suggestive, they are not formalized precisely enough to form the basis for a computational theory. This chapter, in contrast, formalizes and illustrates the computational implementation of a suite of argumentative techniques as plan operators.

Argument, like the previous types of text, is a goal-based activity which employs a range of rhetorical, illocutionary, and surface speech acts. TEXPLAN produces three principal forms of argument: *deduction*, *induction*, and *persuasion*. The first two are used to convince the hearer to believe a proposition and the latter is used to persuade the hearer to act.

Figure 7.1 shows TEXPLAN's top-level plan operator for arguments, `argue-for-a-proposition`, which argues for the truth of a proposition. The plan operator has the intended effect of getting the hearer to believe the proposition. This effect is achieved by claiming the proposition, optionally explaining it (using plan operators defined in the previous chapter), and finally attempting to convince the hearer of its validity. The first communicative act in the decomposition, `Claim`, is defined as the `claim-proposition-by-inform` plan operator in Figure 7.2. A claim consists simply of informing the hearer of the proposition, the intended effect being that the hearer believes the speaker believes the proposition. Of course the hearer may not believe the proposition themselves. To achieve this, the speaker must convince them of it.

Two types of reasoning can convince a hearer to believe a proposition: deduction and induction. The former moves top-down, from general truisms to specific conclusions whereas the latter builds arguments bottom-up, from specific evidence to a general conclusion. The next section formalizes deduction while the one after that formalizes induction within the TEXPLAN framework, specifically for arguments with the structure of Figure 7.1. Because they are the basis for several plan operators, the next section details deductive rules of inference, although no contribution to reasoning strategies is claimed.

```
NAME              argue-for-a-proposition
HEADER            Argue(S, H, proposition)

CONSTRAINTS       Proposition?(proposition)
PRECONDITIONS
  ESSENTIAL       KNOW-ABOUT(S, proposition) ∧
                  WANT(S, BELIEVE(H, proposition))

  DESIRABLE       ¬ BELIEVE(H, proposition)

EFFECTS           BELIEVE(H, proposition)

DECOMPOSITION     Claim(S, H, proposition)
                  optional(Explain(S, H, proposition))
                  Convince(S, H, proposition)
```

Figure 7.1 Top-Level, Uninstantiated `argue-for-a-proposition` Plan Operator

```
NAME              claim-proposition-by-inform
HEADER            Claim(S, H, proposition)

CONSTRAINTS       Proposition?(proposition)
PRECONDITIONS
  ESSENTIAL       WANT(S, BELIEVE(H, BELIEVE(S, proposition)))
  DESIRABLE       nil

EFFECTS           BELIEVE(H, BELIEVE(S, proposition))

DECOMPOSITION     Inform(S, H, proposition)
```

Figure 7.2 `claim-proposition-by-inform` Plan Operator

## 7.2 Deductive Argument

*Deductive* argument moves from general to particular. The classical form of deduction is *syllogism*, which attempts to prove the truth of a proposition by asserting a major and minor premise which together imply a conclusion. For example:

| **Syllogism** | **Example** | **Logical Form** |
|---|---|---|
| major premise | All men are mortal. | $\forall x$ man(x) $\supset$ mortal(x) |
| minor premise | Socrates is a man. | man(Socrates) |
| conclusion | Therefore, Socrates is mortal. | mortal(Socrates) |

This categorical syllogism first makes a general statement true of all members of some class (*major premise*), then states that the individual being considered (the term of the proposition) is a member of that class (*minor premise*), and finally, concludes that the general statement made about the class can be applied to the specific instance (*conclusion*). Categorical syllogisms come in other forms such as:

| **Syllogism** | **Example** | **Logical Form** |
|---|---|---|
| major premise | All men are mortal. | $\forall x$ man(x) $\supset$ mortal(x) |
| minor premise | God is not mortal. | $\neg$ mortal(God) |
| conclusion | Therefore, God is not a man. | $\neg$ man(God) |

A syllogism is valid if and only if no argument of that form can have true premises and a false conclusion (although a syllogism can in fact have false premises and a true conclusion). That is, no terms can be substituted into the syllogism such that the premises are true and the conclusion false. When one of the premises is dropped, this is termed an *enthymeme* which literally means "in mind" since the dropped premise is assumed to be inferable or in the mind of the hearer (the more general concept is called "modus brevis" whereby any portion of the deduction is dropped). Enthymemes occur frequently in naturally occurring arguments, especially if one (or both) of the premises can be inferred by the hearer. This is important both for an interpreter which needs to fill in a missing premises when analyzing arguments (cf. Cohen, 1986) and for a generator which can omit them from the arguments it produces when it believes they are unnecessary. The principal classes of syllogism are categorical, disjunctive, and hypothetical (also called conditional).

Syllogisms rely on basic rules of inference. Most categorical syllogisms use two rules of inference. The Socrates categorical syllogism is based on *modus ponens* (affirm antecedent) and the God example is based on *modus tollens* (deny consequent). The most common forms of logical inferences, both well-formed and ill-formed, are shown in Figure 7.3 (all rules except for 5 and 6 adapted from Cohen, 1986, p. 9). Another example of modus tollens is "All bachelors are single. Joe is not single. Therefore, Joe is not a bachelor."

Figure 7.3 relates inference rules with syllogistic forms. While inference rules 1 and 2 in Figure 7.3 are used by categorical syllogism, rules 3 and 4, *modus tollendo ponens* and *modus ponendo tollens*, are used for *disjunctive syllogism*. Modus tollendo ponens (rule 3) denies one member of a conjunction and then asserts the other, as in "Someone is male or they are female. Socrates is not female. Therefore, Socrates is male." In contrast modus ponendo tollens (rule 4) asserts one member of a conjunction and then denies the other, as in "Someone is either married or single. John is single. Therefore, John is not married" or "A person is dead or alive. I am alive. Therefore, I am not dead." Finally, *hypothetical syllogism* (rule 5, also called conditional or "if-then" syllogism) is illustrated by "If Bush is elected he will support education. If Bush supports education he will raise taxes. Therefore, if Bush is elected he will raise taxes." While deductive plan operators in TEXPLAN are currently limited to categorical syllogism based on inference rules 1 and 2, the plan operators could be extended to incorporate other inference rules. Even some complex inference chains could be formalized as plan operators since they rely on these basic inferences.

Cohen (1986) suggests representing the first four inferences in Figure 7.3 as frames to recognize arguments (as opposed to generating them) given that the argument may not be presented in the "standard" order, i.e., the minor premise or even conclusion may precede the major premise. While Cohen did not implement her ideas, she suggested how clue words (e.g., "therefore", "and", "so") could be used to recognize the structure underlying arguments that are presented in "pre-order" (i.e., claim followed by evidence), "post-order" (evidence before claims), and "hybrid-order" format (using both pre-order and post-order). Unlike argument recognition, argument generation need not handle ill-formed inference (although some invalid inferences can be very convincing), but it does need to be able to vary presentational order. While TEXPLAN's plan operators produce pre-order arguments, they could be easily modified to produce post-order and hybrid arguments (e.g., a post-order argument might be used to build up to a claim that the speaker knows the hearer does not believe).

| WELL-FORMED | MAJOR PREMISE | MINOR PREMISE | CONCLUSION | Syllogism |
|---|---|---|---|---|
| 1. modus ponens | P ⊃ Q | P | Q | categorical |
| 2. modus tollens | P ⊃ Q | ¬ Q | ¬ P | categorical |
| 3. modus tollendo ponens | P ∨ Q | ¬ P | Qdisjunctive | |
| 4. modus ponendo tollens | P ∨ Q | Q | ¬ Pdisjunctive | |
| 5. hypothetical syllogism | P ⊃ Q | Q ⊃ R | P ⊃ R | hypothetical |
| 6. hypothetical syllogism | P ⊃ Q | R ⊃ ¬ Q | R ⊃ ¬ P | hypothetical |

| ILL-FORMED | MAJOR PREMISE | MINOR PREMISE | CONCLUSION |
|---|---|---|---|
| 7. asserting consequent | P ⊃ Q | Q | P |
| 8. denying antecedent | P ⊃ Q | ¬ P | ¬ Q |

**KEY:** "⊃" means "implies"; "¬" negation ; "∧" conjunction; "∨" disjunction

Figure 7.3 Inference Classes

Figure 7.4 shows TEXPLAN's `convince-by-categorical-syllogism-modus-ponens` plan operator which proves a proposition using the modus ponens inference rule. The effect of the plan operator is to get the hearer to believe the proposition by indicating a major premise, minor premise, and conclusion. As in the Socrates example, the first statement is a logical implication, the last two are simply propositions. In a logic-based application the plan operator could use theorem proving to find an inference chain to support the proposition. However, the plan operators were tested using FRL (Roberts and Goldstein, 1977), which has non-monotonic reasoning facilities such as automatic inheritance. Thus the constraints on the plan operator in Figure 7.4 dictate that the term of the proposition it is attempting to prove must have a superclass in the knowledge base. In the Socrates example, the term of the claim `Mortal(Socrates)` is `Socrates`, which can be extracted from the proposition and used to retrieve its superclass, `Man`, from the generalization hierarchy in the knowledge base. The essential preconditions of the plan operator make sure the speaker is familiar with this class, and then examine all instances of this class to see if the predicate of the proposition, `Mortal`, holds for them. If all these preconditions are satisfied, then the plan operator can be used. The decomposition of the plan operator first states the major premise, a universal definition such as "All men are mortal." Next, it indicates the minor premise, a logical definition such as "Socrates is a man". It concludes by simply informing the hearer of the initial claim, for example "Socrates is mortal".

```
NAME            convince-by-categorical-syllogism-modus-ponens
HEADER          Convince(S, H, proposition)

CONSTRAINTS     Proposition?(proposition) ∧
                ∃c |Superclass(term(proposition), c)
PRECONDITIONS
  ESSENTIAL     ∀x ∈ instances(superclass)
                     predicate(x)[1] ∧
                KNOW(S, Universal-Definition(superclass, predicate))

  DESIRABLE     KNOW-ABOUT(H, entity) ∧
                ∃c | Superclass(entity, c) ∧KNOW-ABOUT(H, c) ∧
                ¬ KNOW(H, Universal-Definition(superclass, predicate))

EFFECTS         KNOW-ABOUT(H, entity) ∧
                ∀x ∈ superclasses(entity)
                     KNOW(H, Superclass(entity, x)) ∧
                ∀y ∈ differentiae(entity)
                     KNOW(H, Differentia(entity, y)) ∧
                KNOW(H, Universal-Definition(superclass, predicate))

DECOMPOSITION   Inform(S, H, Universal-Definition(superclass, predicate))
                Inform(S, H, Logical-Definition(entity))
                Inform(S, H, Conclusion(proposition))

WHERE           entity = term(proposition)
                predicate = predicate(proposition)
                superclass = c | Superclass(entity, c) ∧KNOW-ABOUT(S, c)
```

Figure 7.4 `convince-by-categorical-syllogism-modus-ponens` Plan Operator

To test this plan operator, a knowledge base representing entities and relationships from the Socrates example was developed, schematically illustrated in Figure 7.5. In response to the goal `BELIEVE(H, Mortal(SOCRATES))`, TEXPLAN uses the above defined plan operators to produce the text plan in Figure 7.5 and the corresponding surface form. The hierarchical text plan in Figure 7.5 is a decomposition of communicative acts. In particular, the top-level communicative act, argue, decomposes into a claim followed by a convince act. The convince act is further decomposed into the assertion of a universal definition (the major premise, "All men are mortal.") followed by an assertion of a logical definition (the minor premise, "Socrates is a man.") and finally the assertion of the conclusion. Because the hierarchical text plan captures the communicative function of the different types of content in the text plan (e.g., universal-definition, logical definition, conclusion) this can be used to signal the information structure to

[1] The plan operators in the implementation actually use a function which takes a predicate along with terms and returns a composed proposition.

the hearer. For example, in the final utterance the linguistic realizer signals the conclusion with the connective "therefore".

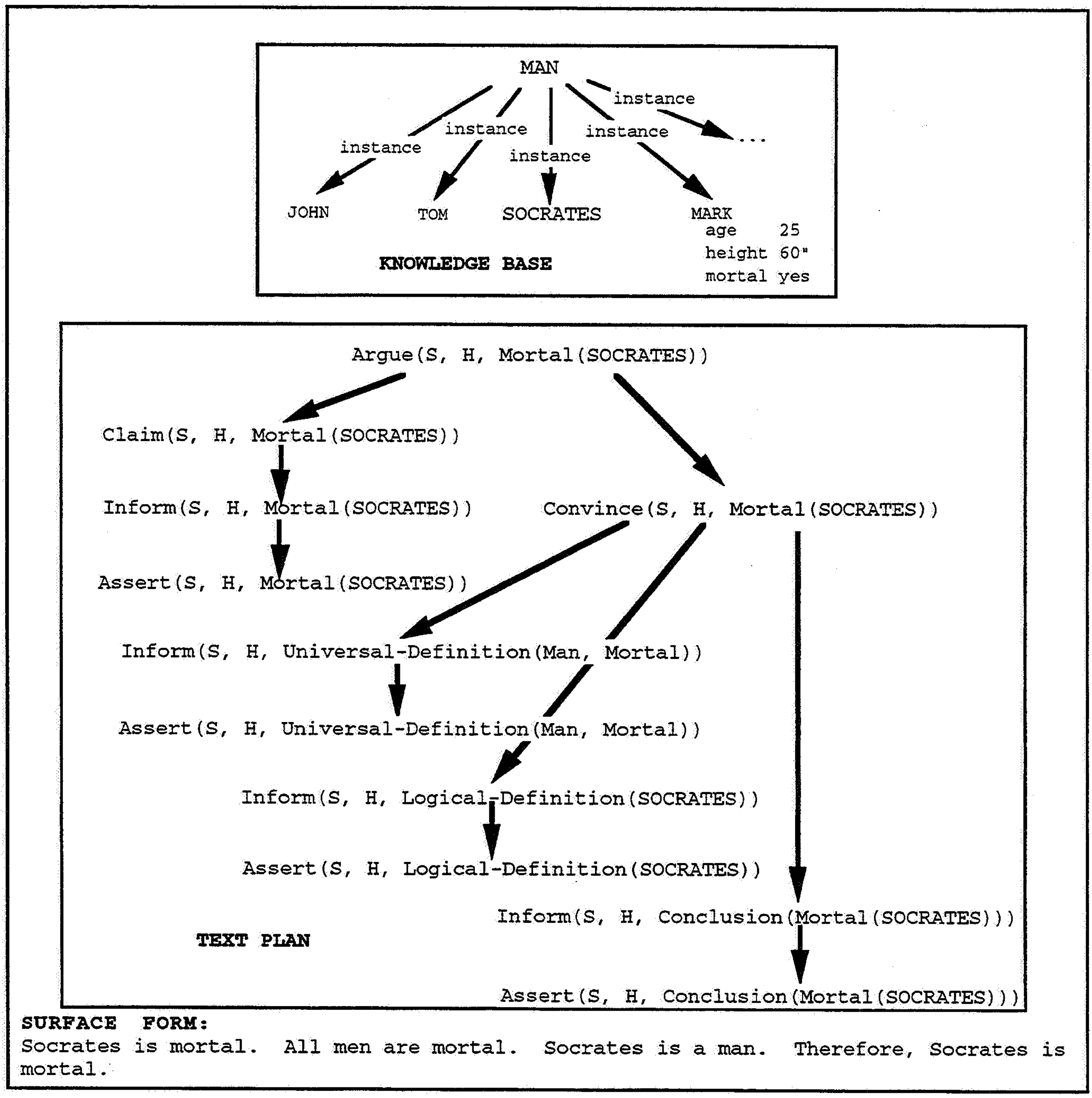



Figure 7.5 Socrates Syllogism Text Plan

In contrast to categorical syllogism based on modus ponens, in a categorical syllogism based on modus tollens a negated proposition ($\neg P$) is proved valid by asserting a major premise as before ($P \supset Q$)

but instead with a negated minor premise (¬Q). For example the claim "God is not a man" can be supported by the major premise "All men are mortal" and the minor premise "God is not mortal". The `convince-by-categorical-syllogism-modus-tollens` plan operator is shown in Figure 7.6. For example, given the proposition `¬Man(GOD)`, the precondition of the plan operator finds all instances for which the predicate, Man, is true. All common properties of these individuals (i.e., all features which are true of all men) are collected. These properties yield a set of universal statements about men (e.g., "All men are male", "All men are mortal", etc.). This set of properties can then be compared to the set of properties true of the term of the given proposition (i.e., GOD). Any property not true of the term GOD but true of all individuals in the set (of all men) can be used as the minor premise. Therefore, the major premise, a universal statement, simply indicates this property is true of all individuals (i.e., all men) while the minor premise states how the term GOD fails to possess this property (e.g., "God is not mortal."). Therefore, it can be concluded that the term is not a member of the class (e.g., "God is not a man."). The resulting text structure is very similar to that of Figure 7.5.

While more sophisticated syllogisms may seem complex, they are often based on standard patterns of inference. Consider the following two syllogisms (from Baum, 1981, p. 200 and p. 204, respectively) where "some" is interpreted as "at least one".

| **Example 1** | **Logical Form** |
| --- | --- |
| All bacteria are organisms visible through a light microscope. | $\forall x$ bacteria(x) $\supset$ visible(x) |
| No viruses are organisms visible through a light microscope. | $\forall x$ virus(x) $\supset \neg$ visible(x) |
| Therefore, no viruses are bacteria. | $\forall x$ virus(x) $\supset \neg$ bacteria(x) |

| **Example 2** | **Logical Form** |
| --- | --- |
| Some relatives are friends. | $\exists x$ relative(x) $\supset$ friend(x) |
| No friends are enemies. | $\forall x$ friend(x) $\supset \neg$ enemy(x) |
| Therefore, some relatives are not enemies. | $\exists x$ relative(x) $\supset \neg$ enemy(x) |

While these have not been implemented, the first uses the hypothetical syllogism rule 6 in Figure 7.3 and the second is a variation on hypothetical syllogism using quantification. These above forms can be similarly formalized as plan operators, particularly if the underlying application is logic based. At this point it is important to emphasize, however, that the focus here is not on the process of reasoning itself and the complexities therein (e.g., close world assumptions), but rather with the way some argument structure (e.g., modus ponens) relates to a hierarchical text plan and its linearization as an English text.

| | |
|---|---|
| NAME | `convince-by-categorical-syllogism-modus-tollens` |
| HEADER | Convince(*S*, *H*, *proposition*) |
| CONSTRAINTS | Proposition?(*proposition*) ∧ Negated?(*proposition*) |
| PRECONDITIONS | |
| ESSENTIAL | KNOW(*S*, Universal-Definition(*predicate1*, *predicate2*)) |
| DESIRABLE | ¬ KNOW-ABOUT(*H*, *entity*) ∧<br>¬ KNOW(*H*, Universal-Definition(*predicate1*, *predicate2*)) |
| EFFECTS | KNOW-ABOUT(*H*, *entity*) ∧<br>KNOW(*H*, Universal-Definition(*predicate1*, *predicate2*)) |
| DECOMPOSITION | Inform(*S*, *H*, Universal-Definition(*predicate1*, *predicate2*))<br>Inform(*S*, *H*, ¬ *predicate2*(*entity*))<br>Inform(*S*, *H*, Conclusion(*proposition*)) |
| WHERE | *entity* = term(*proposition*)<br>*predicate1* = predicate(*proposition*)<br>*predicate2* = *property* \| ∀x *predicate1*(*x*) ∧ *property*(*x*) ∧<br>¬ *property*(*entity*)[2] |

Figure 7.6 `convince-by-categorical-syllogism-modus-tollens` Plan Operator

In argument it is often not sufficient to simply apply rules of inference. A speaker must also argue for the validity of a rule itself and its applicability to the case at hand in order to truly convince the hearer to believe it (belief, while used throughout this dissertation and particularly in argument operators in its absolute sense, should rather be viewed as a degree of belief). Also the manner of presentation is important in an argument, and an argument may thus require an organization beyond what is necessary for the logic of the argument alone. Toulmin (Toulmin, 1958; Toulmin, Rieke, and Janik, 1979) suggests the model of argument structure shown in Figure 7.7. A general inference of the form $P \supset Q$ is termed a *warrant*. A warrant can be instantiated with *grounds* which match the antecedent of the rule to yield the *claims*, the instantiated consequent of the rule. *Backing* supports the credibility or correctness of the warrant by providing additional argument or supporting evidence, *modality* or qualifier indicates the degree of support for a claim, and finally *rebuttals* indicate counter argument, counter evidence, exceptions, or special conditions, which may refute the claim, discount it, or qualify it in some way. Unfortunately, Toulmin does not formalize backings or rebuttals except to indicate that they affect the hearer's belief in the inference. In contrast, Neches et al. (1985), detailed in Chapter 2, take the view that domain principles and domain knowledge serve as backings for domain inferences but that these backings may not simply be taken for granted as assumed domain knowledge and may need to be explicitly indicated. Similar to

[2]In frame or object-oriented knowledge bases this amounts to examining the attributes and attribute-value pairs of entities.

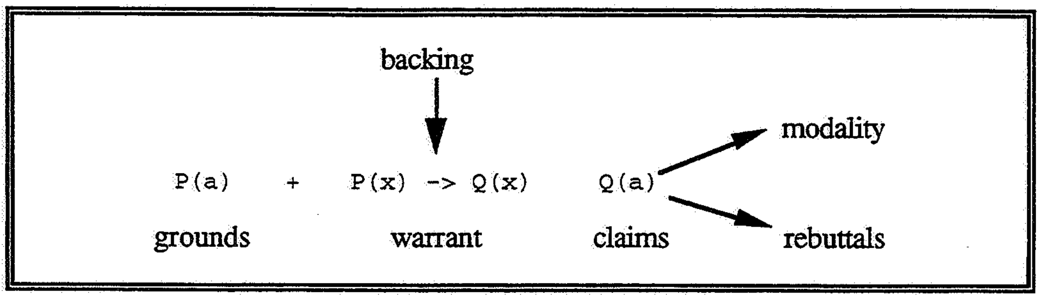



Figure 7.7 Toulmin Model of Argument Structure

Neches et al., TEXPLAN supports claims by instantiating warrants in the form of deductive arguments which are backed by inductive arguments bearing on grounds. This section has focused on warrants; the next section details backings.

Bench-Capon et al. (1990) discuss the application of Toulmin's model of argument structure to the explanation of logic programs, extending Toulmin's single inference model to characterize chains of inference. By annotating the clauses in the bodies of the rules of a logic program to indicate the various roles they play (i.e., ground (or data), claims, rebuttal, warrant, or backing), his program is able to order explanation content according to the type of information it embodies (e.g., present the data followed by the warrant and rebuttal). He illustrates how this improves upon traditional traces of inference chains.

Just as Toulmin suggests a structure for argument, Birnbaum, Flowers and McGuire (1980) and Birnbaum (1982) argue that there are two ways in which propositions in an argument can relate to one another: support or attack. They represent these propositions as nodes in an *argument graph* connected by attack and support relations. Once a program (not detailed) has interpreted this structure, they suggest three ways to attack an argument (called *argument tactics*): attack the main proposition, attack the supporting evidence, and attack the claim that the evidence supports the main point (Toulmin's "backing" above). Unfortunately, no computational details are given and the two relations of support and attack do not provide as rich an argument structure as in Toulmin's model. Finally, it is important to distinguish between producing an argument and debating a point. Debate is beyond the scope of this dissertation as it requires, among other things, richer models of argument strategies and tactics, and can benefit from computational techniques such as case based reasoning.

Dialogue is fundamental to debate, and natural dialogue (i.e., spontaneous, casual conversation) was the focus of Reichman (1981ab). Reichman characterizes discourse using a number of conversational moves (e.g., support, interrupt, challenge) which she claims underlie all forms of prose (e.g., narration, exposition, and argument). She claims that "clue words" such as "because", "but anyway" and "no but" signal these moves. Conversational moves are represented as the arcs in an ATN that captures a "discourse grammar", i.e., a network of legal moves in a dialogue. Reichman also introduces the notion of "context spaces"—hierarchical segmentations of utterances—and shows how conversational moves relate to, for example, context space suspension and resumption (e.g., "conceding a subargument but continuing a debate, necessarily entails popping back to one of the context spaces that generated the subargument" (Reichman, 1981b, p. 199)).

Unlike Reichman's work, TEXPLAN does not address discourse moves that control or direct a dialogue, indeed the focus of this dissertation is on generating multisentential text rather than characterizing conversation. In addition, a key difference between Reichman's model and the communicative acts formalized in TEXPLAN is that the effects of Reichman's conversational moves are not defined with respect to the cognitive or psychological state of the hearer, but rather "an act's preconditions stem from the preceding discourse structure, and its effects are on this discourse structure" (Reichman, 1981b, p. 235). In contrast, a principal claim of this dissertation is that communicative acts (i.e., rhetorical, illocutionary, and surface speech acts) and communicative goals (i.e., effects on the knowledge, beliefs, or desires of the hearer) are inextricably tied, so one cannot be considered without the other. In Reichman's work, communicative acts and communicative goals are not linked, and her conversational moves are not related to higher-level physical or linguistic actions (e.g., argue by informing the hearer of evidence, get the hearer to perform a physical act by requesting and persuading). While this dissertation makes the important connection between a range of communicative acts and their effects on the cognitive state of the addressee (i.e., their knowledge, beliefs, and desires), it makes no claims concerning the accurate representation, maintenance, and revision of beliefs and intentions, as this remains an active research area (cf. Cohen and Levesque, 1985; Galliers, 1989). Thus I have treated beliefs in the context of the generation of arguments in a fairly straightforward manner.

While a conversant has the advantage of immediate feedback to direct his or her utterances, there are many instances in which there is no immediate feedback (e.g., television or radio advertisement), so a writer of prose must compose an argument carefully to ensure success. As Toulmin's model above illustrates, one cannot necessarily convince the addressee by making claims based on warrants and grounds. General rules or statements must be supported by backings. Therefore, the next section defines several inductive techniques as plan operators which can be used to back general rules (e.g., supporting the premises of deductive arguments), or which can be used simply to support a claim.

## 7.3 Inductive Argument

While deductive techniques such as the syllogism work from general statements to particular ones, *inductive* arguments are defined here in a broad sense as all non-deductive arguments. This includes induction in the more narrow sense, that is providing particular instances to support general claims. This is analogous to the scientific method which examines a number of examples and from these attempts to develop generalizations. Induction has a close tie to deduction when the premises of deduction originate from examination of a number of specific cases in the world. For example, the major premise "All men are mortal" may be motivated by observations of individual men's life spans over the centuries, and the minor premise "Socrates is a man" by the observation or evidence of Socrates' physical characteristics (e.g., his picture). In practice, therefore, deduction may be no more powerful than the inductive base for its premise allows. Even if it is logically well-formed, a deduction may yield an invalid conclusion if its premises are false because they are based on poor inductive reasoning.

The best way to convince a hearer of a proposition using induction is to provide enough evidence to support it. *Evidence* is defined as support for a proposition, in particular a sign or indication of a state or event (e.g., "His flushed look was visible evidence of this fever."). *Counter evidence* is evidence that indicates that some state or event is not the case. Figure 7.8 shows the `convince-by-evidence` plan operator used in TEXPLAN which attempts to increase the hearer's belief in some proposition. The decomposition of the plan operator first concedes counter evidence and then informs the hearer of supportive evidence. When detailing supportive evidence, the decomposition optionally recurses to convince the hearer of the validity of the evidence if the speaker believes the hearer does not believe the supporting evidence. To accomplish this the plan operator uses a conditional construct (e.g., **if** state **then** action). Evidence is ordered according to its degree of importance so that least important evidence is followed by more convincing evidence. For example in a medical diagnosis domain this might correspond to the degree of relevance and certainty of evidence supporting a diagnosis. In a political debate it might be the saliency of statistics which serve as evidence of an opponents flawed economic policy. The strength with which evidence supports a claim is a function of the relevancy, accuracy, and completeness of the evidence. While this is explicit in some domains (e.g., probabilistic medical diagnosis systems), it may be implicit in others. Therefore, the domain-independent plan operator in Figure 7.8 assumes a function, `order-by-importance`, that can order the evidence according to its importance. The decomposition of the plan operator first concedes any counter evidence and then presents evidence supporting the proposition. This is reminiscent of McCoy's (1985ab) misconception correction strategy, detailed in Chapter 2, which first denies a false proposition, P, that the hearer claims, next states some competing proposition, Q, which is the correct version of P, then concedes evidence supporting P, but then finally overrides this with counter evidence supporting Q. The focus of McCoy's work, however, was on recovering from misconceptions and her strategies do not (except implicitly) indicate that they convince the hearer or change their beliefs.

```
NAME            convince-by-evidence
HEADER          Convince(S, H, proposition)

CONSTRAINTS     Proposition?(proposition) ∧ ∃x Evidence(proposition, x)
PRECONDITIONS
  ESSENTIAL     ∃x | Evidence(proposition, x) ∧
                    KNOW(S, Evidence(proposition, x))

  DESIRABLE     ∃x | Evidence(proposition, x) ∧
                    ¬ KNOW(H, Evidence(proposition, x))

EFFECTS         ∀x ∈ contra-evidence(proposition)
                   KNOW(H, Counter-Evidence(proposition, x)) ∧
                ∀x ∈ evidence(proposition)
                   KNOW(H, Evidence(proposition, x))

DECOMPOSITION   ∀x ∈ order-by-importance(contra-evidence(proposition))
                   Concede(S, H, Counter-Evidence(proposition, x))
                ∀x ∈ order-by-importance(evidence(proposition))
                   Inform(S, H, Evidence(proposition, x))
                   optional(if BELIEVE(S, ¬ BELIEVE(H, x)) then
                                 Convince(S, H, x))
```

Figure 7.8 `convince-by-evidence` Plan Operator

The plan operator in Figure 7.8 was tested in the context of justifying conclusions in the medical consultation and diagnosis system, NEUROPSYCHOLOGIST (Maybury and Weiss, 1987). The system simulates neuropsychological diagnosis, an approach to identifying neuropsychological dysfunction in a given patient. NEUROPSYCHOLOGIST reasons about evidence which comes in the form of physical tests (e.g., from simple blood pressure tests to more sophisticated CAT scans) as well as patient behavior, measured by standardized tests and clinical observations of the patient performing perceptual or memory tasks. As Figure 7.9 illustrates, the system first consults, then diagnoses, and finally explains its conclusions. Diagnosis in the system simulates that of a neuropsychologist. After collecting the empirical data (test scores) and subjective data (clinical and qualitative observations), a neuropsychologist attempts to match the symptoms with particular categories of cerebral disorders.

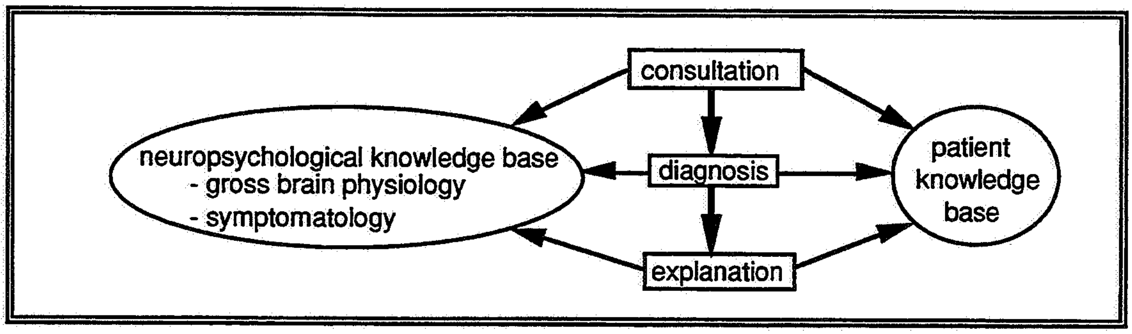



Figure 7.9 NEUROPSYCHOLOGIST System Overview

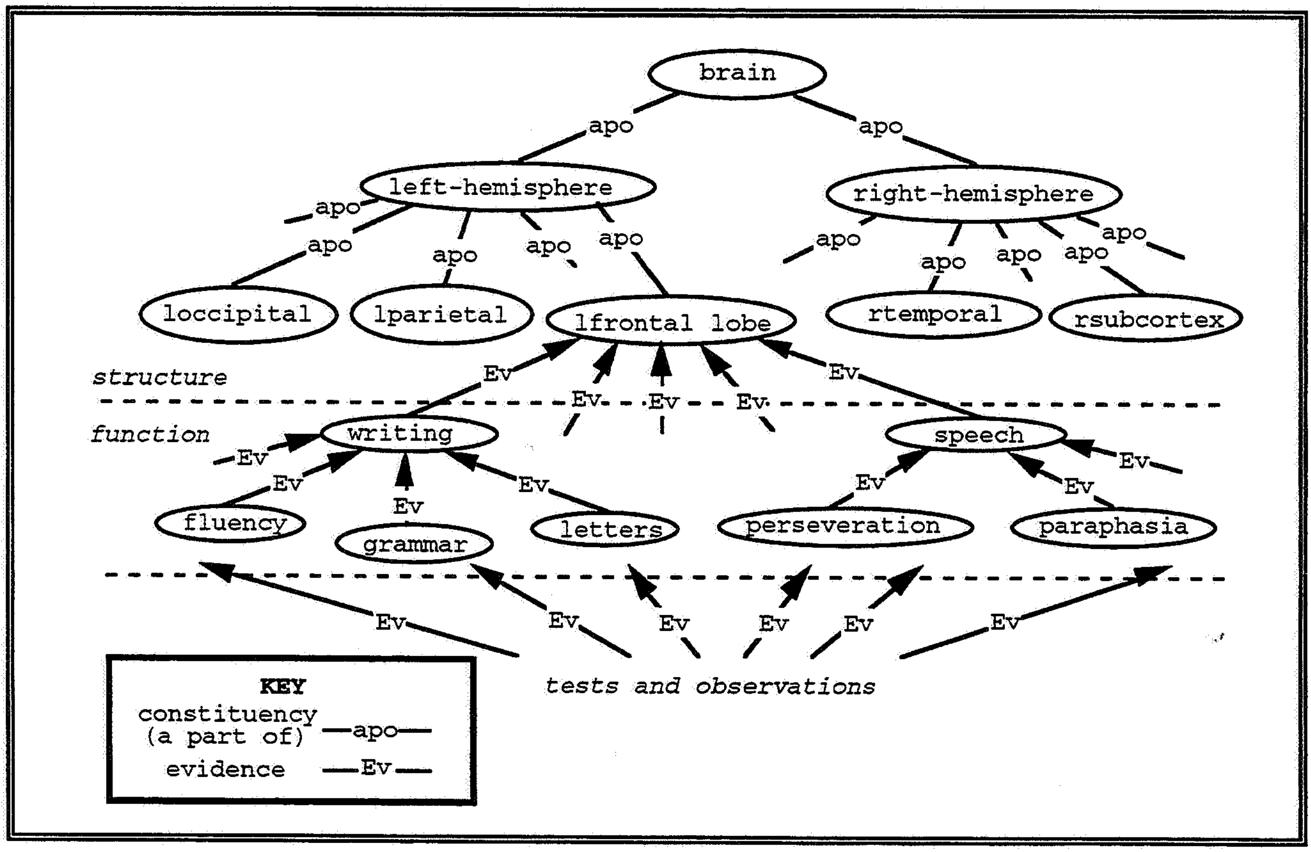



Figure 7.10 Brain Structure/Function Hierarchy

A domain expert, Dr. Charles Weiss, described his problem-solving method as producing a mental image of a brain, with millions of tiny lights of variable brightness attached to different regions. When a particular test or observation suggested dysfunction in a particular region, the corresponding light would increase in brightness. At the end of the analysis, a density of brightness would indicate the most probable area of damage. If the patient suffered solely from a focal disease such as a stroke, only that region affected by the lesion would be brightly lit. In the case of a global dysfunction, such as in Alzheimer's disease, the entire brain would glow.

NEUROPSYCHOLOGIST performs diagnosis in a similar fashion by instantiating a knowledge base of 142 hierarchically-organized frames which relate gross neurophysiology to symptomatology (i.e., the symptom complex of a disease). The system has two hierarchical models: one of the brain (the *structure/function hierarchy*) and one of cognitive disorders (the *symptom/disorder hierarchy*). These two models are instantiated using tests and observations from the user about a particular patient. The structure/function hierarchy, a model of the individual patient's brain, is constructed from tests and observations, the evidence from which are combined using Bayesian heuristics. Figure 7.10 illustrates a frame schema that relates tests and observations to specific lobal structures and functions. For example, tests which suggest paraphasia indicate speech impediments which are associated with the left frontal lobe. However, in contrast to this neurophysiological model, the results of tests and observations at the same

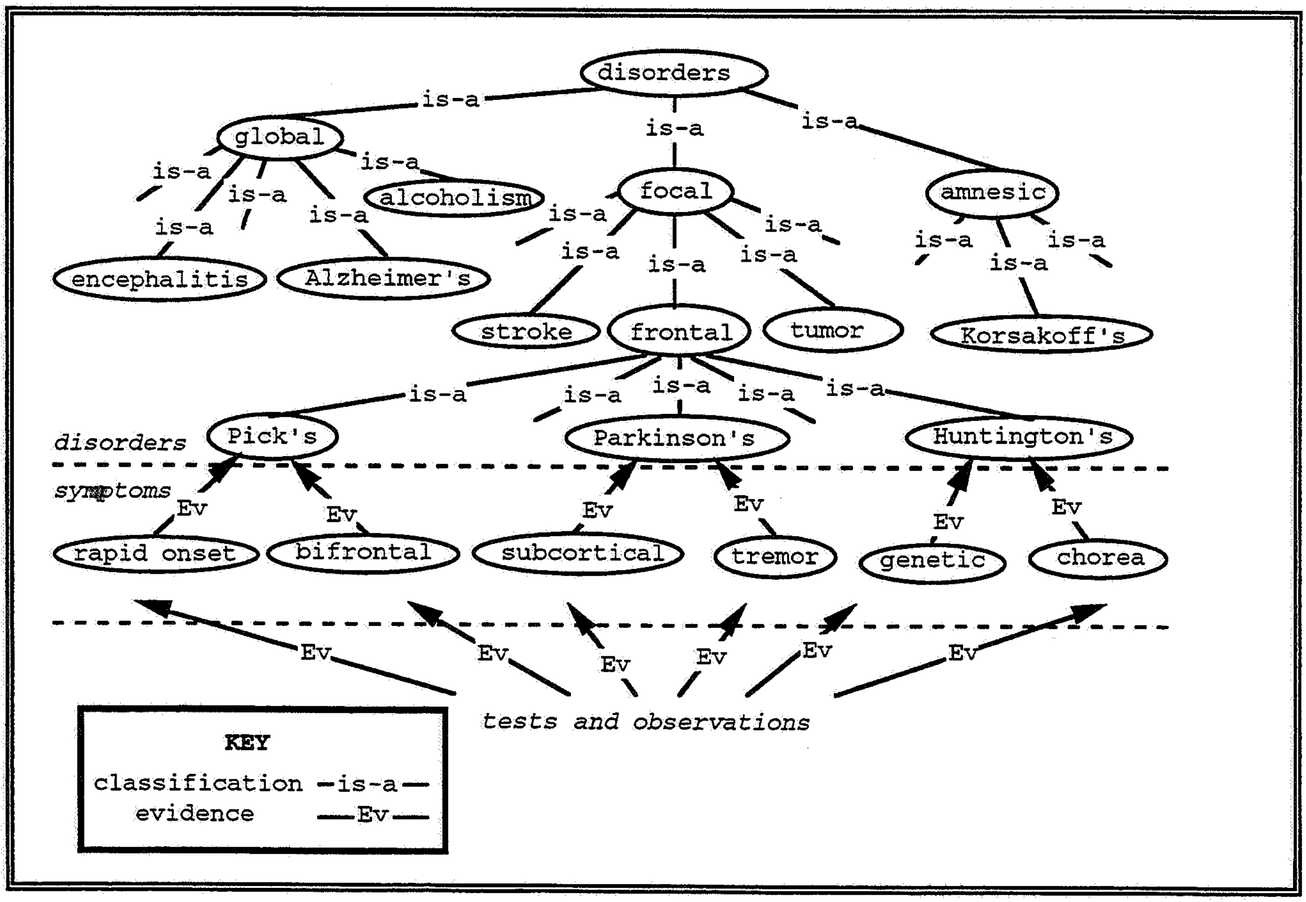



Figure 7.11 A Portion of the Symptom/Disorder Hierarchy

time instantiate the symptom/disorder hierarchy, partially shown in Figure 7.11. For example, different frontal lobe disorders (e.g., Pick's, Parkinson's and Huntington's disease) have associated symptoms which are identified by tests and clinical observations.

In the original system, after a fifteen minute to half an hour consultation with a domain expert about a patient, the system would post a diagnosis or claim followed by a listing of evidence or causes that support that claim. The user could then query the brain and disorder models by typing in keywords like "WHY" and "HOW" along with some entity (i.e., frame) in the underlying model. The system would then use the functions shown in Figure 7.12 to trace the underlying hierarchical model to justify conclusions about which part of the brain was damaged and which disorders the patient was most likely suffering from. Since NEUROPSYCHOLOGIST had no syntactic, semantic, or discourse level representation of natural language, explanations were essentially templates filled with variables. This approach is inadequate for reasons outlined in Chapter 2. TEXPLAN has, however, been used to generate output for NEUROPSYCHOLOGIST.

The following procedures justify NEUROPSYCHOLOGIST's inferences:

(HOW-BAD? *entity*) If *entity* is a brain area (e.g., left-occipital-lobe) it prints the extent of damage diagnosed. If *entity* is a cognitive disorder (e.g., Parkinson's), it prints the probability of that disorder. If *entity* is a test or observation it tells how well/poorly a patient scored on a particular test or clinical observation.

(WHY-DAMAGE? *entity*) If *entity* is a particular brain region that has damage, it justifies the damage in this region by moving <u>down one level</u> in the structure/function decomposition.

(WHY-DISORDER? *entity*) Analogous to WHY-DAMAGE? function. If *entity* is a disorder, this function prints out the reason(s) for determining that a patient has a particular disorder by moving <u>down one level</u> in the disorder/symptom decomposition.

(WHY-USEFUL? *entity*) Tells why a given symptom, function, test, or observation is useful in the diagnosis of organic brain disorders by indicating how that entity contributes to other entities <u>one level up</u> in the structure/function or disorder hierarchy.

Figure 7.12 Explanation Procedures for NEUROPSYCHOLOGIST

Thus using the output from a typical diagnosis, TEXPLAN's argument plan operators were tested to convince the hearer of a given diagnosis. In one session, the system has just diagnosed Korsakoff's disorder. When the user of NEUROPSYCHOLOGIST asks "Why did you diagnose Korsakoff's disorder?", simulated by posting the goal `BELIEVE(H, Has(KORSAKOFFS PATIENT1))`, TEXPLAN reasons about information in the symptom/disorder hierarchy using the plan operator in Figure 7.8 along with others to produce the text plan shown in Figure 7.13. In this text plan the variable *P* refers to the proposition `Has(KORSAKOFFS PATIENT1)` and the evidence relations are based on those shown in Figure 7.11. The hierarchical text plan (which embodies the communicative structure and order of the text) is realized as the English surface form:

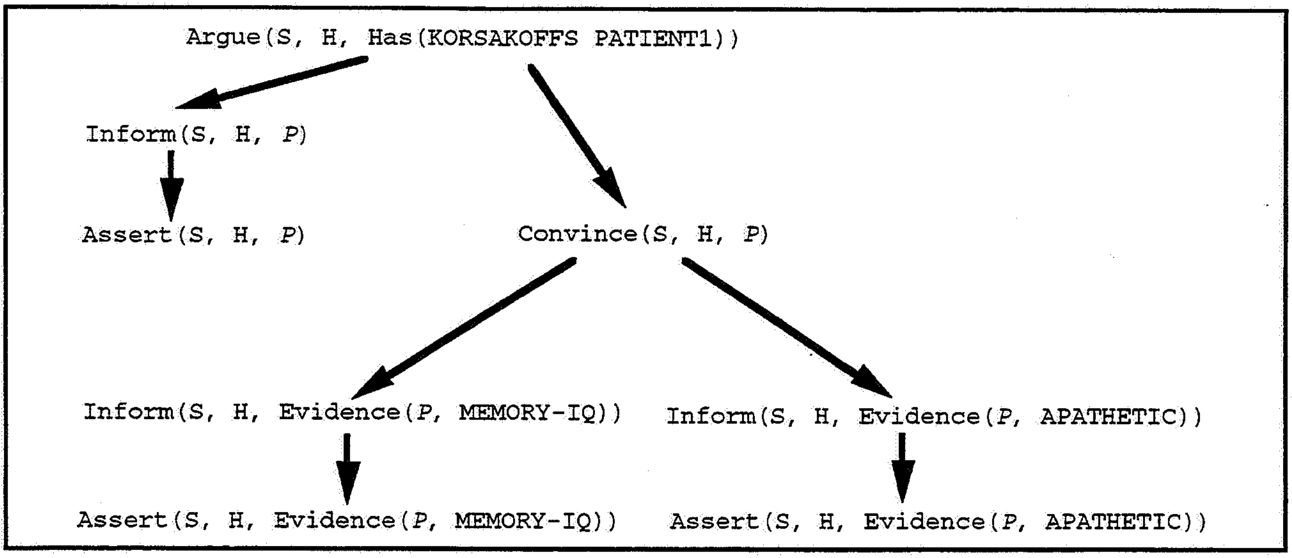



Figure 7.13 Diagnosis Text Plan

```
Patient1 has Korsakoff's disorder with 75% probability.  An apathetic
demeanor indicates a 70% probability of Korsakoff's disorder.  A poor
memory and low IQ scores indicates a 80% probability of Korsakoff's
disorder.
```

In contrast to asking questions about the symptom/disorder hierarchy, the user can query for information about the structure/function hierarchy, the system's model of the patient's brain physiology. For example, if the user asks "Why is the left frontal lobe damaged?", simulated by posting the goal `BELIEVE(H, Damaged(L-FRONTAL PATIENT1))`, TEXPLAN examines the brain structure/function hierarchy in Figure 7.10 to produce a text plan similar to the one above which is realized as:

```
Patient1 has left frontal lobe damage with 90% probability.  A loss of
mental control indicates a 85% probability of left frontal lobe damage.  A
loss of left cognitive flexibility indicates a 90% probability of left
frontal lobe damage.  A writing dysfunction indicates a 90% probability of
left frontal lobe damage.  A speech dysfunction indicates a 95% probability
of left frontal lobe damage.
```

In these examples the arguments do not concede any counter evidence because there is none in the underlying application system. This may of course not reflect the realities of the relevant domain, but at an epistemological level, TEXPLAN can perform no better than the application system.

While NEUROPSYCHOLOGIST represents structure and function, disorder and symptom knowledge, it has no underlying causal model of the domain. Unlike evidence (i.e., a sign or indication of a state or event) which increases the hearer's belief in a proposition, causal relations simply allow the possibility or probability of proof. Cause explicates rather than convinces. But by indicating a likely or actual cause of a proposition (a state or event), the hearer may better understand why or how the proposition came into being, even though they may not necessarily believe it. If I state "Mary is hurt" and support this by a cause "She fell down", this may make you understand how Mary could have got hurt, but not necessarily convince you that she in fact fell down or indeed was hurt. Evidence like, "I saw blood" or "she was crying", might convince not only of that fact that the event happened, but that Mary was indeed hurt.

This distinction between understanding how something could be the case and believing it is the case is used in the inductive plan operator, `convince-by-cause-and-evidence`, shown in Figure 7.14. The plan operator first explains what caused the event or state represented by a proposition and then increases the hearer's belief in the proposition by providing evidence. The constraints on the plan operator dictate that there must be both a cause and evidence for the proposition in the knowledge base. The preconditions state that the speaker must know at least one cause and one piece of evidence for the proposition. The decomposition first calls the `Explain-How` plan operator defined in the previous chapter (which details the preconditions, motivations, and causes of the proposition) and then informs the hearer of any evidence supporting the proposition (optionally convincing them of this).

```
NAME            convince-by-cause-and-evidence
HEADER          Convince(S, H, proposition)

CONSTRAINTS     Proposition?(proposition) ∧
                ∃x | Cause(x, proposition) ∧
                ∃x | Evidence(proposition, x)
PRECONDITIONS
  DESIRABLE     ∃x ∈ evidence  ¬ KNOW(H, Evidence(proposition, x))

EFFECTS         ∀x ∈ evidence  KNOW(H, Evidence(proposition, x))

DECOMPOSITION   Explain-How(S, H, proposition)
                ∀x ∈ evidence
                    Inform(S, H, Evidence(proposition, x))
                     optional(Convince(S, H, x))

WHERE           evidence =
                  order-by-importance(
                          ∀x | Evidence(proposition, x) ∧
                               KNOW(S, Evidence(proposition, x)))
```

Figure 7.14 `convince-by-cause-and-evidence` Plan Operator

To illustrate this type of argument, a popular argument claiming academics are devalued in America was represented in FRL (Roberts and Goldstein, 1977). Figure 7.15 illustrates a portion of the knowledge base as well as the text plan and corresponding surface form that TEXPLAN produces when the goal `BELIEVE(H, Devalued(ACADEMICS))` is posted to the system. To achieve this effect, TEXPLAN uses its plan operators to select, structure, and order content from the knowledge base to argue for the proposition.

In Figure 7.15, the claim is first explicated by a number of causes, and then supported by several pieces of evidence, by instantiating the plan operator from Figure 7.14. Instead of simply realizing each

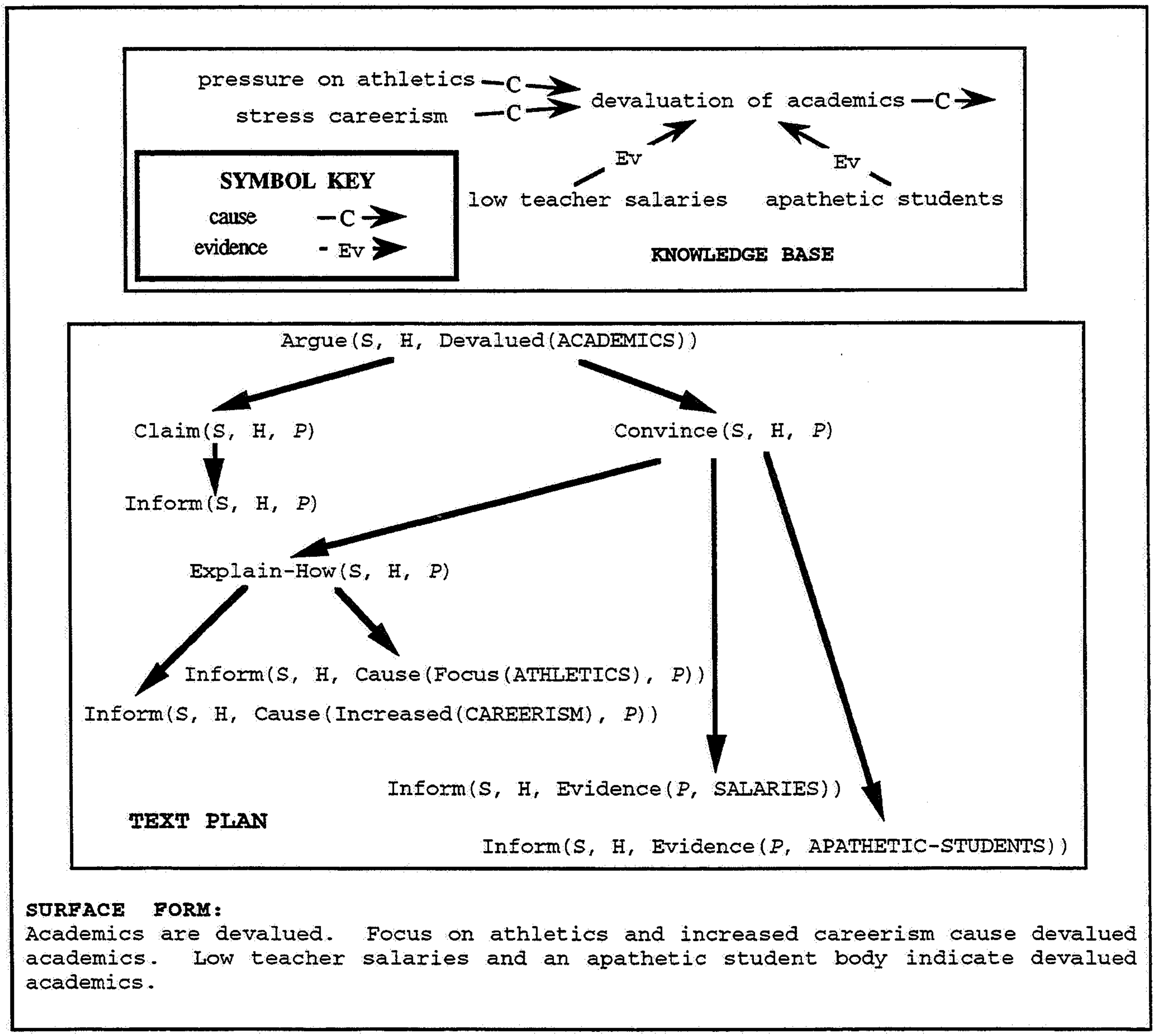



Figure 7.15 Example Argument Knowledge Base, Text Plan, and Surface Form

leaf-node (i.e., surface speech act) in the text plan as a sentence, the surface generator can combine contiguous cause and evidence propositions that have common terms (e.g., in Figure 7.15 the two causes and the two pieces of evidence each form a single utterance.) Evidence predicates were not combined in the NEUROPSYCHOLOGIST output because each evidence rhetorical message had differing certainty percentages.

### 7.3.1 Supporting Tactics for Inductive Argument

Inductive argument, in its narrow sense, moves from particular to general as illustrated by the above plan operators, which show cause and evidence in an attempt to convince a hearer of a proposition. There are other more general techniques that a speaker can use to support a claim that are obviously relevant to, and therefore illustrated for, induction. These include illustration, comparison/contrast and analogy, and parallel those used for entity description in Chapter 4. That is, some communicative acts are multipurpose and can achieve different communicative goals in different contexts, so they can be appear in multiple text types. For example, the communicative act of illustration, used previously to make an abstract description concrete, is here used to convince the hearer of a proposition by exemplifying it (see `convince-by-illustration` plan operator in Figure 7.16). For example, the speaker could illustrate the claimed evidence that the salaries of teachers are low by stating "For example, in Charleston elementary school, the salaries of teachers are below poverty level." As with evidence, examples should be ordered according to their ability to convince, although it is unclear how to do this computationally.

Just as examples can convince the hearer of a proposition, other techniques such as comparison/contrast and analogy can be equally convincing. We can support the above claim about American academics by comparing American and Japanese education to highlight America's low respect for the teaching profession. In contrast, analogy entails comparing the proposition, P, which we are trying to convince the hearer to believe, with a well-known proposition, Q, which has several properties in common with P. By showing that P and Q share properties $\alpha$ and $\beta$, we can claim by analogy that if Q has property $\chi$, then so does P. Figure 7.17 shows what the structure of an analogy plan operator might look like which convinces the hearer of a proposition by providing an analogous proposition the hearer is familiar with. Unlike the other operators in this dissertation, this analogy plan operator has not yet been implemented: it would require a proposition differentia formula similar to but more sophisticated than the entity differentia formula detailed in Appendix A (used for entity analogy in Chapter 4). One reason proposition analogy has not been implemented is the danger of false analogy: while two propositions may share several features with another, other features critical to the comparison may be different. False analogy is one of a larger class of argumentative fallacies detailed in the penultimate section of this chapter.

```
NAME            convince-by-illustration
HEADER          Convince(S, H, proposition)

CONSTRAINTS     Proposition?(proposition) ∧
                ∃x | Illustration(proposition, x)
PRECONDITIONS
  DESIRABLE     ∀x ∈ examples
                   ¬ KNOW(H, Illustration(proposition, x))

EFFECTS         ∀x ∈ examples
                   KNOW(H, Illustration(proposition, x))

DECOMPOSITION   ∀x ∈ examples(proposition)
                   Inform(S, H, Illustration(proposition, x))

WHERE           examples = {e | Illustration(entity, e) ∧
                                KNOW-ABOUT(H, e) ∧
                                KNOW(S, Illustration(entity, e)) }
```

Figure 7.16 `convince-by-illustration` Plan Operator

```
NAME            convince-by-analogy
HEADER          Convince(S, H, proposition)

CONSTRAINTS     Proposition?(proposition)∧
                ∃x | Analogous(proposition, x)

EFFECTS         KNOW(H, Analogous(proposition, analogue))

DECOMPOSITION   Inform(S, H, Analogy(proposition, analogue))

WHERE           analogue = x | Analogous(proposition, x) ∧
                               KNOW-ABOUT(H, x) ∧
                               KNOW(S, Analogous(proposition, x))
```

Figure 7.17 `convince-by-analogy` Plan Operator

This section has formalized several inductive methods of argument including giving evidence and cause. Inductive argument, in its extended sense, can employ more general rhetorical techniques including illustration, comparison/contrast, and analogy. Together with the deductive arguments of the previous section, these serve as a repertoire of communicative acts which can achieve the higher level goal of convincing the hearer to believe a proposition. Other reasoning strategies such as abduction or case-based reasoning should also be able to produce similar effects (although they may have their own particular presentation strategies) and would clearly require handling in a fuller treatment of the forms of argument.

These strategies are all defined to generate texts which affect hearer beliefs. In contrast, the next section examines texts which affect the hearer's knowledge of domain actions and their desire to perform them.

## 7.4 Persuasion and Arguments that Promote Action

An important link between physical and communicative actions concerns getting the hearer to perform some *action*. While different forms of argument such as deduction and induction can be belief or action-oriented, the previous sections have defined deductive and inductive forms narrowly as primarily affecting hearer beliefs; this section will similarly define persuasive techniques in the narrow sense as primarily affecting hearer actions. (Of course in the act of convincing someone to believe a proposition using deductive or inductive techniques you can also persuade them to act. Similarly, in the course of persuading someone to act you can change their beliefs.) The following invitation exemplifies arguments that encourage action:

> Come to my party tonight. It's at 1904 Park Street. We are serving your favorite munchies and we have plenty of wine and beer. Everybody is going to be there. You'll have a great time.

The text tells the reader what to do, enables them to do it, and indicates why they should do it. This common communicative strategy occurs frequently in ordinary texts intended to get people to do things. It consists of *requesting* them to do the act (if necessary), *enabling* them to do it (if they lack the know-how), and finally *persuading* them that it is a useful activity that will produce some desirable benefit (if they are not inclined to do it). In the above example the action, coming to the party, is enabled by providing the address. The action is motivated by the desirable attributes of the party (i.e., tasty munchies and abundant supply of liquor), the innate human desire to belong, and by the desired consequence of coming to it (i.e., having fun).

This general strategy corresponds to the `request-enable-persuade` plan operator shown in Figure 7.18. The operator gets the hearer to do some action by requesting, enabling, and then persuading them to do it. `Enable`, the second communicative act in its decomposition, applies the enablement plan operator used for exposition in Chapter 6 to argument plan operators. This is a particular instance of the more general property of the plan operators presented in this dissertation: compositionality. The plan operator in Figure 7.18 distinguishes among (1) the hearer's knowledge of how to perform the action (i.e., knowledge of the subactions of the action) (KNOW-HOW), (2) the hearer's ability to do it (ABLE), and (3) the hearer's desire to do it (WANT). For example, the hearer may want and know how to get to a party, but they are not able to come because they are sick. If the speaker knows this, then they should not use the plan operator below because its constraints fail. The assumption is that a general user modelling/acquisition component will be able to provide this sort of information.

```
NAME            request-enable-persuade
HEADER          Argue(S, H, Do(H, action))

CONSTRAINTS     [illegible]
PRECONDITIONS
  ESSENTIAL     [illegible]
                WANT(S, Do(H, action)) ∧
  DESIRABLE     [illegible]
                ¬ KNOW-HOW(H, action)  ∧
  DESIRABLE     [illegible]

EFFECTS         KNOW(H, WANT(S, Do(H, action))) ∧
                KNOW-HOW(H, action) ∧
                WANT(H, Do(H, action)) ∧
DECOMPOSITION   [illegible] Do(H, action))
---------------------------------------------------
DECOMPOSITION   Request(S, H, Do(H, action))
                En[illegible]
                Persuade(S, H, Do(H, action))
```

Figure 7.18 Top-Level, Uninstantiated `request-enable-persuade` Plan Operator

The order and constituents of a communication that gets an individual to act, such as that in Figure 7.18, can be very different indeed depending upon the conversants involved, their knowledge, beliefs, capabilities, desires, and so on. Thus to successfully get a hearer to do things, a speaker needs to reason about his or her model of the hearer in order to produce an effective text. For example, in an autocratic organization, a request (perhaps in the linguistic form of a command) is sufficient. In other contexts no request need be made because the hearer(s) may share the desired goal, as in the case of the mobilization of the Red Cross for earthquake or other catastrophic assistance. Similarly, if the hearer wants to do some action, is able to do it, and knows how to do it, then the speaker can simply ask them to do it. Because the hearer is able to do it, the speaker need not enable them. And because the hearer wants or desires the outcome of the action, the speaker need not persuade them to do it. This situation corresponds to the `request` plan operator in Figure 7.19 which argues that the hearer perform an action by simply asking them to do it. A variation on plan operator in Figure 7.19 could model delegation, whereby the speaker may know the hearer is not willing to do or does know how to perform some task, but the speaker simply asks them because it is expected that they figure out how to do it. As with the autocratic example above, his would require a model of the interpersonal relations of the speaker and hearer (c.f, Hovy, 1987).

In addition to a request for action, enablement may be necessary if the audience does not know how to perform the task. The following text from the NYS Department of Motor Vehicles *Driver's Manual* (p. 9) informs the reader of the prerequisites for obtaining a license:

```
To obtain your driver's license you must know the rules of the road and
how to drive a car or other vehicle in traffic.
```

The writer indicates that being knowledgeable of both road regulations and vehicle operation are necessary preconditions for obtaining a license. In some situations, however, the reader may be physically or mentally unable to perform some action, in which case the writer should seek alternative solutions, eventually perhaps consoling the reader if all else fails. On the other hand, if the user is able but not willing to perform the intended action, then a writer must convince them to do it, perhaps by outlining the benefit(s) of the action. Consider this excerpt from the *Driver's Manual*:

```
The ability to drive a car, truck or motorcycle widens your horizons.
It helps you do your job, visit friends and relatives and enjoy your
leisure time.
```

Of course it could be that the hearer already wants to do something but does not know how to do it. This situation corresponds to the `request-enable` plan operator shown in Figure 7.20 which requests and then enables the hearer to perform some action.

```
NAME            request-enable
HEADER          Argue(S, H, Do(H, action))

CONSTRAINTS     Action?(action) ∧ ABLE(H, action)
PRECONDITIONS
  ESSENTIAL     WANT(S, Do(H, action)) ∧ WANT(H, action)

  DESIRABLE     ¬ KNOW(H, WANT(S, Do(H, action))) ∧
                ¬ KNOW-HOW(H, action)

EFFECTS         KNOW(H, WANT(S, Do(H, action))) ∧
                KNOW-HOW(H, action) ∧
                Do(H, action)

DECOMPOSITION   Request(S, H, Do(H, action))
                Enable(S, H, Do(H, action))
```

Figure 7.20 `request-enable` Plan Operator

The above plan operators define the top-level communicative actions which TEXPLAN uses to get the hearer to perform some action. On the basis of an assumed model of the user's knowledge, abilities, and desires, TEXPLAN is able to select from these different strategies by examining their various constraints and preconditions in order to produce a text tailored to that user. While requesting and enablement have been defined in previous chapters (4 and 6, respectively), the communicative act of persuade is formalized in the next subsection. A final subsection illustrates arguments that induce action in the domain of a mission planning system.

```
NAME            persuade-by-motivation
HEADER          Persuade(S, H, Do(H, action))

CONSTRAINTS     Act(action) ∧ ∃x |Motivation(x, action)
PRECONDITIONS
  ESSENTIAL     ¬WANT(H, Do(H, action))
  DESIRABLE     ∃x |Motivation(x, action) ∧
                    ¬KNOW(H, Motivation(x, action))

EFFECTS         WANT(H, Do(H, action)) ∧
                ∀x |Motivation(x, action)
                  KNOW(H, Motivation(x, action))

DECOMPOSITION   ∀x |Motivation(x, action)
                  Inform(S, H, Motivation(x, action))
                  optional(∀y |Cause(y, x)
                           Inform(S, H, Cause(y, x)))

WHERE           events-or-states = {x | Motivation(x, action) }
```

Figure 7.21 `persuade-by-motivation` Plan Operator

### 7.4.1 Persuasive Techniques

When a hearer does not want to perform the action, a speaker must persuade them to act. There are a variety of ways to persuade the hearer including indicating (1) the motivation for the action, (2) how the action can enable some event, (3) how it can cause a desirable outcome, or (4) how the action is a part of some overall purpose or higher level goal. For example, the plan operator named `persuade-by-motivation` in Figure 7.21 persuades the hearer to act by simply indicating the motivation for the action, where the `Motivation` predicate is the same as that used in Chapter 5 for narrative plan operators. An option in the decomposition of the plan operator in Figure 7.21 indicates how the motivating event or state came about, i.e., it explains the motivating circumstances.

Another persuasive technique involves telling the consequences of the action which are beneficial to the hearer. An action can either cause a positive result (e.g., approval, commendation, praise) or avoid a negative one (avoid blame, disaster, or loss of self esteem). Advertisement often uses this technique to induce customers to purchase products by appealing to the emotional benefits (actual or anticipated) of possession. This technique is formalized in Figure 7.22 as the `persuade-by`-desirable-consequences plan operator which gets the hearer to want to do something by telling them all the desirable events or states that the action will cause. An extension of this plan operator could warn the hearer of all the undesirable events or states that would result from their inaction.

```
NAME            persuade-by-desirable-consequences
HEADER          Persuade(S, H, Do(H, action))

CONSTRAINTS     Act(action) ∧ ∃x | Cause(action, x)
PRECONDITIONS
  ESSENTIAL     ¬ WANT(H, Do(H, action))
  DESIRABLE     nil

EFFECTS         WANT(H, Do(H, action)) ∧
                ∀x ∈ desirable-events-or-states
                   KNOW(H, Cause(action, x))

DECOMPOSITION   ∀x ∈ desirable-events-or-states
                     Inform(S, H, Cause(action, x))

WHERE           desirable-events-or-states =
                     {x | Cause(action, x) ∧ WANT(H, x) }
```

Figure 7.22 persuade-by-desirable-consequences Plan Operator

Some actions may not cause a desirable state or event but may enable some other desirable action (that the hearer or someone else may want to perform). For example, in the NYS driving example, obtaining a license is a precondition of driving a car, which enables you to visit friends, go shopping, etc. This communicative act is captured in the persuade-by-enablement plan operator shown in Figure 7.23. Just as the persuade-by-motivation plan operator of Figure 7.21 could be extended to indicate causes of an action, an extension of the persuade-by-enablement plan operator could warn the hearer of all the undesirable events or states that would be enabled by their inaction.

```
NAME            persuade-by-enablement
HEADER          Persuade(S, H, Do(H, action))

CONSTRAINTS     Act(action)
PRECONDITIONS
  ESSENTIAL     ¬ WANT(H, Do(H, action))

EFFECTS         WANT(H, Do(H, action)) ∧
                ∀x ∈ desirable-events-or-states
                   KNOW(H, Enablement(action, x)

DECOMPOSITION   ∀x ∈ desirable-events-or-states
                   Inform(S, H, Enablement(action, x))

WHERE           desirable-events-or-states =
                    {x | Enablement(action, x) ∧ WANT(H, x) }
```

Figure 7.23 persuade-by-enablement Plan Operator

```
NAME            persuade-by-purpose-and-plan
HEADER          Persuade(S, H, Do(H, action))

CONSTRAINTS     Act(action)
PRECONDITIONS
  ESSENTIAL     ¬ WANT(H, Do(H, action))

EFFECTS         WANT(H, Do(H, action)) ∧
                KNOW(H, Purpose(action, goal))

DECOMPOSITION   Inform(S, H, Purpose(action, goal))
                Inform(S, H, Constituent(plan, action))

WHERE           goal = g | Purpose(action, g) ∧ WANT(H, g)
                plan = p | Constituent(p, action) ∧ WANT(H, Do(p))
```

Figure 7.24 persuade-by-purpose-and-plan Plan Operator

One final form of persuasion, persuade-by-purpose-and-plan, shown in Figure 7.24, gets the hearer to perform some action by indicating its purpose or goal(s) and how it is part of some more general plan(s) that the hearer wants to achieve. For example, one subgoal of shopping is writing a check, an action which has the effect or purpose of increasing your liquid assets. These operators give a range of persuasive possibilities. Tests with them are illustrated in the next section.

### 7.4.2 Persuasion in the Knowledge Replanning System

All the persuasive devices described—motivation, enablement, cause, and purpose—allow TEXPLAN to persuade the hearer to act. To illustrate persuasion requires an advisory system which makes recommendations, or a cooperative problem solver. The plan operators were in fact tested with the cooperative Knowledge based Replanning System, KRS (Dawson et al., 1987), used for mission planning (but emphasizing resource allocation and scheduling rather than planning a sequence of steps to accomplish some goal). Unlike previous examples in this dissertation, where the underlying application was based on some type of semantic network formalism, KRS is implemented in an extension of FRL (Roberts and Goldstein, 1977), a hybrid of rules and hierarchical frames. (In producing descriptions from KRS, TEXPLAN accesses only frame knowledge structures). KRS employs meta-planning (Wilensky, 1983) whereby high-level problem solving strategies govern lower-level planning activities. Furthermore, KRS is a mixed-initiative planner which cooperates with the user to produce an Air Tasking Order, a package of air missions (e.g., offensive counter air, air refueling, air escort, surface-to-air-missile suppression) that achieve some desired goal (e.g., destroy an enemy target). Because of this multi-agent problem solving, the system and user can make choices which result in an ill-formed mission plan. If directed by the user, KRS then replans the mission plan using dependency-directed backtracking (e.g., making changes in the plan by reasoning about temporal and spatial relationships). KRS initially attempts to retract system-supplied choices. As a last resort, KRS suggests to the user that they remove user-supplied choices to recover from the ill-formed plan. In this case the system tries to justify its recommendation on the basis of some underling rule governing legal plans.

For example, assume the user has interacted with the system to produce the mission shown in Figure 7.25 (simplified for readability). The frame, OCA1002, is an offensive counter air mission, an instance of (AIO) the class offensive counter air (OCA), with attributes such as the type and number of aircraft, the home airbase, and the target. Each attribute has actual and possible values as well as STATUS slot which indicates who supplied the value (e.g., user, planner, meta-planner (called the strategist)). Frames also record interactional information, for example in Figure 7.25 the HISTORY slot records that the user just selected a target and the WINDOW slot indicates where the mission plan is visually displayed. KRS represents domain-dependent relations among slots so that values for some of the slots can be automatically calculated by daemons in reaction to user input (e.g., when the UNIT and ACNUMBER slot of a mission are filled in, the CALL-SIGN slot can be automatically generated).

During planning the system monitors and detects ill-formed mission plans by running rule-based diagnostic tests on the mission plan. For example, in Figure 7.25 the offensive counter air mission has an incompatible aircraft and target. KRS signals the constraint violation by highlighting the conflicting slots (e.g., AIRCRAFT and TARGET) of the mission frame which is represented visually in a mission window to the user. Before TEXPLAN was interfaced to KRS, KRS would then simply state the rule-based constraint which detected the error in the mission plan, and then list some of the supporting knowledge (see Figure 7.26).

```
(OCA1002
   (AIO         (VALUE      (OCA)))
   (AIRCRAFT    (POSSIBLE   ((F-4C F-4D F-4E F-4G F-111E F-111F)))
                (VALUE      (F-111E))
                (STATUS     (USER)))
   (TARGET      (VALUE      (BE30703))
                (STATUS     (USER)))
   (ACNUMBER    (POSSIBLE   ((1 2 ...  25)))
                (VALUE      (3))
                (STATUS     (USER))))
   (AIRBASE     (POSSIBLE   ((ALCONBURY))))
   (ORDNANCE    (POSSIBLE   ((A1 A2 ... A14))))
   (HISTORY     (VALUE      (#<EVENT INSERT TARGET BE30703 USER>)))
   (DISPLAY     (VALUE      (#<MISSION-WINDOW 1 1142344 deexposed>)))
```

Figure 7.25 Simplified Mission Plan in FRL

```
The choice for AIRCRAFT is in question because:

BY TARGET-AIRCRAFT-1:3 THERE IS A SEVERE CONFLICT BETWEEN TARGET  AND
AIRCRAFT FOR OCA1002

            1. THE TARGET OF OCA1002 IS BE30703
            2. BE30703 RADIATES4
            3. THE AIRCRAFT OF OCA1002 IS F-111E
            4. F-111E IS NOT A F-4G
```

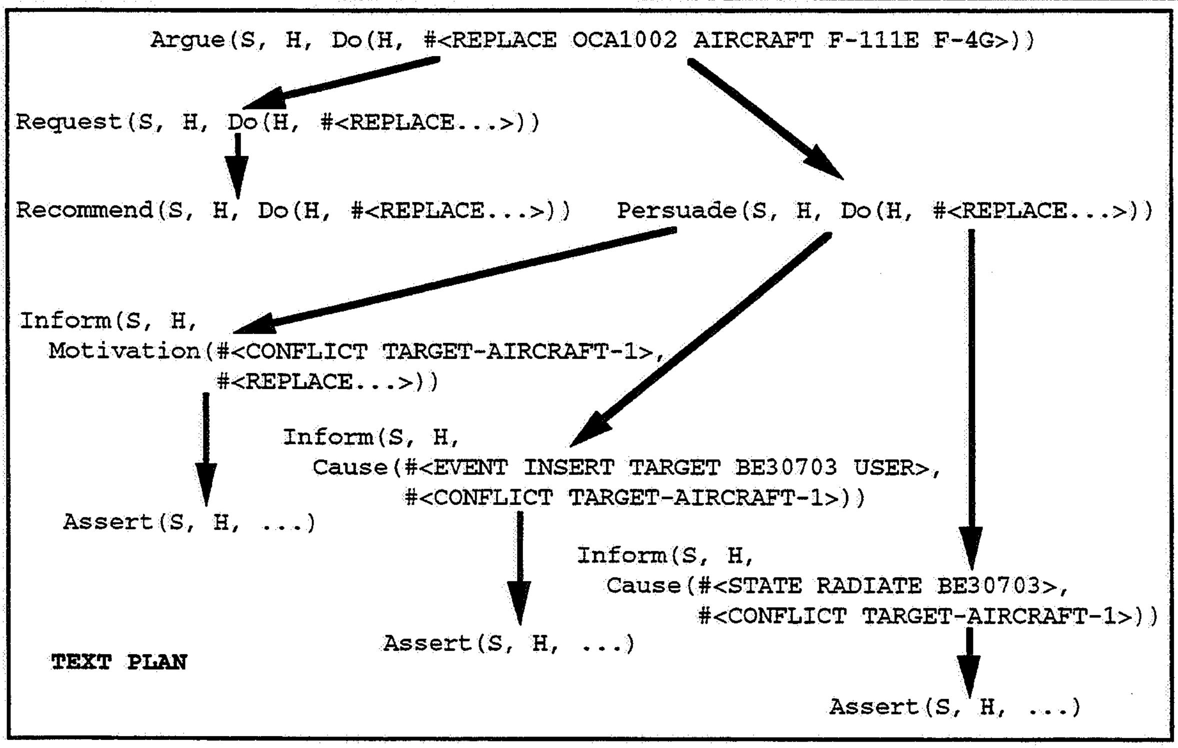
Figure 7.26 Current Explanation of Rule Violation

The first two sentences of the explanation in Figure 7.26 were produced using simple templates (canned text plus variables for the mission, rule name, and conflicting slots). The list 1-4 is simply a sequence of information supporting the constraint violation although there is no indication as to how these relate to each other or to the rule. Because the relationships among entities are implicit, this text lacks cohesion. More important, it is not clear what the system wants the user to do and why they should do it.

[3] TARGET-AIRCRAFT-1 is the name of the rule that detected the conflict. OCA1002 reads "Offensive Counter Air Mission 1002" and BE30703 reads "Battle Element number 30703".

[4] The fact that BE30703 is radiating indicates that it is an operational radar. KRS expects domain users (i.e., Air Force mission planners) to know that only anti-radar F-4g ("Wild Weasel") aircraft fly against these targets.

Rather than achieving organization from some model of naturally occurring discourse (such as TEXPLAN's plan operators), the presentation in Figure 7.26 is isomorphic to the underlying inference chain. In contrast, TEXPLAN was interfaced to KRS by relating rhetorical predicates (e.g., cause, motivation, attribution) to the underlying semantic relations of the domain embodied both in rules justifying constraint violations and in frames representing the mission plan and other domain entities (e.g., aircraft and target frames). Unlike the template and translate-the-code approach used to produce the text in Figure 7.26, KRS posts the goal `WANT(H, Do(H, #<REPLACE OCA1002 AIRCRAFT F-111E F-4G>))` to TEXPLAN, which then reasons abstractly about epistemology and rhetoric (using, for example, the plan operator of Figure 7.21) to generate the text plan and corresponding surface form shown in Figure 7.27. The output is improved not only by composing the text using communicative acts, but also by linguistic




**SURFACE FORM:**
You should replace F-111e aircraft with F-4g aircraft in Offensive Counter Air Mission 1002. A conflict between the aircraft and the target in Offensive Counter Air Mission 1002 motivates replacing F-111E aircraft with F-4g aircraft. You inserted Ludwigslusts-Alpha in the target slot and Ludwigslusts-Alpha was radiating which caused a conflict between the aircraft and the target in Offensive Counter Air Mission 1002.

Figure 7.27 TEXPLAN Argument to get user to act -- Motivated by Rule Violation devices, such as using a dictionary to produce lexemes such as "Offensive Counter Air Mission 1002" and "Ludwigslusts-Alpha" instead of `OCA1002` and `BE30703`. Unlike the previous examples in this thesis in which the discourse goal corresponds to an explicit user query, here the discourse goal that drives the text is produced by the underlying system (KRS).

In addition to showing cause and motivation, another way to persuade a hearer to do an action is to show how that action supports some more general plan or goal (see the plan operator in Figure 7.24). Since KRS employs meta-planning, it can justify its actions by referring to the higher level strategy it is employing. For example, Figure 7.28 shows a number of strategy plans (e.g., plan an air tasking order, replan an air tasking order, replan an attack mission, and so on) which govern lower-level planning activities (e.g., prescan a package of missions, plan a package of missions, plan an individual mission, and so on). Associated with each meta-plan shown in Figure 7.28 are several types of information including its name, type, purpose, subgoals, relations among subgoals (e.g., enablement, sequence, etc.), planning history, associated entities (e.g., the name of the mission being replanned), and failure handlers. Therefore when the actions encoded by the plans are executed, the meta-planner knows why particular actions occur when they do. For example if the user is not persuaded that scanning a plan is a useful activity and they may ask "Why is scanning the plan necessary?" (simulated by posting the goal `WANT(H, DO(S, #<PRESCAN-ATO>)))` To achieve this goal, TEXPLAN can use the `persuade-by-purpose-and-plan` operator of Figure 7.24 to examine the meta-plan structure and produce the response shown in Figure 7.29.

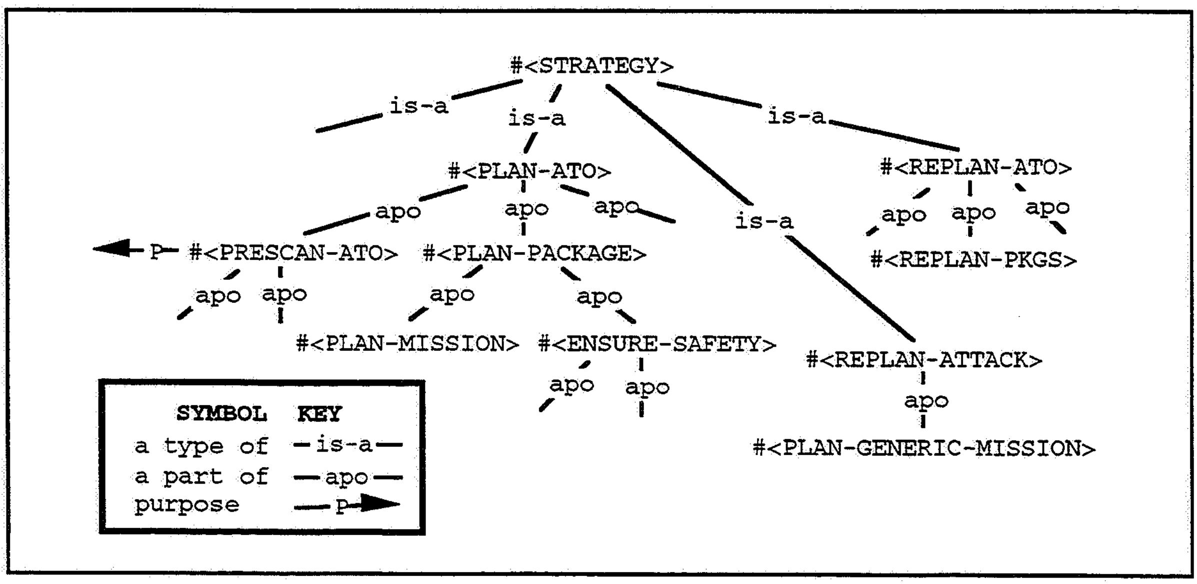
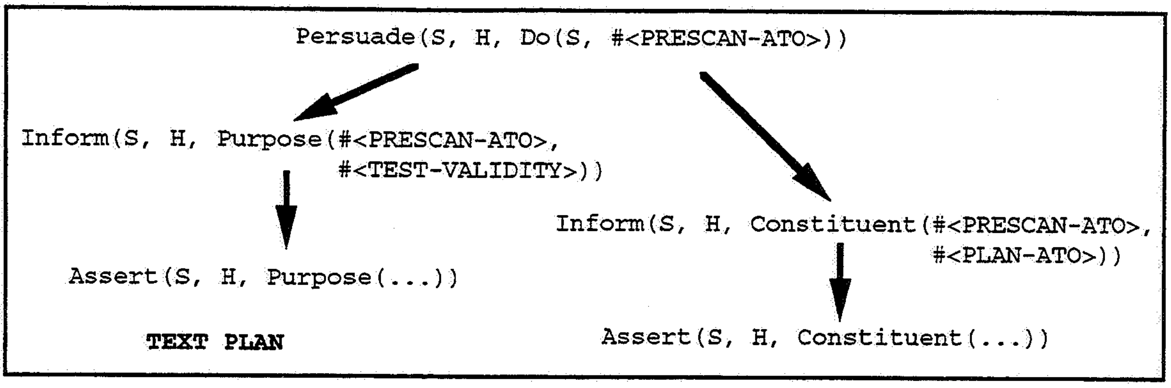



Figure 7.28 Structure of Plans and Meta-Plans in KRS




**SURFACE FORM:**
The purpose of prescanning the Air Tasking Order is to test the validity of the Air Tasking Order. Prescanning the Air Tasking Order is part of planning an Air Tasking Order.

Figure 7.29 TEXPLAN `Persuasion` of a Domain Action

Having illustrated TEXPLAN's resources for persuasive argument, we now note the key differences between these plan operators and Moore's (1989) "recommend-enable-motivate" strategy detailed at the end of Chapter 3. First, the "recommend-enable-motivate" strategy is but one (preordered) pattern (with optional components) in a family of operators that can be used to get the hearer to do something. In contrast, TEXPLAN has a number of plan operators at this level. Second, Moore's plan operators do not distinguish rhetorical acts (e.g., enable, persuade) from illocutionary acts (e.g., request) from surface speech acts (e.g., command, recommend). Furthermore, Moore's system can persuade by three techniques: motivating (by telling the purpose and/or means of an action), showing how an action is a step (i.e., subgoal) of some higher-level goal (elaborate-refinement-path), and giving evidence. Some of the techniques in her system are domain specific (e.g., motivate-replace-act, where "replace" is a domain (i.e., Program Enhancement Advisor) specific action), and others are architecture/knowledge representation specific (e.g., elaborate-refinement-path is a technique based on the Explainable Expert System's architecture (Swartout, 1983; Neches et al., 1985)). In contrast, TEXPLAN can persuade by showing motivation, enablement, cause, and purpose. Moreover, unlike Moore's system, individual plan operators can (and often do) have multiple effects (indicated by a list of effects in the effect slot of plan operators). Finally, Moore's system does not distinguish as does TEXPLAN between convince and persuade (i.e., convincing a hearer to believe a proposition versus persuading them to perform an action).

Despite the flexibility and range of TEXPLAN's plan operators, one unresolved issue concerns the multi-function nature of text types and their interaction. The plan operators used by TEXPLAN allow for multiple effects as well as the composition of communicative acts, however, the flexible and multi-function nature of naturally occurring prose eludes current text planners including TEXPLAN. For example, the advertisement below compels the reader to action using a variety of techniques including description, comparison, and persuasion.

> Buy Pontiac. We build excitement. The new Pontiacs have power brakes, power steering, AM/FM stereos, and anti-lock brakes. And if you buy now, you will save $500. An independent study shows that Pontiacs are better than Chevrolet. See your Pontiac dealer today!

In this example, the initial request for action (i.e., purchase) is supported by indicating the desirable attributes of the product, the desirable consequences of the purchase, comparing the action with alternative courses of action/competing products, and finally imploring the hearer to act again. While some of these techniques may be implemented as plan operators in a straightforward manner (e.g., describe desirable attributes), the interaction of various text types remains a complex issue. For example, how is it that some texts can persuade by description, narration or exposition, and entertain by persuasion? This issue is an exciting area for future research.

## 7.5 Fallacious Arguments

There are well known rhetorical techniques, termed *fallacies* by logicians, which may convince the hearer of a proposition or persuade them to act, however this end will be achieved by means based not on sound logic but rather on weaknesses of human nature or attention. These fallacies are identified here because it is useful to distinguish them from more legitimate argument techniques and because they may prove useful to the interpretation and detection of ill-formed argument. These techniques are quite effective in debating although they actually obfuscate the truth rather than uncover it, hence the morality of their application is questionable.

The three most common categories of fallacious arguments include those based on *false inferences*, those based on *incomplete knowledge*, and those which are *linguistically flawed.* One type of inferential error is *overgeneralization*, for example in a syllogism assuming a major premise of "All green apples are sour." Others include *petitio principii* (begging the question), *post hoc ergo propter hoc* (circumstantial argument), *false analogy*, and *contradiction.* In addition to these false inferences, arguments are often flawed by incomplete knowledge. This includes fallacies such as *argumentum ad ignorantiam* (meaning argument from ignorance, that is assuming something is false because there is no compelling evidence supporting the proposition) and *ignoratio elenchi* (ignorance of refutation). In addition to these inferential and epistemological fallacies, an argument can be flawed linguistically. This can occur a number of ways including the use of ambiguous terms (*equivocation*), ambiguous English syntax (*amphiboly*), or ambiguous referents.

In addition to logical, epistemological, and linguistic fallacies, there are a number of other fallacious arguments. These include *argumentum ad hominem* (argument against the person), *argumentum ad* *populum* (appeal to the people), *argumentum ad baculum* (argument toward the stick or appeal to force or threat), and *argumentum ad verecundiam* (argument toward reverence or appeal to an authority, theory, or maxim) as well as others. The above techniques can be quite effective, especially if the audience is young, impressionable, or uninformed.

In the context of simulating human behavior, several implementations have investigated some of the above types of persuasive techniques. As described in Chapter 5, characters in Meehan's (1976) TALE-SPIN simulation could persuade others to do actions by, for example, threatening them. More recently, Sycara's (1989) PERSUADER program simulates labor negotiations in which three agents (company, union, and mediator) can select from nine persuasive techniques (e.g., appeal to "status quo", appeal to "authority", threat) to effect other agent's plans and goals. The agents in PERSUADER engage in argument with the purpose of influencing another agent's belief of how important or feasible a particular goal is to that agent's overall goal (e.g., the top-level goal of a company is profit). However the selection of persuasive techniques are based on simple if-then heuristics and no argument structure or language are produced. Moreover, PERSUADER does not distinguish between convincing an addressee to believe a proposition and persuading them to act. While these coercive techniques may emulate human behavior, their use by an advisory system is probably not appropriate except in special cases (e.g., persuading someone to take their prescribed medicine).

## 7.6 Conclusion

Argument, perhaps the most important form of text, allows us to change others beliefs and influence their actions. Like the text types of description, narration, and exposition examined in the previous chapters, this chapter characterizes argument as a plan-based communicative activity. The chapter deals with two classes of argument that are used principally to change beliefs: deduction and induction. The focus here is on the presentation of arguments (i.e., their form) rather than their representation (i.e., the underlying inference or reasoning strategies), so no claims are made concerning the latter. Furthermore, no claims are made concerning the representation of intentions and beliefs. Following the methodology of previous chapters, communicative acts that appear to constitute deductive and inductive arguments are identified in naturally occurring text and then expressed as plan operators which encode the necessary preconditions and constraints of the actions as well as their effects and decomposition into other subacts. The chapter then formalizes a number of communicative acts that can be used to persuade the addressee to perform some physical action, making the important link between physical and linguistic action. The closing section of this chapter identifies a number of argumentative fallacies. Except in special cases, I argue that these should not be employed in communicative interfaces to advisory systems because they are founded on human limitations rather than reason.

Because of the differing content and force of each of the types of argument (i.e., deduction versus induction versus persuasion), the plan operators are illustrated with implemented examples from several domains. Some plan operators were more seriously tested using real applications (e.g., NEUROPSYCHOLOGIST for induction; KRS for persuasion), whereas others were illustrated by developing a knowledge base with the necessary underlying semantic relations (e.g., the Socratic syllogism; the academic devaluation claim supported by cause and evidence) and thus are merely indicative of how the plan operators would function in a real application.

The plan operators in this chapter are compositional, in many cases calling upon previously defined communicative acts to accomplish their goals. For example, the top-level `argue-for-a-proposition` plan operator in Figure 7.1 claims some proposition, next explains it (using the expository plan operators of the previous chapter which may invoke the descriptive operators of Chapter 4), and finally attempts to convince the hearer of its validity. One method of convincing, `convince-by-cause-and-evidence` shown in Figure 7.14, calls upon the `Explain-How` communicative act from Chapter 5 as illustrated in the academic devaluation example. The current plan operators define where and how in a text communicative acts can serve various functions. For example, illustration can be used both to make an abstract concept concrete or to support a proposition (cf. Section 7.3.1). What remains to be investigated is how content and context modifies the effect of different text types (e.g., deduction can both change beliefs and move to action depending upon context). This seems analogous to the case where the force of illocutionary speech acts can be altered by syntactic form or intonation.

This chapter, and the previous three, have detailed how TEXPLAN reasons about content, form, and effect to produce hierarchical text plans which characterize four types of text: description, narration, exposition, and argument. The next chapter considers how these text plans are linearized and linguistically realized as cohesive English text.

# Chapter 8

# LINGUISTIC REALIZATION

"When I use a word" Humpty Dumpty said, in a rather scornful tone,
"it means just what I choose it to mean -- neither more nor less."
"The question is," said Alice, "whether you *can* make words different things."
"The question is," said Humpty Dumpty, "which is to be master -- that's all."

*Through the Looking Glass*

## 8.1 Introduction

As detailed in Figure 2.0 at the beginning of Chapter 2, language generation can be broadly divided into strategic (i.e., text planning) and tactical (i.e., linguistic realization) stages. The last four chapters—description, narration, exposition and argument—have described how TEXPLAN produces hierarchical communicative plans which characterize the structure, content, and effect of a text. In contrast, this chapter shows how the hierarchical communicative plan which results from text planning is executed, i.e., linguistically realized as English. In particular, this chapter details how the text plan's leaf nodes—surface speech acts and their propositional content—are linguistically realized by selecting appropriate words and phrases, grammatical structures and intersentential connectives to produce well-formed and cohesive output. This tactical realization component uses the same linguistic apparatus for all the different types of text that TEXPLAN can produce.

Figure 8.1 provides a serial view of the various stages and sources of knowledge used for linguistic realization in TEXPLAN. In Figure 8.1, a hierarchical communicative plan is completely constructed before it is linearized by a simple depth-first search which outputs a list of *messages*, where each message consists of a leaf-node locutionary act and its associated rhetorical proposition. Following McKeown (1982), each rhetorical proposition is a rhetorical predicate (e.g., attribution, constituency) instantiated with information from the application knowledge base. Some of these messages are then grouped (e.g., multiple assertions of evidence for a common claim can be combined into a single evidential assertion). Each item on the resulting list is then realized as English (or also as Italian in a few test cases) using information about focus (discourse, temporal, and spatial), semantics, grammatical relations, syntax,

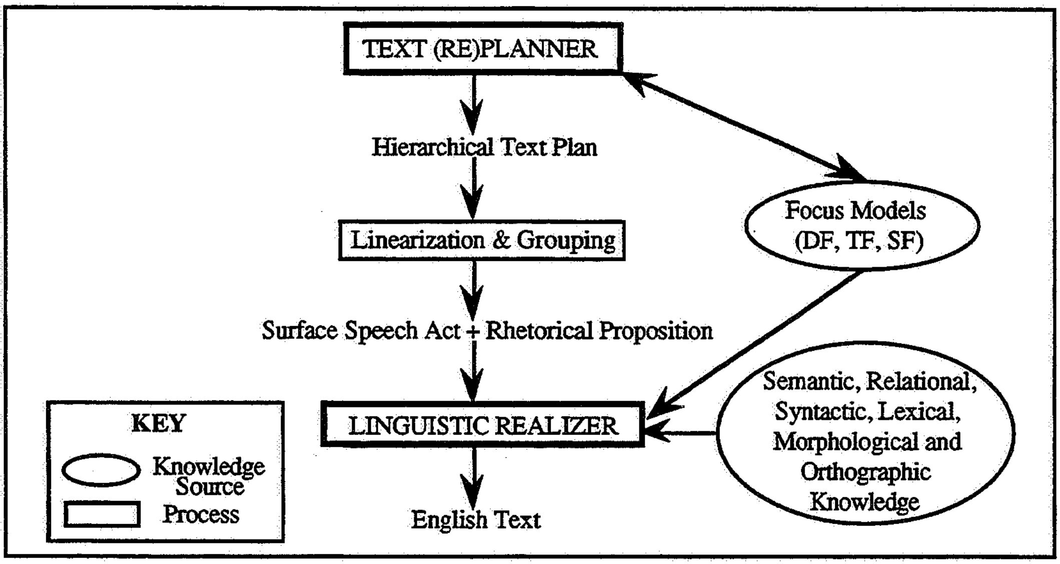



Figure 8.1 Linguistic Realization Component

lexemes, morphology, and orthography. This chapter details all of these knowledge sources in TEXPLAN.

In fact, TEXPLAN can operate in two modes: (1) *serially*, with text planning followed by linguistic realization as indicated in Figure 8.1 or (2) with planning and realization *interleaved*. In the latter case, when the hierarchical text planner encounters a surface speech act (with its associated propositional content), it immediately calls the realization component. If linguistic realization fails, the text planner attempts alternative communicative actions, ultimately failing when it has exhausted all options. Since the generator can utter information before a complete plan is produced in this interleaved mode, it is possible to begin to linguistically realize utterances as part of a partial plan which later fails after further processing. In this case the planner backs up to the most recent decision and begins replanning. Interleaved planning and realization is analogous to MUMBLE's incremental realization of sentences, although MUMBLE's indelibility does not allow for backtracking. Both serial and interleaved text planning and linguistic realization were investigated in order to examine the properties of the two forms of processing.

From a practical standpoint, interleaved planning and realization is perceptually quicker than serial processing because the user can read output before the entire text is planned. Also, in interleaved mode, for any given subgoal TEXPLAN will attempt alternative strategies if linguistic realization fails for a particular utterance (e.g., there are no lexical resources appropriate for the given propositional content). Finally, while not yet investigated computationally in TEXPLAN, with interleaved processing a planner can obtain feedback before it has completely planned a text by monitoring user reactions after each utterance is produced, and it can use this feedback to decide whether to continue, modify (i.e., repair), or abandon (i.e., replan) the current plan.

Significant linguistic realization components have been developed using systemic grammar (Matthiessen, 1981; Mann and Matthiessen, 1983; Fawcett, 1988) and tree-adjoining grammar (McDonald and Pustejovsky, 1985bc). This dissertation does not claim to supersede these efforts, but rather suggests an alternative, hierarchical model of linguistic realization from an abstract level of intentions down to morphology. Given a surface speech act and its propositional content, TEXPLAN's linguistic realization component maps this information onto English via three distinct levels of linguistic representation: a verb-case semantics, grammatical relations (e.g., subject, object), and a feature-enhanced phrase structure grammar. Final surface form is produced by morphology and orthography (i.e., layout) algorithms. The next section outlines each of these levels of linguistic representation and the remainder of the chapter details each in turn.

## 8.2 Linguistic Realization Framework

Figure 8.2 shows the levels of representation in TEXPLAN's linguistic realization component. Recall from Figure 8.1 that the input *message* to the linguistic realization component is a surface speech act (e.g., assert, ask, command, recommend) with its corresponding rhetorical proposition (e.g., logical-definition, constituency, or evidence). This message originates from the strategic text planner as described in the next section. Figure 8.3 illustrates the levels of representation in the linguistic realizer with a particular example from the NEUROPSYCHOLOGIST application: an assert surface speech act with the accompanying logical-definition rhetorical proposition for the domain entity, #<left-hemisphere>. (To constrain the size of the figures, the levels of morphology (i.e., word forms) and orthography (i.e., layout, including capitalization and punctuation) which lie between syntax and surface form are omitted.)

As Figure 8.2 shows, the input message (i.e., surface speech act and rhetorical proposition) is transformed through three levels—semantics, grammatical relations, and syntax—before a morphological and orthographic component (not illustrated) produce the final surface form. To begin with, when the linguistic realization component receives an input message, its attentional model first extracts entities and events from the rhetorical proposition to update the global registers that record the current and past discourse, temporal, and spatial foci. These registers are later used to guide surface choices. Next the semantic interpreter maps entities in the message onto semantic roles (stage 1) (Fillmore, 1968, 1977) using verb case frames associated with each type of rhetorical predicate (e.g., logical-definition, constituency, evidence). These semantic roles are then mapped onto grammatical relations (stage 2) where syntactic experts use discourse focus, semantic and syntactic knowledge to produce grammatical constituents such as subject, direct-object, and verb.

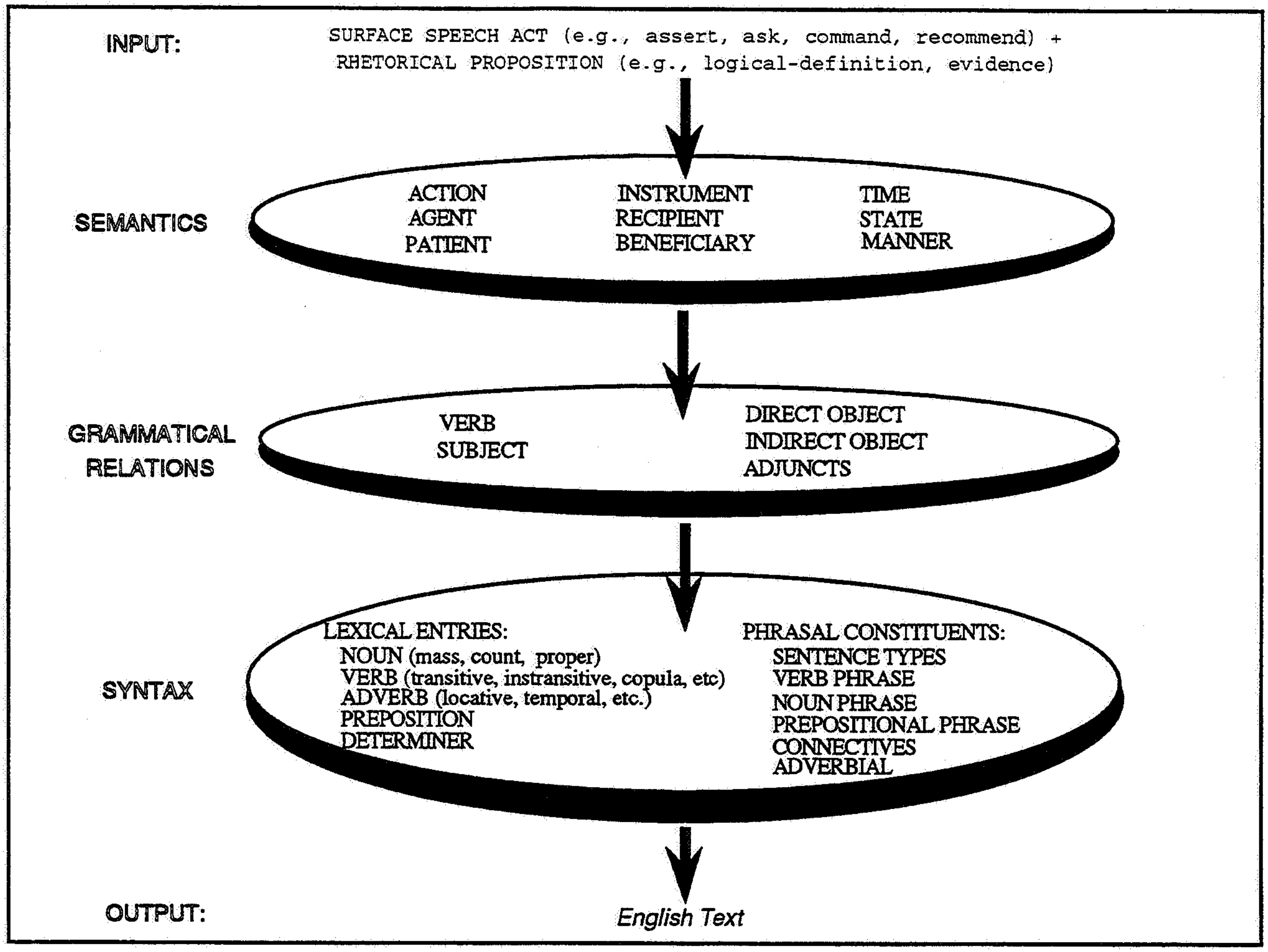



Figure 8.2 Linguistic Realization Framework

Syntactic and focus information guide the selection of voice (active, passive), which determines the ordering of constituents. Because syntactic information constrains decisions at this level, domain entities in the message are translated to lexical entries using the dictionary system which is detailed in Section 8.7.1. The rhetorical role the message plays in the overall discourse (e.g., cause, illustration, conclusion) may suggest particular clue words (e.g., "because", "for example", "therefore") which enhance low-level connectivity. Finally, a syntax tree (stage 3) is generated using a feature-enhanced phrase structure grammar (motivated by GPSG (Gazdar, 1982)), and surface form is provided by morphological and orthographic routines. While Figure 8.2 presents processing as essentially serial and modular, in many instances lower levels of processing (e.g., anaphora selection) must access higher-level information (e.g., discourse focus information) which is retained in global registers.

Figure 8.3 illustrates these transformations delivering the utterance "The left-hemisphere is a region for feature-recognition located in the brain." This is part of a multisentence description generated by TEXPLAN for NEUROPSYCHOLOGIST (Maybury and Weiss, 1987) in response to the query "What is a left-hemisphere?", simulated by posting the discourse goal, KNOW(H, #<left-hemisphere>). The realization process is initiated by the text planner which passes an ASSERT surface speech act along with the

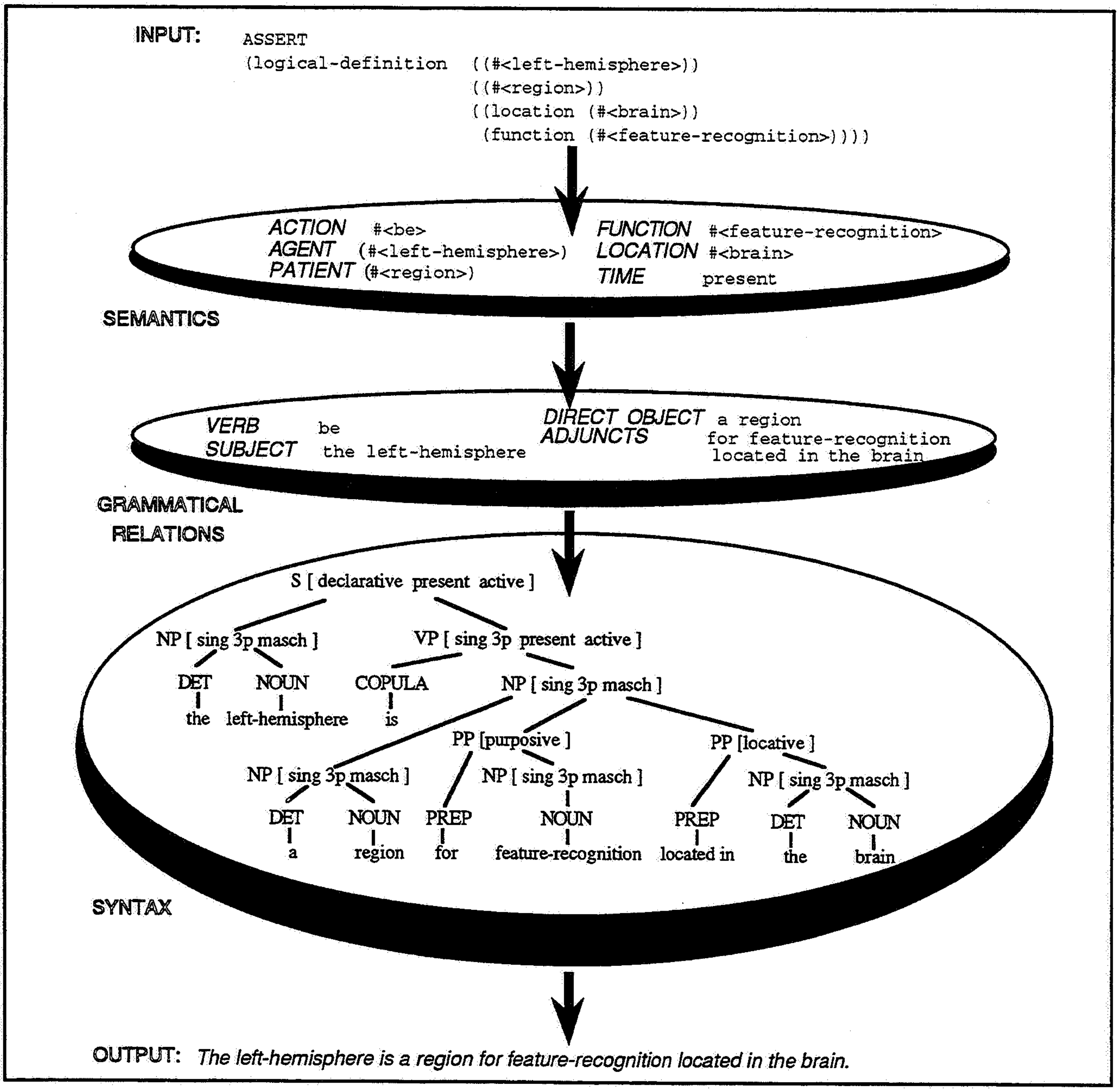



Figure 8.3 An Example of Linguistic Realization

propositional content for a `logical-definition` (see Chapter 4) of a left hemisphere to the linguistic realization component. Before linguistic realization begins, focus information (discourse, temporal, and spatial) is extracted from the rhetorical proposition, as described in Section 8.4. After this, there are the three main levels of processing: semantics, grammatical relations, and syntax. At the first level, semantics, entities are extracted from the rhetorical proposition to fill verb case frames based on the type of rhetorical predicate (e.g., logical definition versus cause), the location of information in the rhetorical proposition (e.g., first position is agent, second is patient), and by examining explicit semantic markers (e.g., location, instrument, and beneficiary). For example, in Figure 8.3 the semantic markers `(location #<brain>)` and `(function (#<feature-recognition>))` in the input rhetorical proposition are mapped onto the semantic case roles, location, and function. The embedded-list format of rhetorical propositions allows for multiple agents, patients, etc. In addition to explicit semantic markers, each semantic role can be restricted (e.g., the "blue" block) or quantified (e.g., "all" blocks). The rhetorical predicate type indicates the action (e.g., `cause` → `#<cause>`). In Figure 8.3 the action for a logical definition is the copula, `#<be>`.

In the second stage of processing, each semantic role is mapped onto grammatical relations (e.g., subject, object). Thus in Figure 8.3 the agent, `#<left-hemisphere>`, becomes the subject and the patient, `#<region>`, becomes the direct object. In the third stage, syntactic constituents are built using focus information, the dictionary entry, and a declarative, feature-enhanced phrase structure grammar. In the example in Figure 8.3, the semantic markers of function and location of the left hemisphere become prepositional phrases in the final surface form. Finally, a morphological synthesizer examines the features on each leaf node of the tree to determine the final surface form of lexemes (e.g., plurals, verb endings, etc.). Punctuation is guided by the surface speech act type, in this case "assert" indicates a period. For simplification, TEXPLAN assumes that each utterance performs only one speech act although humans are capable of producing utterances which perform multiple acts (e.g., simultaneously inform and warn). Having briefly exemplified these levels of representation, the remainder of this chapter examines each successive level in turn.

## 8.3 Input: Surface Speech Acts and Rhetorical Propositions

The linguistic realization of an English utterance begins with a message consisting of two items: a surface speech act (e.g., assert, command, ask, recommend) and its associated rhetorical proposition. The basic function of the first item, the surface speech act, is to guide the selection of the appropriate sentence structure for the underlying propositional content. For example, the rhetorical proposition:

```
(event   (#<walk>)
         ((#<John-001>) (#<Mary-001>))
         ((location (#<park-001>))))
```

can be realized as a variety of forms depending on its accompanying surface speech act. For example, `assert` indicates declarative form ("John and Mary walk to the park."), `ask` indicates interrogative ("Do John and Mary walk to the park?"), `command` indicates imperative ("(John and Mary) walk to the park.") and `recommend` indicates the use of an obligation modal ("John and Mary should walk to the park."). This analysis could be extended to include other direct and indirect surface speech acts such as `suggest` ("John and Mary could walk to the park."), `ask-ability` ("Can John and Mary walk to the park?"), `ask-recommend` ("Should John and Mary walk to the park?") and so on (cf. Litman and Allen, 1987).

The second input to linguistic realization is the rhetorical proposition (McKeown, 1982) which accompanies the surface speech act. A *rhetorical proposition* is a rhetorical predicate (e.g., logical-definition) instantiated with particular propositional content from the application knowledge base. A *predicate semantics* maps the entities, relations, and values of the knowledge base to the appropriate positions in a frame associated with each rhetorical predicate. Rhetorical predicates that encode associations between propositions (e.g., cause, evidence) are termed *rhetorical relations*. Table 8.1 lists the types of

| Rhetorical Predicate | Predicate Semantics | Surface clue word |
|---|---|---|
| logical-definition | entity + superordinate + differentia | |
| synonymic-definition | equivalent entity but different indicator | |
| antonymic-definition | opposite entity | |
| universal-definition | property that applies to all individuals | |
| attribution | attributes or attribute-value pairs | |
| * purpose | function of entity or goal of action | "in order to" |
| location | locative information | |
| * illustration | instance or subtype | "for example" |
| classification | types or subclasses | |
| constituency | parts, components, steps, ingredients, etc. | |
| comparison-contrast | attribute-value comparison | "in contrast" |
| * analogy | similar but distinct attributes | "is like a" |
| inference | inference on comparison | "therefore" |
| conclusion | conclusion for syllogism premises | "therefore" |
| * cause | cause | "because" |
| * constraints | constraints on event/action | |
| * enablement | precondition of event/action | "so that" |
| * motivation | motivation of event/action | "motivates" |
| * evidence | evidence | "suggests" |
| event | incident, action, or process | |
| state | state | |

Table 8.1 Rhetorical Predicates and Predicate Semantics

rhetorical predicates in TEXPLAN along with their semantic constituents and signalling surface cues (* indicates a rhetorical relation). The rhetorical predicate abstracts distinct classes of information away from the details of the underlying model of the application and so it is relatively easy to port TEXPLAN from one domain to another (e.g., from neuropsychological diagnosis to mission planning to battle simulation), and even from one knowledge representation formalism to another (e.g., frames, rules, object-bases).

For example, Figure 8.4 relates a fragment of a knowledge base from the medical diagnosis system NEUROPSYCHOLOGIST (Maybury and Weiss, 1987) to a logical definition rhetorical proposition. As the example illustrates, given the domain entity, `#<brain>`, and the rhetorical predicate type, "logical-definition", the predicate semantics (dashed lines) return its superordinate and distinguishing features. In some cases, defining the semantics of a predicate involves simple retrieval from the underlying domain model (e.g., retrieving the superordinate in a generalization hierarchy) whereas in other cases it requires more sophisticated inference (e.g., determining entity differentia as defined in Section 4.3.1). Once instantiated with content from the knowledge base, this rhetorical proposition is then realized by the previously outlined mechanisms as the utterance "A brain is an organ for understanding located in the human skull."

One area for further work is to enhance the predicate semantics to include a richer language for expressing epistemological content as in Suthers's (1988a,b) view retriever. Furthermore, it should be possible to tailor the content of an individual predicate to the familiarity or relevancy of the information to the user (e.g., Sarner and Carberry, 1990). For simplicity, in TEXPLAN (as in McKeown, 1982) the entity provided to the predicate semantics from the text planner (originating or derived from the simulated user's query) is assumed to be a specific entity in the knowledge base.

While TEXPLAN's rhetorical predicates enumerated in Table 8.1 model common discourse elements of human-produced text, these alone will not generate well-connected and plausible text. Humans use knowledge of focus of attention and context to decide what to utter and how to utter it.

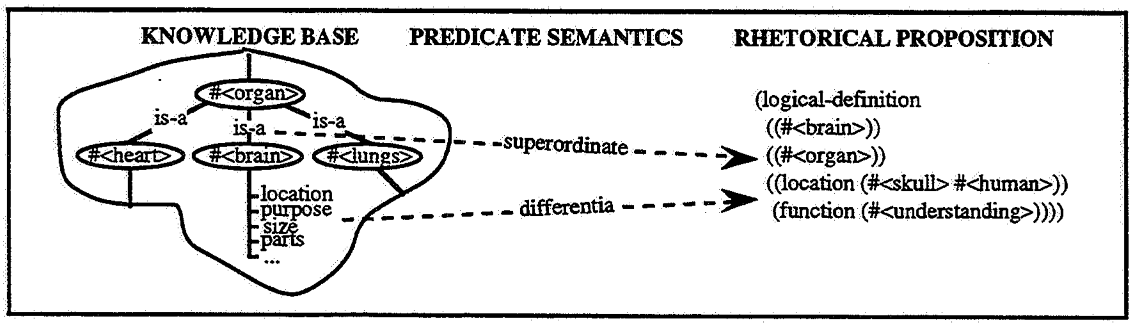



Figure 8.4 Logical Definition Related to Knowledge Base

## 8.4 Attentional Models

The realization of a rhetorical proposition is guided by focus of attention. The current focus is the entity placed at the forefront of our mind by implicit or explicit means, by grammatical constructs, or by phonological stress. While it is possible to focus on a proposition or an entire discourse segment (Webber, 1988b), TEXPLAN explicitly represents only domain entity focus although it can be argued that the hierarchical text plans act as a kind of global discourse focus. TEXPLAN represents three types of entity focus: *discourse*, *temporal*, and *spatial*. Motivated by Sidner (1979) and McKeown (1982), TEXPLAN records two types of utterance level *discourse focus* in two global registers: the *current discourse focus* (CDF), the *past discourse focus* (PDF) stack:

CDF -- generally the semantic actor, the subject of the sentence, the leftmost np of the sentence, and given.
PDF -- stack of past CDF

These registers are used to guide anaphoric reference. Entities for these global registers are extracted from the rhetorical proposition and updated after each utterance is produced, using default locations associated with each type of rhetorical predicate. In the logical definition instantiated in Figure 8.4, the default current discourse focus is #<brain>. The past foci stack contains a stack of current discourse foci from previous utterances. Instead of a simple stack, a more sophisticated memory device for past foci might have a decay register whereby with time (perhaps measured by the number of utterances produced) previously focused entities fade away from the forefront of discourse. In addition, a spreading activation mechanism (similar to that employed in CAPTURE (Alshawi, 1983)) could encourage entities that are related to the current focus of attention (in terms of the types and strengths of knowledge base links) to become more strongly in focus, as they are spoken about or referred to. McKeown's (1982) TEXT system recorded current and past focus as well as a potential discourse focus list (generally the semantic patient, object of the sentence, residing at the end of the sentence, new information), to guide the selection of future propositions.

In addition to recording discourse focus, TEXPLAN's tracks *temporal focus* and *spatial focus*, i.e., the event or entity which is the current focus in time or space. For example, consider the following:

1. Gorbachev addressed the Politburo at the Kremlin yesterday.
2. *He* did *it there* in front of live Soviet television.

The anaphoric references in utterance 2 refer to the discourse focus, temporal focus and spatial focus of utterance 1 ("he" refers to "Gorbachev; "it" refers to "addressed"; "there" refers to "the Kremlin"). To produce these kinds of references as well as adverbials, global registers similar to those discussed above for discourse focus record the current and past temporal and spatial focus. The temporal focus is updated by examining the events in rhetorical propositions and the spatial focus by examining entities with locations in space (e.g., places, people, things). Temporal focus is used primarily in narrative plans to guide the realization of temporal adverbials, whereas spatial focus is used to guide the realization of locative adverbials in locative instructions (i.e., route plans). Temporal focus can also be used to guide the selection of verb tense, since the Reichenbachian reference time can be interpreted as the time associated with the past temporal focus and the Reichenbachian event time as the time associated with the current temporal focus. The Reichenbachian speech time is recorded in a variable that is bound just before linguistic realization.

While temporal and spatial focus can parallel the discourse focus, there may be instances in which time or space remains constant while the discourse focus progresses (e.g., entity description amidst temporally sequenced event narration). Furthermore, discourse, temporal, and spatial focus are often distinct, as illustrated by the anaphoric references in the above Gorbachev example. Sections 8.6.1 and 8.6.2 detail how focus affects surface form as in the production of anaphora and temporal and locative adverbials. But first the rhetorical proposition must be interpreted by the semantic component.

## 8.5 Semantic Interpretation of Rhetorical Propositions

After focus information is recorded for the rhetorical proposition, the rhetorical proposition is mapped onto deep case roles (Fillmore, 1968) such as action, agent, patient, beneficiary, instrument, location, function, external location, manner, and so on. A variety of semantic case roles have been suggested ranging in length from few (nominative, ergative, locative) (Anderson, 1971) to many (Sparck Jones and Boguraev, 1987), and they can be deep or shallow. TEXPLAN's case roles are not claimed to be complete, but rather sufficient for the given rhetorical predicates.

TEXPLAN semantically interprets a rhetorical proposition based on the position of items in the rhetorical message and on their semantic markers. For example in Figure 8.3 the input rhetorical proposition is a list of the form:

```
(logical-definition  ((#<left-hemisphere>))
                     ((#<region>))
                     ((location (#<brain>))
                      (function (#<feature-recognition>))))
```

The first item in the input is the type of rhetorical predicate, `logical-definition`, which is associated with the semantic action, "be". The second item is the list `((#<left-hemisphere>))` which is mapped onto the semantic agent. The third item is the list `((#<region>))` which is mapped onto the semantic patient. There may be multiple agents or patients and both agent and patient may include special semantic markers which restrict their interpretation, such as *location, external-location, function, instrument.* These semantic markers and their associated content eventually translate to prepositional phrases (e.g., "located in", "on", "with", and "for"). The fourth item in the rhetorical proposition includes similar semantic markers which apply to the entire proposition. In the rhetorical proposition from Figure 8.3, the location and function of the semantic agent, `#<left-hemisphere>` are mapped onto the semantic roles of location and function. A richer range of semantic markers and their corresponding deep case roles would be required to represent and generate other surface forms (e.g., *means* as in "They went to Kyoto by train.").

One problem is how case roles are mapped onto syntactic constituents. Fillmore (1977, p. 70) recognized the need for "a level of representation including the grammatical relations subject and object" to bridge this semanticosyntactic gap. In TEXPLAN, this is accomplished by grammatical relations, considered in the next section.

## 8.6 Grammatical Relations

Relational Grammar (Perlmutter and Postal, 1977; Pullum, 1977; Perlmutter and Soames, 1979; Perlmutter, 1980; Perlmutter and Rosen, 1984) was developed to fill the gap between the semantic units of a case representation (such as agent and patient) and phrasal constituents (such as NP and VP) by exploring language dependent grammatical relations (e.g., subject, object). Relational Grammar uses a hierarchy of sentence participants so that in English, for example, the subject is 1, the direct-object is 2, and the indirect-object is 3. Rules can then capture generalities such as "to form the passive, promote 2 to 1" (direct-object to subject). In this case, the 1 element becomes the chômeur (French for "unemployed"), so it can either be dropped from the sentence or transferred to a satellite phrase.

Some inter-lingual studies support a relational level of analysis (Perlmutter, 1980). Winograd (1983, p. 324) points out that in a language with a more developed case system (e.g., Russian and Japanese), the use of Relational Grammar's verb-centered analysis could be even more beneficial. In addition, some psycholinguistic evidence (Bock and Warren, 1985; Bock, 1987) supports this intermediate level of processing.

Relational Grammar was first used for linguistic analysis in the GUS (Genial Understanding System) (Bobrow, 1977). GUS parsed input in two phases, first into grammatical registers (subject, direct-object, indirect object) with prepositional phrases placed in an adjunct list, and second into a structure indicating verb-case roles. Like GUS, with its intermediate level of representation, MUMBLE (McDonald and Pustejovsky, 1985c, p. 804) employs linguistic realization classes which specify a range of subject-verb-object patterns including active, passive, gerundive, and nominalization constructs. However, many linguistic realization components in natural language generation systems simply map semantic units directly onto syntactic structures, as in the original dictionary component of TEXT (McKeown, 1985, p. 167), which translates knowledge base entities into phrasal constituents via a hand-encoded dictionary.

As in GUS, TEXPLAN has an explicit representation of relational constituents including verb, subject, indirect and direct objects, and adjuncts. TEXPLAN uses the past and current discourse, temporal, and spatial focus caches to guide eventual syntactic structures via the intermediate relational choices for case structure elements. Relational constituent assignment is controlled by discourse focus information and predicate types. For example, passive voice is chosen over the active voice to stress what is normally the object by promoting it to the subject position as in:

(a) "John hit Mary with the stick."
(b) "Mary was hit by John with the stick."

We select (b) to emphasize Mary. If we want to emphasize that John (not Mark) hit Mary we could use it-extraposition ("It was John who hit Mary"), or intonational stress ("*John* hit Mary").

Assume, for example, a sentence is being generated where the message formalism translates to the grammatical relations: verb → `#<cause>`, subject → `#<alcoholism>`, and object → `#<amnesia>`. This might be realized as *Alcoholism causes amnesia.* In contrast, focus information may suggest that the utterance is best described from the perspective of *amnesia.* In this case the relational grammar would indicate that to achieve this the 2 (object) should be promoted to 1 (subject). In the typical case, the verb would be passivized (be + past participle of main verb), the preposition "by" would be added before the new constituent of the 2 (object) register. Generation would eventually culminate in the surface form: *Amnesia is caused by alcoholism.*

However, there are some verbs (like the one in this sentence) which cannot be passivized. In these cases (e.g., "be", "have") syntactic ordering must account for focal prominence. So we can utter "It was a brain tumor (not a stroke) that killed the patient." to emphasize the semantic patient, "tumor". In TEXPLAN, there-insertion is used to promote the object to the subject position where the passive construction is not possible (e.g., with a copula verb), and it-extraposition is used to stress the current subject (e.g., "It was John who hit Jill").

Not only prominence (intonational or structural), but also lexical connectives can sew together discourse. The rhetorical function of an utterance in discourse suggests appropriate connectives or clue words (e.g., illustration → "for example", cause → "because", conclusion → "therefore"). Their position in the relational structure is determined by the particular predicate type (e.g., "therefore" is sentence initial for conclusion propositions.) In addition to these rhetorical predicate-driven connectives, temporal connectives that indicate event co-occurrence ("simultaneously"), event repetition ("again"), or lateral shifts in temporal focus ("meanwhile") are inserted to temporally relate events.

After determining relational structure and inserting connectives, TEXPLAN employs syntactic experts to build relational grammar constituents (e.g., subject, verb, object) using the rhetorical proposition, the surface speech act, and focus information. The next two subsections indicate (1) how relational experts translate relations to phrasal constituents (e.g., noun phrases and adverbials) and (2) how focus affects the realization of phrasal constituents.

### 8.6.1 Relational Grammar Experts

Relational constituents (verb, subject, objects, and adjuncts) are built by procedures which are experts at forming syntactic phrases to realize these relational constituents. Provided with the semantic message together with syntactic and focus constraints, these procedures attempt to generate well-formed constituents. There are four principal relational grammar experts, namely those for constituents that will appear as noun phrases (NP), verb phrases (VP), adverbials (ADV), and prepositional phrases (PP) at the next level. Subject and object are constructed by the NP builder, verb by the VP builder and ADV builder, and adjuncts by the PP and NP builder. These experts can produce the most basic syntactic constituents (e.g., noun and verb phrases) as well as conjunction and quantification (although no formal treatment of these complex linguistic phenomena is claimed).

As individual relational grammar experts are constructing syntactic constituents, they consult the dictionary (detailed in the next section) to look up the entry for each domain entity found in each grammatical relation. Lexical choice and lexical representation (cf. Goldman, 1975; Nirenburg, 1987; McDonald, 1980) were not a major focus of this work, so domain entities are mapped onto English surface forms in a domain-dependent manner and there is no lexical variation (cf. Granville, 1984).

The NP builder consists of the pattern: `NP → quantifier + article + adjective-list + nominal-modifier-list + head-noun + post-modifiers`. The `adjective-list` incorporates adjectives and ordinals while the `nominal-modifier-list` includes only nominals. Compound nouns were generated on the assumption that the message order passed from the semantic component indicates the head noun as distinct from modifying nouns. The proper handling of compound nouns, however, is a major enterprise involving word sense, nominal phrase structure, and semantic word relations (see e.g., Sparck Jones, 1985) and so the approach adopted in the system is fairly simple.

The NP builder selects articles guided by syntactic and focus constraints. The article selection algorithm shown in Figure 8.5 considers both the entity and its lexical entry as returned by the dictionary mechanism. The selection between definite and indefinite article is based first on local syntactic constraints and then on discourse distinctions of given and newness. Another distinction which would indicate definiteness is considering whether the entity is an individual or if it is an instantiable class that can be uniquely identified (i.e., whether the referent is recoverable by the hearer) (Matthiessen, 1987, p. 256-257). For example, when generating the first utterance in a text plan that defines a brain, with no previous discourse context TEXPLAN says *A brain is an organ located in the human skull.* Both the subject and object have indefinite articles as both are *new*. In contrast, if the entity `#<organ>` had been previously introduced in the discourse, then it could be referred to using the definite article (i.e., "the brain"). While it can be argued that the noun phrase within the prepositional phrase in the example could also use an indefinite article, as it too represents new information, the adjectival modifier specifies the *human* skull and therefore the definite article is chosen. Article choices are morphologically consistent with the subsequent

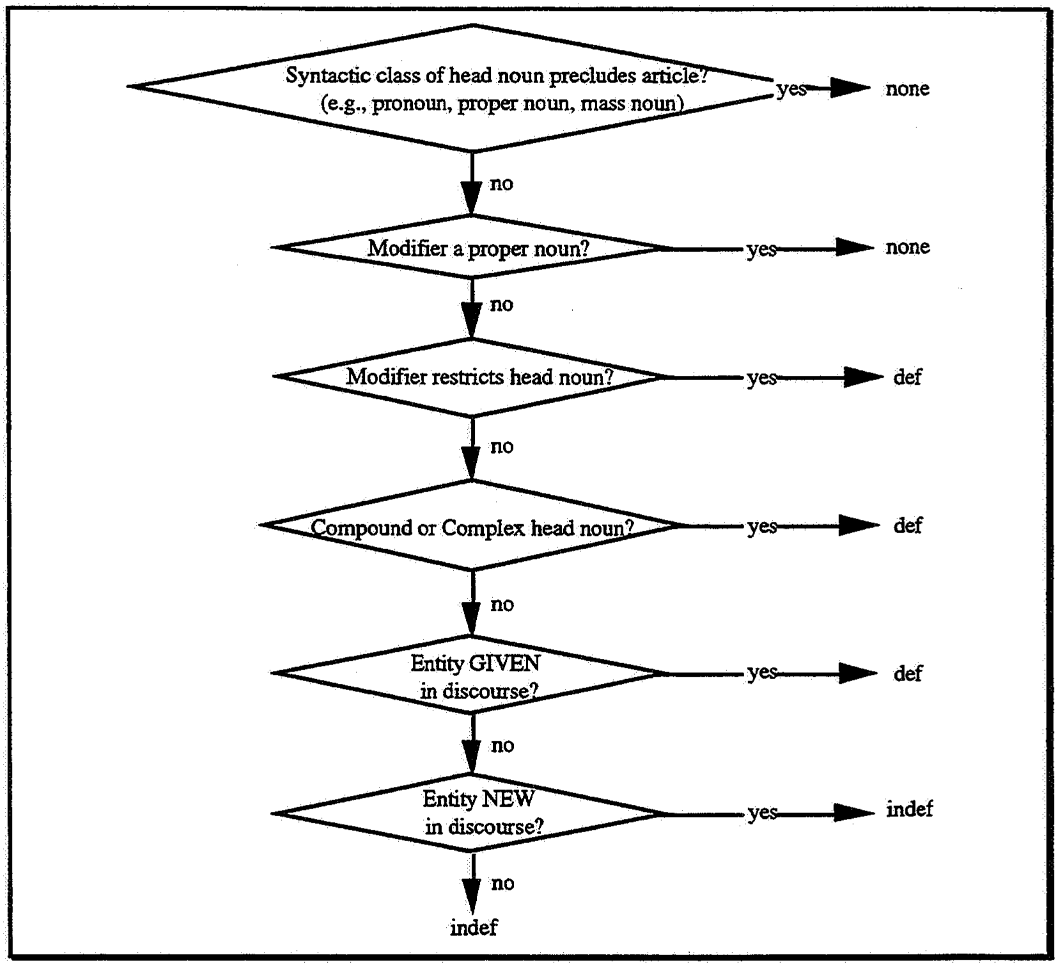



Figure 8.5 Article Selection Algorithm

lexical item, that is the morphological component discerns between "a" and "an" based on vocalic structure of the next lexeme in the linear sequence as in the above choice of "an" in "an organ".

The VP builder consists of the pattern `VP → verb + (NP + ADV[locative] + PP[locative])` or `VP → auxiliary + verb[past-participle] + particle`, depending upon the provided voice (where parentheses indicate optionality and brackets include necessary features). If the voice is active, the VP builder will use the first rule shown above where the semantic action is the subject. In contrast, if the voice is passive, the VP builder will select an appropriate auxiliary (such as "be") followed by the lexical entries for the verb, followed by an appropriate particle if necessary, which will eventually realize as, for example, "is contained in" or "is indicated by". The surface speech act also guides verb choice (e.g., the surface speech act command indicates imperative form; ask indicates interrogative; recommend indicates the use of the obligatory modal "should"; suggest indicates "could"). The VP builder uses the previous temporal focus (indicating Reichenbachian reference time) and current temporal focus (indicating event time) as well as the global variable bound to the speech time to determine verb tense. In addition to tense, spatial adverbials which indicate a distance and direction as in "Two kilometers East" can modify the verb. These are constructed by the adverb builder.

The ADV builder consists of the pattern ADV[type] → adverb[type], where type can be locative, temporal, and so on. For example, the temporal adverbial "Two minutes later" is produced by computing the differential between the time of the current temporal focus and the time of the previous temporal focus. Similarly, a locative adverbial such as "Five kilometers Northwest" is computed by considering the location and direction of the current spatial focus relative to the previous spatial focus. Whereas spatial adverbials default to VP modification (see previous paragraph), temporal adverbials modify the sentence as a whole as in S → ADV[temporal] + NP + VP or S → NP + VP + ADV[temporal].

Finally, a PP builder follows the pattern PP → preposition + NP, recursively calling the NP builder to complete its description (passing along focus information to allow pronominalization if an entity is in current focus). The preposition is provided to the routine by examining the semantic case role given with the entity (e.g., location -> "located in"). Several case roles are eventually realized as PPs such as location ("located in"), external-location ("on"), instrument ("with") and function ("for"). Also, a temporal prepositional phrase (e.g., "at 8:20 on Tuesday December 12") or locative prepositional phrase (e.g., "located in block 33UU55030") can arise from an event or entity where there is not previous temporal or spatial focus to refer to which makes production of a temporal or spatial adverbial (as above) not possible.

To illustrate these phrasal experts, consider again the example in Figure 8.3. The semantic action, #<be> is used by the VP builder to select the verb "be". The semantic agent, #<left-hemisphere>, is used by the NP builder to select a determiner and head noun to produce the subject "the left-hemisphere" (the definite determiner is selected since the modifier "left" restricts the interpretation of "hemisphere"). Similarly, the semantic patient, #<region>, is used to produce the direct object "a region". The semantic function, #<feature-recognition>, eventually becomes the purposive prepositional phrase "for feature-recognition" whereas the semantic location, #<brain>, is mapped onto the locative prepositional phrase "in the brain" using the PP builder which recursively calls the NP builder. These constituents are then combined using the phrase structure grammar detailed in the next section to produce the final utterance "The left-hemisphere is a region for feature-recognition located in the brain."

TEXPLAN degrades gracefully when unable to translate or build certain phrasal constituents by attempting to utter what it can. For example, if the dictionary has all lexical entries needed to produce the utterance "The brain is an organ located in the skull." except for "skull", then the system will still utter "The brain is an organ." instead of failing. For greater perspicuity, future work could investigate implementing TEXPLAN's procedural syntactic experts declaratively.

### 8.6.2 Anaphora

TEXPLAN's linguistic realization component generates two forms of anaphoric or backward-looking reference. The first is the use of pronominalization to refer to the previous discourse focus (which is the same as the current one) and the second is the use of temporal or spatial adverbials which refer to the space or time in which preceding utterance took place.

In the first case, the decision to pronominalize is made by the NP builder. The pronominalization algorithm deals only with intersentential definite pronominal anaphora and simply states that if the referent is equivalent to the previous discourse focus and it is given in the current utterance (i.e., it is the current discourse focus), then pronominalize. Referring expressions are selected from the set of possible pronominals by unifying syntactic features such as person, number, and gender. These could be extended to include, for example, animacy. Fillmore (1977) suggests that entities in an event are *perspectivized* and claims a need for a saliency hierarchy—a prioritized list of foreground choices which can be used to decide on focus. He suggests an animacy hierarchy can aid perspectivization decisions. Given a choice, egocentric people tend to focus first on humans, then animate things, and finally on inanimate objects. Animacy knowledge could easily be added to lexical entries and TEXPLAN's focus algorithm could be adapted to make such decisions (indeed the information would also be useful for selectional restrictions). The text below, produced by TEXPLAN, illustrates the results of using the discourse focus algorithm:

```
A brain is an organ for understanding located in the human skull.  It
has an importance value of ten.  It consists of two regions: the left-
hemisphere and the right-hemisphere. The left-hemisphere, for example,
is a region for feature-recognition.
```

The subject in sentences two and three is attenuated since they are forefronted in the reader's mind. In contrast to this intersentential pronominalization, TEXPLAN's intrasentential pronominalization algorithm simply states that if the current focus of an utterance is repeated subsequently in that utterance, those references can be pronominalized. The importance of this is evident, for example where the proposition `tell(Mary-1, John-1, angry(Mary-1))` can be realized as "Mary told John that she was angry" versus "Mary told John that Mary was angry", which has a rather different implication.

Just as discourse focus guides pronominalization, temporal and spatial focus can affect surface form as indicated in the description of the VP builder in the previous section. Thus, when producing adverbials, TEXPLAN can attenuate the surface form by making reference to the current focus in space or time. For example, when producing narrative text, once a temporal focus has been established (e.g., "Offensive Counter Air Mission 100 began mission execution at 8:20 Tuesday December 2, 1987.") successive utterances can refer to this anchor point (e.g., "Ten minutes later ..."). Similarly, in locative instructions subsequent utterances can refer to previous spatial anchor points both in terms of distance and heading (e.g., "`From Wiesbaden take Autobahn A66 Northeast for thirty-one kilometers to Frankfurt-am-Main.`"). The realization of other constituents might also benefit from spatial information.

For example, Wahlster et al. (1978) discuss how entities could be identified during noun-phrase generation by using two-dimensional spatial relations (e.g., "in front of", "to the right of") to distinguish entities by referring to their spatially related neighbors.

TEXPLAN's anaphoric generation is only a first step, as longer texts will require reference mechanisms which incorporate more than just syntactic, recency, and focus information. For instance if we introduce both "Alzheimer's disease" and "Huntington's disease" in discourse, subsequent nominal reference must uniquely identify the entity in discussion: the word "disease" alone is insufficient. Referential procedures that are sensitive to uniqueness and prototypicality (as defined in Chapter 4 and Appendix A) are therefore required to avoid this kind of ambiguity. Appelt (1985) investigated the generation of referring expressions guided by models of the hearer's knowledge and beliefs. Dale (1989, p. 73) examined the production of one-anaphora by considering if the current referent shared properties with the previous referent, as in "Slice the large green capsicum. Now remove the top of the small red one." Reiter (1990) examined producing implicature-free referring expressions by considering the user's domain and lexical knowledge, and Carter (1983) investigated similar issues.

There are several other anaphora problems which require further research. Hobbs (1978) found that after examining 100 subsequent pronominalizations in three very different texts, 98% of the antecedents were in the same or the immediately preceding sentence. Nevertheless, long distance pronominalizations do occur and their relation to discourse structure remains an important research issue (cf. Sidner and Grosz, 1986). Other forms of anaphora that have received little attention include VP anaphora (e.g., "John was sleeping. Mary was doing it too."), sentence anaphora (e.g., "He won $10,000 in the lottery. It made him rich.") and discourse anaphora (e.g., "That was very confusing" where "that" refers to an entire discourse segment) (Webber, 1988b). Cataphora (forward-pointing references) and exophora (extratextual references) also require further investigation.

Having considered how Relational Grammar acts as a bridge between semantics and syntax, and how individual constituents are realized guided by models of (discourse, temporal, and spatial) focus, the next section details the syntactic grammar.

## 8.7 Unification Grammar and Lexical Semantics

TEXPLAN's surface syntactic knowledge is represented declaratively in a Phrase Structure Grammar (PSG) based on an extension of Context Free Grammar (CFG) and motivated by Generalized Phrase Structure Grammar (GPSG) (Gazdar, 1982). Typical rewrite rules such as "S → NP + VP" are augmented with features which constrain the possible well-formed syntactic trees. These rules cover agreement and morphology and could be extended to include missing/moved constituents. For example, the active, present tense sentence level rule in the linguistic realization component is:

```
S    [(type declarative) (voice active) (tense present)] →
NP   [(count C?) (person P?) (gender G?)]  +
VP   [(count C?) (person P?) (tense present) (voice active)]
```

Each rule has an associated symbolic name (s<dec>→ np+vp, for the above rule). The capitalized characters in the rule (e.g., S, NP, VP) indicate non-terminal symbols, which are followed by a list of feature-value pairs which dictate syntactic constraints. Note that some feature values are constants whereas others are variables (italicized and terminating in a question mark) which indicate feature agreement. In the above rule, for example, the count (e.g., plural) and person (e.g., third-person) feature values of the noun phrase and verb phrase must agree as indicated by variables *C?* and *P?*. The grammar includes rules for active and passive sentences, multi-sentential connectivity (e.g., conjunction and disjunction) and relative clauses, as well as for phrasal constructs (NP, VP, PP, etc.). The values of certain features are determined by higher level choices. For example, sentence type (e.g., declarative, interrogative, imperative) is determined by the surface speech act and voice is constrained by focus and verb information. The grammar is documented in Maybury (1987b, volume II) along with grammar development and application tools (e.g., grammar rule editor, preparser for efficiency).

The process of generation is handled by the process of unification. Unification consists of using the grammar and features to build constituents which are placed on a well-formed sub-string table (WFSST) or chart (see Pulman, 1987 for detail). The unifier percolates features up the chart (by matching and then binding feature variables), and generates all possible syntax trees from the given lexical entries. At the end of the generation, another routine simply reads off the completed trees (or partial trees, as in the case of ellipsis or fragments). The first successful syntax tree is selected, which is a crude selection mechanism—something more sophisticated is needed. The unbound variables in the syntax tree are bound with values from their agreeing constituents.

### 8.7.1 Dictionary

The leaf nodes in the WFSST are lexical entries. Lexical entries are listed in the dictionary in the format *<token syntax semantics realization>* where *token* refers to a lexical entity, *syntax* includes categorical, agreement and morphological information, *semantics* includes a logical form meaning representation of the lexical item, and *realization* indicates the actual translation of the domain token into natural language. For example, the entry for the singular, first person, present tense, of the verb "to be" is:

```
TOKEN:        be
SYNTAX:       ((class verb) (subclass copula) (number singular) (tense present) (person first))
SEMANTICS:    (lambda (P?) (lambda (wh?) (P? (lambda (y?) (equal wh? y?)))))
REALIZATION:  "am"
```

The surface form "am" is related to feature-value pairs of syntactic constraints and a compositional, λ-calculus meaning representation (Montague, 1974). Lexical entries consist only of root or irregular forms of words. In normal cases, the syntactic feature list of a word is modified to indicate morphological variants. For example, the syntactic feature list of the plural entry of the verb "contain" is ((class verb) (subclass transitive) (number plural) (tense present) (person third)), which is modified to ((class verb) (subclass transitive) (number plural) (tense past-participle)) to form the past participle. These features subsequently guide the morphological synthesizer in modifying root word forms. In addition to this syntactic information, each lexical entry includes a $\lambda$-calculus semantics field. Together with the above phrase structure syntactic rules (each of which have an associated $\lambda$-calculus meaning representation), the semantics can be used to convert syntactic trees to logical form (Pulman, 1987), although the current implementation uses the deep case semantics detailed above. Levine (1990, forthcoming) uses similar compositional Montague (1974) semantics in a bidirectional question-answering system.

The dictionary sub-system built for TEXPLAN contains dictionary entry generation, access, edit, and removal functions. To facilitate portability, a kernel dictionary was developed which contains frequently used words such as numbers, determiners, pronouns, prepositions, punctuation, conjunctions, connectives, and core verbs. This was exploited when porting TEXPLAN between applications for evaluation. For efficiency and compactness, the dictionary lists only root forms of regular nouns and verbs, allowing a morphological synthesis component to produce the other variants based on feature values (e.g., number, tense). Furthermore, instead of explicitly listing all entities in the underlying domain in the dictionary, if dictionary look up fails to find a given token it automatically queries the underlying application in an attempt to infer the given token's lexical category. For example, concepts in the generalization hierarchy are assumed to be count nouns, attributes are assumed to be nouns, attribute values are assumed to adjectives, and events default to verbs. For instance, the entity #<fighter> and its attributes such as #<length> or #<speed> are assumed to have the syntactic properties ((class noun) (subclass count) (number *singular-or-plural?*)) where the *singular-or-plural?* variable is subsequently bound by context. In contrast, the syntax for numerical values defaults to ((class number) (number singular)) if the ordinal is one and ((class number) (number plural)) if it is greater than one. In contrast, non-numerical attribute-values such as "big" and "red" have the default syntax of ((class adjective) (subclass attributive)). An event has the default syntax ((class verb) (subclass transitive) (number *singular-or-plural?*) (tense present) (person *1-2-or-3?*)) where the variables for number and person are bound with subsequent context. Lexemes with irregular syntax, semantics, or realizations must be listed explicitly in the lexicon.

In addition to inferring syntactic classes and features, the surface form of entities can sometimes be computed, for instance by parsing their printed representations (e.g., #<intersection-A9-R52> → "intersection A9-R59"). A trivial case is the automatic calculation of the realization of numerical entries. For example, in TEXPLAN, for numbers with cardinality below 100, the dictionary automatically produces the realization of numbers in textual form (i.e., "ninety-nine"). In contrast, numbers with cardinality over 100 are listed numerically with appropriate place indications (e.g., "1,435"). Automatic acquisition of lexical knowledge, either from the underlying application (Weischedel, 1989) or from on-line sources (e.g., dictionaries or corpora (cf. Boguraev, 1989)), will become increasingly important in interfaces to application systems as the underlying knowledge bases grow in size and complexity.

To complete the production, TEXPLAN linearizes the syntax tree, morphologically synthesizes lexical entries and then applies final orthographic conventions. Morphological synthesis is guided by the syntactic features of lexemes on the leaf nodes of the syntax tree based on an inverted version of Winograd's (1972, p. 74) morphological analysis finite state machine. Orthography includes text layout (spacing, pagination, new lines) and conventions such as capitalization and punctuation. Text layout can be signalled by the intentional structure of the text plan, for example, the introduction of a major rhetorical act such as describe or narrate signals a new paragraph. At this stage sentence initial words are capitalized and punctuation is determined by examining the surface speech act (e.g., assert → ".", ask → "?", exclaim → "!"). While not implemented, focal prominence (e.g., introducing key terminology) could be signalled by a contrastive font (e.g., times, courier, London), size (e.g., 10, 12, 14 point), and/or style (e.g., *italic*, **bold**, underlined). Abbreviation also could be used for terseness which could perhaps be guided by a model of the user's lexical or domain knowledge.

## 8.8 Summary

This chapter describes how the hierarchical communicative plan produced by TEXPLAN's strategic generator is realized as English text. The chapter outlines the multiple layers of linguistic representation that transform a surface speech act and its associated rhetorical proposition onto surface form via case semantics, grammatical relations, and a feature-enhanced phrase structure grammar. Mechanisms for morphological synthesis and orthographic layout are also discussed. The linguistic realizer exploits discourse, temporal, and spatial focus to guide structural choices, referring expressions, tense, and adverbial production. This chapter notes the novel use of a relational grammar and the ability to plan and realize text in either a serial or interleaved fashion. The chapter concludes by indicating several areas which require future research including tense, aspect, adverbials, anaphor, and the relationship of planning and realization.

While linguistic realization was not the principal focus of this dissertation, the tactical generator does provide adequate and reasonably well-motivated mechanisms for translating a message—a surface speech act and its rhetorical proposition—onto surface form. Furthermore, surface choices are guided by several different types of focus (discourse, temporal, and spatial). This, together with connectives motivated by the types of rhetorical predicates (e.g., "for example", "therefore") and temporal focus (e.g., "and then") help to enhance the cohesion of the text. Finally, relational grammar seems to be a convenient level of representation between deep case roles and syntactic constituents which helps capture not only active/passive distinctions but also supports multi-lingual text generation.

While the separation of discourse, semantics, grammatical relations, syntax, and morphology allows for local control of linguistic issues, there are still many linguistic phenomena which require further investigation. At a syntactic level this includes ellipsis and structural ambiguity. Lexical ambiguity and word choice is also an important issue. A more formal syntactic and semantic account of tense, aspect, and adverbials is required, as in Schubert and Hwang's (1989) investigation of duratives, manner adverbials, locatives, and negation. In general, a more sophisticated linguistic realization component (e.g., McDonald, 1980) could have enhanced the syntactic fluency of the text output by, for example, the use of subordinate clauses, gerundives, nominals, and lexical variance. The produced texts should therefore be evaluated more with respect to their content and form than their linguistic fluency. We turn to evaluation in the next and final chapter.

# Chapter 9

# SUMMARY, TESTS, EVALUATION, AND FUTURE DIRECTIONS

Ancora Imparo

*Michelangelo Buonarroti*

## 9.1 Summary

This final chapter summarizes the dissertation, indicating principal claims and contributions. These claims are supported by a description of the various procedures used to test and validate TEXPLAN. The research is then evaluated with respect to its goals and with respect to other computational models of explanation with similar goals. The chapter concludes by indicating key problems which require further research.

When interacting with users, knowledge based systems often must define terminology and concepts, narrate events and states, elucidate plans, processes, and propositions, and support recommendations or conclusions. While application systems *represent* this range of information, a major problem is that they often cannot effectively *present* this to a user because they do not have mechanisms which can select, structure, order, and linguistically realize a range of explanations. These explanations are often lengthy and so require more than just generating sentences that are locally cohesive; they require mechanisms for global coherence. Motivated by an analysis of human-produced explanations, this dissertation claims that multisentential text can be characterized by a tripartite theory of communicative acts: rhetorical acts, illocutionary acts, and surface speech acts. Communicative acts with associated effects are formalized as over sixty compositional plan operators in a computer system that both plans and realizes multisentential English text. The implemented system, TEXPLAN, produces various generic discourse types each of which are intended to have unique effects on the user's knowledge, beliefs, and desires. Taken as a whole, the communicative acts characterize four types of text (see Figure 9.1): (1) entity description (definition, division, detail, comparison, and analogy), (2) event narration (reports, stories, biographies), (3) plan,

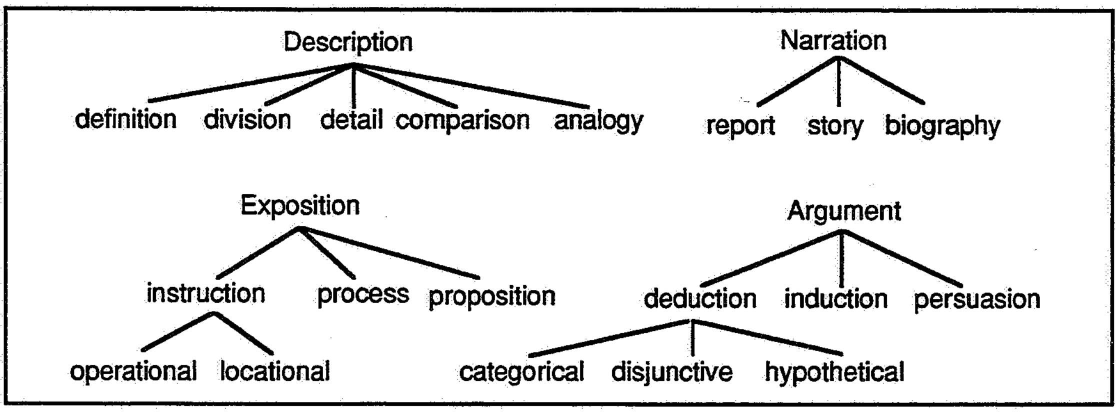



Figure 9.1 Text Types: Description, Narration, Exposition, Argument

process, and proposition exposition, and (4) deductive, inductive, and persuasive argument (used to support a claim or evoke action). Motivated by the need to communicate information about objects, events, and locations (e.g., in event reports and route plans), three distinct notions of focus—discourse, temporal, and spatial—were exploited to guide the order and realization text.

This work is novel from several perspectives. First, this dissertation computationally investigates the claim that there are distinct but interrelated types of text that can be characterized by their content and the types of communicative acts they employ. The text plans produced by the implementation consists of two structures: a communicative action decomposition (used to update the discourse model) and a related effect decomposition (used to update the user model). The action decomposition includes both locutionary and illocutionary acts (with associated rhetorical propositions) which are organized into higher level abstractions, termed rhetorical acts. Each of these three types of communicative acts—locutionary, illocutionary, and rhetorical—is formalized as a plan operator, a list and expository grammar of which can be found in Appendix B. The dissertation considers the nature of these high-level plan operators which achieve discourse goals and how these plan operators are altered as a consequence of the user's knowledge, beliefs, and desires. The resulting computational system produces a broader range of texts than previous work, including description, narration, exposition, and argument. This is made possible only by taking account of a number of constraints to guide the selection, order, and realization of a range of rhetorical propositions, including three distinct notions of focus: discourse, temporal, and spatial focus. The key claims of the dissertation are summarized in Figure 9.2.

1. *Integrated, Tripartite Theory of Communicative Acts* -- Multisentential text can be characterized at the level of rhetoric, illocution, and locution which are hierarchically related and have distinct effects on the addressee's knowledge, beliefs, and desires. These are formalized as over sixty plan operators in the computational system, TEXPLAN.

2. *Rhetorical Predicates* -- Twenty-one rhetorical predicates characterize distinct communicative content and are related to knowledge base relations (see Table 8.1).

3. *Multiple Text Types* -- description, narration, exposition and argument (and several subtypes) are formalized as particular collections of communicative acts with associated rhetorical predicates.

4. *Tripartite Theory of Focus of Attention* -- discourse, temporal and spatial focus, which play a local cohesive role in text, are identified and formalized.

5. Since the plan operators distinguish between a communicative act (e.g., define, compare) and its communicative effect on the addressee's cognitive state, the implemented system, TEXPLAN, is able to build a distinct *discourse model* and *user model.*

Figure 9.2 Principal Claims and Contributions

This work differs from recent work which plans rhetorical relations (Hovy, 1988a; Moore, 1989) in that it recognizes and formalizes the distinction between the rhetorical relations in a text (e.g., evidence, enablement, purpose) and the rhetorical acts establishing these. Rhetorical acts (e.g., describe, compare, define, narrate, argue, convince, persuade) are expressed through other rhetorical acts or by illocutionary acts (e.g., inform, request), which are in turn expressed by surface speech/locutionary acts (e.g., assert, command) which may have associated rhetorical predicates (e.g., logical-definition, cause). These distinct classes of executable communicative acts (rhetorical, illocutionary, or surface speech/locutionary acts) are formalized as plan operators in a common framework. These communicative acts are then reasoned about by a hierarchical planner in order to produce a text plan—an executable action decomposition—that achieves some given discourse goal. In contrast, using rhetorical relations, Hovy's (1988a) system constructs a rhetorical structure over propositions and Moore's (1989) system constructs a rhetorical structure over illocutionary acts.

Finally, this dissertation investigates the specific *expected* effects of communicative acts on the user's knowledge, beliefs, and desires. For example, in TEXPLAN the intended effect of informing the user of the logical definition of an entity (an "elaboration" in Rhetorical Structure Theory) is that the user knows about the entity and about its superclass and its distinguishing features. This differs radically from the effect of informing them of a synonymic definition, the intended effect of which is to get the user to know about the entity, and its synonymic relation with some other known entity. In addition to these cognitive effects on the user's knowledge, beliefs and desires, this dissertation briefly considers how psychological effects (e.g., fear, suspense, surprise) relate to the suppression or (re)ordering of communicative acts, although these psychological models have not actually been implemented.

## 9.2 Tests

The tripartite theory of communicative acts was evaluated via a computational implementation which was tested using a variety of techniques. These include (a) the generation of all types of rhetorical predicates illustrated in Table 8.1, (b) the generation of all the types of text (which have associated discourse goals) detailed in Chapters 4-7, and (c) the generation of multisentential text from several applications in diverse domains. The generation of text in different discourse and user contexts was also used to test the implementation. Testing resulted in the production of hundreds of texts, ranging from single utterances to multiple paragraphs.

TEXPLAN produced rhetorically structured text from several independent and preexisting knowledge based systems, including the Knowledge Replanning System (KRS) (Dawson et al., 1987), Land Air Combat in ERIC (LACE) (Anken, 1989; Hilton and Anken, 1990), the Map Display System (MDS) (Hilton, 1987; Hilton and Grimshaw, 1990), and NEUROPSYCHOLOGIST (Maybury and Weiss, 1987). These applications varied on a number of dimensions including their domain (resource allocation, aircraft missions, cartography, neuropsychology), their generic task (simulation, planning, diagnosis), their underlying representational formalisms (object-oriented programming, frames, hybrid rule/frame), and their principal problem-solving techniques (e.g., generic methods, constraint propagation, heuristic classification). For some of them, as indicated in the thesis, many different outputs were generated. But even this broad range of applications did not provide a rich enough basis from which to produce all text types, so a few small knowledge bases were hand-encoded, ad-hoc, to act as the basis for generating story narration (the pilot story), operational instructions (baking cookies), process and proposition exposition (the heart), deductive argument (Socratean syllogism), and inductive argument (devaluation of education). Some text types were also tested in multiple domains (e.g., description), though others were only produced in one domain (e.g., LACE narrative reports). Appendix C illustrates several text types from various domains.

### 9.2.1 Tests of Rhetorical Predicates

At the utterance level, tests were run to validate both the predicate semantics and the linguistic realization of each type of rhetorical proposition (the content of an utterance, abstractly marked to indicate its information content such as attribution, evidence, or cause). This tested the case semantics, relational grammar, phrase structure grammar, dictionary, and morphological component. For rhetorical predicates such as logical-definition, attribution and evidence, testing of the predicate semantics was performed in each domain, automatically and exhaustively where possible. For example, predicates semantics that take a single entity as an argument and return propositional content from the underlying knowledge base (e.g., the predicate semantics of logical-definition, synonymic-definition, antonymic-definition, attribution, constituency, and classification) can be tested automatically by recursing down the generalization hierarchy of the application, instantiating and realizing rhetorical propositions during descent. Unfortunately, not all rhetorical predicates can be tested in this manner, because the predicate semantics for some rhetorical predicates do not always return the same information given the same entity. For example, the illustration rhetorical predicate randomly selects an instance of a given class and so while it can be tested recursively as above, the testing is not exhaustive because for any given instantiation of the predicate a different example may be chosen. Similarly, rhetorical predicates which take multiple arguments (e.g., comparison, inference) require more sophisticated testing algorithms. Even more complex, rhetorical predicates that are based on the context of a session in the underlying application (e.g., cause or evidence predicates in a diagnosis or consultation system) can be tested in general, but their content may vary with each session and so it is much more difficult to validate their correctness. In these cases representative rather than exhaustive testing is all that is claimed.

One other form of testing was performed at the utterance level to test the promise of relational grammar. Texts were generated in both English and Italian from a common application, NEUROPSYCHOLOGIST, after replacing the language-dependent syntactic, lexical, and morphological components. The English and Italian output for a given discourse goal were examined by a native Italian who found them to satisfy the discourse goal and to be reasonable translations.

### 9.2.2 Tests of Text Types

In addition to testing single utterances, the range of multisentential text types was also tested by producing textual output for the entire range of the discourse goals that the plan operators could achieve (e.g., both high-level goals such as get the hearer to know about an entity as well as lower-level goals such as get the hearer to know the superclass of a given entity). All possible generic forms of text were produced in each domain application. While some types of text (e.g., description and comparison) were tested in multiple domains, others were examined in only one (e.g., narration of LACE events).

Since multiple plan operators can achieve the same discourse goal, a text *replanner* was constructed which would attempt alternative strategies when the discourse controller signalled that the previous attempt had failed (e.g., the user rejected it). This reactive planning (Moore, 1989) brings up yet another form of testing, which is how the planner reacts in given contexts, in particular based on the content of the discourse model (i.e., previous queries and responses) and the user model (what the system believes the user believes). While it is much more difficult (if not impossible) to perform exhaustive testing of responses based on context, an attempt was made to consider a range of contexts. Because the text plans underlying the surface form capture both a decomposition of communicative acts and a decomposition of effects, when these plans are executed, a model of the user's knowledge, beliefs, and desires is updated if linguistic realization succeeds and the user accepts the response. TEXPLAN's model of the user's knowledge, beliefs, and desires can then be exploited to plan subsequent texts. If the text plan fails during execution (i.e., it cannot be realized or it is rejected by the user), then alternative communicative acts can be attempted. For example, the following dialogue was produced by TEXPLAN in the context of the mission planning system, KRS (Dawson et al., 1987) (where U is the user and S is the system).

U1: *What is a KC-135?*
S1: `A KC-135 is a tanker for air-refueling and a transport vehicle for cargo transport.`
U2: *What?*
S2: `A KC-135 is a stratotanker.`
U3: *ok.*

U1 is simulated by posting the corresponding goal to the planner (e.g., get the hearer to know about a `KC-135`). Feedback from the user is simulated by allowing the user to choose from a "reaction menu" which contains canned responses such as "accept", "reject", or "elaborate" (No query disambiguation was performed as in Moore (1989)). Because TEXPLAN's first response, S1 (a logical definition), fails, it attempts an alternative strategy in S2 (a synonymic definition). Because this succeeds (the user says "ok"), the user model is updated to indicate all the effects of the previous utterance (i.e., the user knows about a `KC-135` and they know its nick-name is `stratotanker`).

Similarly, consider the following dialogue, also from KRS, which illustrates TEXPLAN's use of analogy.

U1: *What's an F-15?*
S1: `An F-15 is an American fighter for air-to-air-interdiction. It has several standard configuration loads (scl-d1 and scl-d2), a speed of 350 miles per hour, a communications radius of 2,220,000, an empty weight of 12,500, a maximum weight of 24,500, a maximum fuel capacity of 10,600, a turn time of 3,600, a ferrying range of 4,630,000, and a ferry package of 3,000. There are five classes of F-15s: the F-15A, the F-15B, the F-15C, the F-15D, and the F-15E.`
U2: *OK. What's a MiG-29?*
S2: `A MiG-29 is a Soviet fighter for air-to-air-interdiction. It has a speed of 600 miles per hour, and a communications radius of 1,200,000.`
U3: *Huh?*
S3: `A MiG-29 is a Fulcrum.`
U4: *Huh?*
S4: `A MiG-29 is like an F-15 except that it is a Soviet fighter, has a faster speed, and has a smaller communications radius.`
U5: *OK.*

After the system defines an F-15 (S1), the user knows about it, knows its purpose, knows its attributes, and knows its superclass. Therefore, when the extended definition (S2) and synonymic definition (S3) of a MiG-29 fail, TEXPLAN can use an analogy because the MiG-29 is similar to the F-15 and the user model indicates that the user knows about the F-15. Reactive planning was used to test the range of communicative acts.

Instead of using the reactive approach, another method of testing is to simply assume given contexts, that is to alter the discourse or user model by hand. For example, the distinction between terse responses and lengthy ones is captured in the HASTE predicate which indicates if an agent is rushed. This is used, for example, to test TEXPLAN's production of short versus extended descriptions given the same query. This extremely simple representation of terseness could be extended to take note of other factors such as the amount of content, the agent's interest or attention span, and so on. This problem is analogous to the problem of controlling growth points in RST implementations, and Hovy (1988c, p. 15) suggests several criteria for the inclusion and exclusion of content. Just as the HASTE predicate can be preset to guide the selection of short and lengthy responses, different configurations of the user model or discourse model can be assumed to test alternative responses.

### 9.2.3 Multiple Domain Tests

A final form of testing was multi-domain validation. While several linguistic realization components (e.g., MUMBLE and PENMAN) have been ported to various applications in multiple domains over the past decade, most text planners have been tested in single domains (e.g., naval databases (McKeown, 1982), financial investment advising (McCoy, 1985ab), complex physical objects (e.g., a telephone) (Paris, 1987ab), naval fleet management (Hovy, 1988a), and program enhancement advising (Moore, 1989)). Single domain testing was sufficient for previous text planners because they investigated a subset of text types, although these systems thus have not empirically validated any claims of domain-independence. SPOKESMAN (Meteer, 1989) did produce text from several domains although the research did not focus on rhetorical structure or communicative acts.

In contrast, TEXPLAN produced output from a variety of application systems and domains including both autonomous and fairly sophisticated knowledge based systems (e.g., LACE, KRS, NEUROPSYCHOLOGIST) as well as hand-developed knowledge bases encoding generic information about complex physical objects (e.g., the heart), events and situations (e.g., the pilot story), or arguments (e.g., the Socratean syllogism). At the sentence level, many of the same types of rhetorical predicates were tested in multiple domains. For example, if TEXPLAN is given the goal of getting the hearer to know about something, logical definition is an effective domain-independent technique as illustrated by the examples below from mission planning, photography, and vertebrate knowledge bases, respectively.

USER: *What is an A-10?*
TEXPLAN: An A-10 is a fighter for air-to-ground interdiction.

USER: *What is an optical lens?*
TEXPLAN: An optical lens is a component for focusing located in a camera.

USER: *What is a canary?*

```
TEXPLAN:  A canary is a yellow bird with a Canary Islands origin, that
          sings, and is domesticated.
```

Because of its modularity, TEXPLAN was able to produce multisentential text from these different application domains by redefining only the predicate semantics and dictionary (e.g., domain-specific verbs and nominals), although new domains sometimes required grammatical extensions (e.g., the addition of new adverbials or prepositional phrases) or new rhetorical predicates (e.g., the addition of cause, motivation, and evidence predicates).

## 9.3 Evaluation

In addition to testing the computational implementation of the text generator, it is necessary to evaluate the results, both theoretical and practical. Theoretically, the dissertation provides a unified view of rhetorical, illocutionary, and locutionary acts which are formalized in a common plan language. Thus it can be seen as an extension of theoretical work which views language as purposeful behavior (Austin, 1962; Searle, 1969) and of computational implementations of speech acts (Cohen, 1978; Allen, 1979; Appelt, 1982). Guided by previous text linguistics research (Grimes, 1975; van Dijk, 1977; van Dijk and Kintsch, 1983; de Beaugrande, 1984; Mann and Thompson, 1987), psychological research (Meyer, 1975) and computational linguistic research (McKeown, 1982; McCoy, 1985ab; Paris, 1987ab; Hovy, 1988a; Moore, 1989), this thesis uses a tripartite theory of communicative acts to identify, characterize, and formalize four text types as plans: description, narration, exposition, and argument. These plans capture a broader range of text types than previous accounts and operate over a correspondingly wider range of rhetorical predicates (e.g., logical-definition, synonymic-definition, evidence, motivation, etc.). Finally, this work explores how three types of focal constraints (discourse, temporal, and spatial) can guide the order and realization of propositional content (e.g., the realization of adverbial and prepositional phrases).

Practical evaluation of natural language processors, and in particular natural language generators, is in a nascent stage (Palmer et al., 1989). Researchers have identified two broad evaluation methods: *black box* and *glass box*. The former examines input/output pairs whereas the latter considers the internal workings of system components. Of course individual components in turn can be evaluated using the black box technique. Unfortunately, black box evaluation requires an agreed upon corpora of input/output pairs. While the speech processing community has such qualitative and quantitative measures, there is neither an agreed upon set of evaluation criteria (i.e., measurement sticks) nor an agreed upon evaluation methodology for natural language processing systems. This is in part because these systems address a wide variety of tasks (e.g., database query, machine translation, text understanding or generation), employ diverse linguistic formalisms, and aim at different goals. While there exists no accepted set of evaluation criteria, Webber's (1988) "discourse canon" is one range of phenomena at the discourse level that requires testing. Regarding these discourse phenomena, TEXPLAN is able to produce intersentential and intrasentential pronominalization based on a discourse focus model (Sidner, 1979; McKeown, 1982), as well as adverbials and prepositions that are sensitive to temporal and spatial context. However, many discourse phenomena are not addressed by nor were the aim of TEXPLAN including one-anaphora, verb phrase anaphora, or discourse deixis (Webber, 1988b). The remainder of this section considers a number of criteria which are used to evaluate TEXPLAN from both a black box and glass box perspective, accepting that neither method can be applied very stringently and that the judgement on the quality of TEXPLAN's output for form, content, and contextual propriety are necessarily informal.

### 9.3.1 Black Box Evaluation

From a *black box* or global input/output perspective, TEXPLAN produces a broader range of text types than previous systems including several forms of description, narration, exposition, and argument (see Figure 9.1). Hundreds of descriptive texts, dozens of narrative and expository texts, and multiple argument texts were actually planned and linguistically realized. Description was the most investigated form of prose. Hundreds of definitions, comparisons, and extended descriptions were produced in several domains. For example, TEXPLAN can compare entities as in the following comparison of fish and birds in a vertebrate domain:

```
Fish are vertebrates that swim, have fins, have gills, are aquatic, eat
vegetation and fish, have scales, and are cold-blooded.  Birds, on the
other hand, are vertebrates that fly, have wings, are terrestrial, eat
seeds, have feathers, and are warm-blooded.  Fish and birds have the same
superclass, different locomotion, different propellors, different
environments, different diets, different covering, and different blood-
temperatures.  Therefore, they are different entities.
```

While the generation of paragraph-length descriptions and comparisons is not new (McKeown, 1982), TEXPLAN has multiple strategies that achieve a given discourse goal (in the case of comparisons TEXPLAN has three distinct strategies, detailed in Chapter 4). Furthermore, descriptive strategies were mixed with other types of text, as illustrated in the cookie instructions and heart exposition in Chapter 6. In addition to producing these paragraph-length texts, the system addressed the organization and presentation of longer stretches of prose. For example, in the LACE report generation of Chapter 5, the narrative plan operators reasoned about topic, time, and space to structure and order propositions. This resulted in many multi-paragraph (in some instances multi-page) texts. These longer stretches of prose were made possible only by taking advantage of the tripartite model of focus (discourse, temporal, and spatial) and corresponding sequencing operators. The comprehensibility of these longer texts was improved not only by focus but also by exploiting the structure in the text plan which was used to guide orthographic layout. While the sequencing strategies used in narration appear quite effective at organizing large amounts of event information, other strategies seem to work well only for shorter texts. For example the locational instruction plan operators of Chapter 5 produced dozens of texts like the following instruction on how to get from Mannheim to Heidelberg:

```
From Mannheim take Route 38 Southeast for four kilometers to the
intersection of Route 38 and Autobahn A5.  From there take Autobahn A5
Southeast for seven kilometers to Heidelberg.  Heidelberg is located in
block 32umv7070 at 49.39° latitude and 6.68° longitude, 4 kilometers
Northwest of Dossenheim, six kilometers Northwest of Edingen, and five
kilometers Southwest of Eppelheim.
```

While this strategy was very effective in the short range (e.g., 5-10 segments), it was less effective over longer stretches where techniques such as abstraction and reminding seem to be required.

Figure 9.3 compares the rhetorical range of TEXPLAN to previous rhetorically-based text planners including McKeown's (1982) TEXT, Paris's (1987ab) TAILOR, Hovy's (1988a) "structurer" and Moore's (1989) "reactive planner". "+" means the system produces the text class/subclass and "0" means it does not. Of course, finer distinctions can be made. For example while TEXT, TEXPLAN, and Moore's (1989) system could divide an entity in two manners (i.e., classification and constituency), TAILOR considered only constituency and Hovy (1988a) considered neither. Furthermore, TEXPLAN allows combinations of classification and constituency in extended descriptions (see Chapter 4). Similarly, while other systems have at most one method of definition, TEXPLAN employs three (logical, synonymic, and antonymic). While TEXT has one method of comparison, TEXPLAN has three (see Chapter 4). The distinctions at higher levels of organization become more difficult because of differences in data structures (ATNs versus plan operators) as well as alternatives and flexibility in choice. Therefore, Figure 9.3 simply uses a "+" to indicate if the text type was produced in general and "0" if it was not. The only claim that is made is that TEXPLAN has a broader rhetorical range than other systems. However, while these other text planners produce output in one domain, TEXPLAN is also able to generate several types of text from multiple application systems. For example, it produces description and locational directions from a cartographic system, and description and narrative reports from a simulation system. Also, in the neuropsychological diagnosis system it is able to produce Italian as well as English output. It has produced paragraph-length descriptions and comparisons from all of these.

| Criteria | McKeown(82) | Paris(87ab) | Hovy(88a) | Moore(89) | TEXPLAN |
|---|---|---|---|---|---|
| Text Types | | | | | |
| Description | + | + | + | + | + |
| definition (logical, ...) | + | + | 0 | + | + |
| characterization | + | + | + | + | + |
| division (subparts/subtypes) | + | + | 0 | + | + |
| comparison | + | 0 | 0 | + | + |
| analogy | + | 0 | 0 | + | + |
| Narration(report, story, bio) | 0 | 0 | + | 0 | + |
| topical sequence | 0 | 0 | 0 | 0 | + |
| temporal sequence | 0 | 0 | + | 0 | + |
| causal sequence | 0 | 0 | + | 0 | + |
| spatial sequence | 0 | 0 | 0 | 0 | + |
| Exposition | 0 | + | 0 | 0 | + |
| plans | 0 | 0 | 0 | 0 | + |
| processes | 0 | + | 0 | 0 | + |
| propositions | 0 | 0 | 0 | 0 | + |
| Argument | 0 | 0 | 0 | + | + |
| deductive | 0 | 0 | 0 | 0 | + |
| inductive | 0 | 0 | 0 | + | + |
| persuasive | 0 | 0 | 0 | + | + |

Figure 9.3 Black Box Comparison with other Multisentential Text Planners

Comparison of another black box metric, speed, suggests that TEXPLAN is approximately equivalent in efficiency to recent text planning and linguistic realization components, although efficiency is difficult to compare because of varying hardware and lack of detail on measurement techniques. TEXPLAN accomplishes planning and linguistic realization in a few seconds per utterance on a Symbolics 3600 running Genera 7.2. Computational complexity rather than speed is a more appropriate metric although this too is difficult to compute because of differing data structures and algorithms both for text planning and linguistic realization. However, the complexity of TEXPLAN's text planner is roughly equivalent to that of Hovy (1988a) and Moore (1989), whose approaches are based on hierarchical planning where complexity can be measured by the number of alternatives the text planner must consider when expanding a subgoal and the complexity of the constraints on this choice. McKeown's (1982) and Paris's (1987ab) ATN-based strategies appear to have fewer decisions to make at choice points (i.e., deciding which arc to pursue) than hierarchical planners which must consider the entire plan library when expanding a subgoal, and so are computationally less complex but correspondingly less flexible (ATN-based strategies are sometimes viewed as compiled planners). In summary, while comparison of computational complexity may be inconclusive, it is clear that TEXPLAN is able to produce a wide range of rhetorically varied prose when examined from the black box perspective.

### 9.3.2 Glass Box Evaluation

In addition to this black box evaluation, TEXPLAN can be examined from the *glass box* perspective. Figure 9.4 compares the inner components of TEXPLAN and several recent systems using a number of criteria. Some of these are quantitative metrics, including the number of rhetorical predicates and plan operators employed. For feature comparisons, "+" means the system has it and "0" means it does not. The first two systems are rhetorical predicate/schema based whereas the latter two formalize RST as plan operators. TEXPLAN can be viewed as a plan-based approach employing rhetorical predicates as the building blocks of text. The comparison of prose constituents (i.e., rhetorical predicates) between systems was based on the content and not the name of those constituents (e.g., McKeown's and Paris's use of the "identification" predicate is analogous to the "logical-definition" predicate in TEXPLAN, although the differentia in the latter are computed). The chart is only intended as a suggestive comparison as these various system had very different goals. For example, McKeown (1982) was investigating responses to general queries about database content and Paris (1987ab) was examining tailoring responses to a user's level of expertise. The principal differences are that TEXPLAN has the larger number of rhetorical predicates and plan operators that are required for a broader range of text types, and that TEXPLAN uses both a discourse and a user model (following Moore (1989)).

When TEXPLAN is examined from the glass box perspective, several properties become apparent. First, the system is very modular which lends portability, manifest in the generation of text from multiple domains using many common system components. In particular, following McKeown (1982), domain-independent rhetorical predicates are employed to capture generic propositional elements of text. The system is both maintainable and extensible. The system can and has been incrementally augmented either by adding new plan operators to the library of the generic hierarchical (re)planner, or by adding new types of rhetorical predicates to extend the system to handle new classes of propositional content. With respect to the data structures employed, the grammar is declared in a phrase structure grammar although it would be preferable to have an equally declarative semantics (e.g., Montague, 1974). Unlike previous text planners, the plan operator language represents multiple effects and distinguishes between necessary and desirable preconditions which help to guide planning in operator-specific ways. Following Moore (1989), general heuristics also guide plan selection (e.g., prefer plans that introduce fewer new entities in the discourse). One improvement in TEXPLAN would be to replace the linguistic realization component with a more efficient and incremental sentence generator. The linguistic realization component has a narrower coverage than previous work and is less sophisticated than, for example, MUMBLE's incremental and indelible generation although TEXPLAN has more levels of linguistic representation (i.e., intention and rhetoric, semantics, grammatical relations, syntax, morphology, and orthography). Since planning and realization can occur interleaved, the perceived speed of the system is improved by linguistically realizing sentences as they are planned.

| Criteria | McKeown(82) | Paris(87ab) | Hovy(88a) | Moore(89) | TEXPLAN |
|---|---|---|---|---|---|
| Rhetorical Predicates[1] | 10 (16) | 4 (7) | 0 | 0 | 21 |
| Plan Operators | 0 | 0 | ~ 8 | 43 | >70 |
| rhetorical | 0 | 0 | ~ 8 | 40 | >60 |
| illocutionary | 0 | 0 | 0 | 3[2] | 4 |
| locutionary | 0 | 0 | 0 | 0 | 6 |
| User Model | 0 | + | + | + | + |
| knowledge | 0 | + | + | + | + |
| beliefs | 0 | 0 | + | + | + |
| desires | 0 | 0 | 0 | + | + |
| expert/novice distinction | 0 | + | 0 | 0 | 0 |
| Discourse Model | 0 | 0 | 0 | + | + |
| queries | 0 | 0 | 0 | + | + |
| responses | 0 | 0 | 0 | + | + |
| Focus Models | + | 0 | + | 0 | + |
| Discourse Focus | + | 0 | +[3] | 0[4] | + |
| Temporal Focus | 0 | 0 | 0 | 0 | + |
| Spatial Focus | 0 | 0 | 0 | 0 | + |

Figure 9.4 Glass-Box Comparison with Other Multisentential Text Planners

Unlike previous planners that attempt to achieve affects on the user's knowledge and beliefs (Hovy, 1988a; Moore, 1989), TEXPLAN updates its model of the user after each interaction depending upon the user's reaction to its utterances. TEXPLAN's plan operators distinguish between the communicative function of a text and the effects of that text at all levels of communication (rhetoric, illocution, and locution). Thus, the system can produce both a communicative action decomposition and a related effect decomposition. These structures can then contribute to a discourse model (which includes the communicative act decomposition planned to achieve a given communicative goal), and to a model of effects on the user's knowledge, belief, and desires. Finally, the attentional model distinguishes between discourse, temporal, and spatial focus. In summary, the communicative plans attempt to characterize a broader range of text, they distinguish rhetorical, illocutionary, and locutionary acts, and their order and realization as English is constrained by three types of focus: discourse, temporal, and spatial.

[1]Numbers indicate those used in the implementation. Numbers in parentheses indicate those used for text analysis.
[2]Moore(1989) claims four speech acts (INFORM, RECOMMEND, COMMAND, and ASK) but provides plan operators only for the first three. The plan operators provided do not formalize the effects of these speech acts on the cognitive state of the addressee (i.e., their knowledge, beliefs, or desires).
[3]It actually was not until Hovy and McCoy (1989) that discourse Focus Trees (McCoy and Cheng, 1988) were used to constrain planning RST.
[4]Moore (1989) claims the top-level node in her text plan is the global context, although this is not really a discourse focus in the sense of Sidner (1979).

## 9.4 Limitations

There are a number of limitations with the TEXPLAN and several research areas which require further investigation. These include issues concerning plan-based approaches to communication, the relationship of planning and realization, and linguistic realization itself.

One unresolved issue concerns the nature of planning and communicative acts. Pollack (1986), Grosz and Sidner (1989) and others have outlined the weaknesses of current planning technology as a formal representation for language. These criticisms focus on the assumptions made by planning models such as (1) the action taxonomy must consist of mutually exclusive actions and (2) the action decomposition hierarchy must be complete. Grosz and Sidner (1989) note the difficulty of representing collaborative behavior in such formalisms. These criticisms, while problematic for plan recognition, are less of an issue for explanation presentation as the task is not to produce a plan that matches some observed agent behavior but rather to produce some plan that achieves some given high-level goal. However, while hierarchical planning remains an extremely valuable tool for text planning research, more flexible methods of text planning are still needed.

A different issue concerns the actual structure of the plans themselves. While STRIPs-like[5] (Fikes and Nilsson, 1971) planners typically represent the preconditions, body, and effects of an action, what is not explicitly represented are order or enablement relations among subacts, or the relation of the preconditions and effects of a plan operator to its subacts. These problems have, in part, been addressed by work in meta-planning (Wilensky, 1983). Another problem pointed out by Allen (1984) is that these formalisms have difficulty representing simultaneous action or persistent goals (e.g., a desire to stay alive). Persistent action is also difficult to capture (e.g., breathing). A related open research issue concerns the formalization of the semantics of intention and belief (e.g., Moore, 1980; Cohen and Levesque, 1985).

Another issue concerns the problem of failed plans. If a plan fails in TEXPLAN, the system currently recovers by attempting alternative plans that achieve the top-level goal of the text. Moore (1989) considers how a query analyzer can use deictic input and context to attempt to determine which particular clause failed in a previously generated text. Another issue is the relationship between planning from first principles and, at the other extreme, canned plans (e.g., text schemata ala McKeown (1982)). Text planners need mechanisms to choose between planning from scratch, plan modification (i.e., tailoring partially canned plans), or using totally pre-stored strategies to achieve a discourse goal. Because planning is expensive, this brings up the related issue of partial replanning. This is the notion behind Litman and Allen's (1987) and Moore's (1989) examination of correction and clarification subdialogues. In general, mechanisms need to be constructed to perform more sophisticated plan repair. This involves a final notion, that of execution monitoring which involves questions such as: At what level should the generator monitor its utterances (paragraph, sentence, clause, lexeme)? How often should monitoring occur? What should

[5]STRIPS had preconditions and effects whereas NOAH (Sacerdoti, 1977) added bodies (i.e., subacts) to plan operators.

the generator listen for? General plan inference mechanisms will probably be too costly but at the other extreme, canned reaction may not always be appropriate. On a different level there is the issue of combining plans, ad hoc, to achieve novel discourse goals.

Related to planning is the nature of control, in particular between text planning and linguistic realization. Hovy (1987, 1988b) considered two types of planning: prescriptive planning (top-down) and restrictive planning (bottom-up). While the former precedes realization and satisfies goals that are removed from the list of goals to be achieved, that latter is interwoven with realization and consists of choice from several alternative realizations on the basis of (potentially competing) goals that persist even after realization (e.g., the attitude of the speaker toward the content or toward the addressee). Interleaved planning and execution has been an issue in planning at least since McDermott (1978). TEXPLAN considers interleaved planning and linguistic realization whereby failure of linguistic realization signals to the text planner to backtrack or negative user feedback tells the system to replan. However, this needs to be extended to the clause level and made more flexible by, for example, having the planner reason about whether to attempt to repair the current plan or abandon it and replan from scratch.

In addition to planning and the nature of planning and realization, the linguistic realization component, not the principal focus of this dissertation, requires improvement. This includes not only the extension of syntactic grammar to handle more complex constructs but also the investigation of incremental generation, planning which penetrates sentence boundaries, and "ill-formed" but more natural output. This will require more sophisticated control mechanisms that operate at finer levels of detail, not just at the utterance level.

## 9.5 Future Directions

In addition to planning and realization, there are several open issues which require further investigation. These center on text types, general constraints on the generation process (e.g., models of attention), and extending explanation strategies to incorporate dialogue and multi-media communication.

### 9.5.1 Text Types

Generic classes of text, both their nature and particular types, require further research. The text types presented in this dissertation—description, narration, exposition, and argument—convey different propositional content (e.g., entities and relations versus events and states), have particular intended effects on the addressee's knowledge, beliefs, and desires, and are compositional (e.g., narration can invoke description). The communicative acts which compose these text types are hierarchical (e.g., description -> definition or detail or division) bottoming out in rhetorical predicates (e.g., division -> classification or constituency) which helps constrain the search space during planning. Some of the text types such as story narration and plan exposition that were tested in an ad-hoc manner (using hand-encoded knowledge bases) require further investigation and refinement. For example, Mellish and Evans (1989) and Dale (1989,

1990) produce expositions of complex plans from an actual planner (as opposed to hand-encoded plans) and find that abstracting unnecessary details from the underlying plans to be a complex task. Thus it is unclear how some of the less tested and more straightforward explanation strategies would function on complex application systems.

In addition to investigating the suitability of text types to more complex explanation tasks, the building blocks of text types, the twenty-one rhetorical predicates detailed in Table 8.1, may require extension to characterize a broader range of text. And if more communicative acts are added to the system, how will their constraints guide the selection among competing plan operators? That is if there are ten ways to persuade someone to perform an action, how and why do we select one versus the other? Some relevant issues include the content being conveyed, the content of the user model (e.g., relate things to what the user knows), and the speaker's own biases (cf. Hovy, 1987).

### 9.5.2 Lengthy Text

A more fundamental question concerns how this approach handles longer stretches of text. The lengthiest texts generated by TEXPLAN were multi-paragraph LACE reports, which were coherent because the system conveyed the text structure via orthographic layout and exploited the additional constraints offered by the tripartite focus model in realizing the prose. To produce even longer prose, for example a paper or technical report, will require additional mechanisms to guide the reader and indicate structure. This goes beyond layout conventions (e.g., headings and subheadings) and involves more fundamental issues such as the need for recapitulation, reminding, backward pointing (anaphora), and forward pointing (cataphora), in order to overcome the attentional limitations of the reader. Another issue raised by longer texts is how to control recursion, for example determining when to stop subdividing an entity (i.e., when using constituency or classification recursively).

### 9.5.3 User Models

Another fundamental issue concerns the cognitive effects of text types on the addressee. Just as single utterances can have multiple effects simultaneously, texts can affect the knowledge, beliefs, and desires of the reader simultaneously. For example, while description has been defined as affecting the addressee's knowledge of entities and relations, it can equally affect their beliefs and goals. The cheery description of a tropical island in a travel brochure not only conveys an impression of the place but also may convince the reader they want to go there. Several researchers have investigated guiding text generation using models of the user's expertise (Paris, 1987ab) which can guide rhetorical form, the user's point of view (McCoy, 1985ab), rhetorical goals (e.g., to show superiority, to impress, to hasten) (Hovy's 1987), and feedback from the user (Moore, 1988). Related to this is the need for richer models of the user (cf. Kass and Finin, 1988; Kobsa and Wahlster, 1989), especially with respect to the psychological state of the user (e.g., fear, happiness, suspense) and its relation to communicative plans. An interesting investigation would concern the tailoring of the rhetorical structure of responses to particular users (Paris, 1987ab; Haimowitz, 1989), using a broad range of text types guided by a model of the user. More text analysis needs to be performed to identify what constraints guide the mixing and matching of rhetorical strategies to accomplish discourse goals.

### 9.5.4 Constraints on Generation

In addition to issues concerning classes of text, the constraints on the generation process as a whole remains an important research area. In TEXPLAN generation is constrained by a discourse model, user model, focal model, communicative strategies, and information about the domain (e.g., entities, attributes, relations). In some contexts, TEXPLAN uses not only the discourse goal but also the amount and type of knowledge in the underlying application to determine rhetorical ordering. For example, the generator examines the amount and type of semantic connections (e.g., is-a versus instance versus a-part-of links) in the knowledge base to decide between describing the structure of an object (constituency) as opposed to speaking about the object's subclasses or subtypes (classification). In addition, other types of information (e.g., focus, discourse context, and the model of the user) constrain the selection, order, and realization of content.

Global (Grosz, 1978) and local focus (Sidner, 1979) are important constraints on generation. McKeown (1982) used both schematic discourse patterns and discourse focus to constrain generation. Attention is explicit in systems such as TEXT (McKeown, 1982) and TEXPLAN, but not addressed by RST and only implicit in plan operators in Hovy's (1988a) and Moore's (1989) systems. On the other hand, RST explicitly addresses speaker intention, which is not addressed by TEXT but is explicitly represented in TEXPLAN. McCoy and Cheng (1991) advocate representing discourse focus in trees (as opposed to Grosz and Sidner's (1986) use of a stack of focus spaces) although they do not consider the relation of attention, intention, and the structure of the discourse as do Grosz and Sidner. Hovy and McCoy (1989) employ focus trees to constrain Hovy's (1988a) RST-based text planner.

In contrast to past work, this dissertation distinguishes three types of focus: discourse, temporal, and spatial. However, a more complete investigation of the three focus classes is required, in particular temporal focus shift rules (e.g., shift forward, backward, laterally in time) and spatial focus shift rules (e.g., three dimensional spatial shifts). Their effect on content selection, structure, and order as well as on surface form needs to be investigated further. While the representation and use of discourse focus and its shift rules to guide both selection and realization of content has received much attention, temporal focus and temporal focus shifts (Webber, 1988a; Nakhimovsky; 1988) have received less attention. This dissertation investigates the effect of temporal focus on realization in temporal focus maintenance or forward progression; however, lateral or backward shifts in time require further investigation. Also, the notion of spatial focus and spatial focus shift rules have only been computationally investigated in generating two-dimensional route-plans. Three dimensional spatial focus representation and shift rules require formulation and testing. In TEXPLAN discourse focus affects pronominalization and grammatical structure (e.g., voice selection), temporal focus affects tense choice, temporal adverbials, and temporal prepositions, and spatial focus guides the realization of locative adverbials and prepositional phrases. So another area for work is the investigation of the constraints on the generation of other kinds of temporal and locative phrase and of other types of adverbials (e.g., manner adverbials, rate adverbials, and so on).

Other constraints which guide content selection and order include general sequence strategies (e.g., temporal, spatial, general to specific, increasing importance, complexity, and alphabetical order), common illocutionary orderings (e.g., McCoy's DENY-CONCEDE-OVERRIDE clarification strategy), and genre-particular characteristics (e.g., recipes follow temporal sequence whereas scene descriptions follow spatial ordering). The underlying model of the domain or the task structure (Paris, 1987ab; Mellish and Evans, 1989) and the qualities of the propositional content (e.g., reliability, quantity, type) can also help guide the choice, structuring, and sequencing of content. These and other constraints are important areas for further research.

### 9.5.5 Dialogue

In addition to issues concerning text types and constraints on the generation process, another important area concerns the extension of explanation capabilities to function in the context of dialogue and to deal with problems like miscommunication. One area concerns the integration of text generation systems with a natural language interpreters, especially regarding bidirectional (Kay, 1980; Appelt, 1987; Jacobs, 1988; Shieber, 1988; Levine and Fedder, 1989; Levine, 1989, 1990, forthcoming). A related area is the investigation of communicative acts underlying not texts but rather dialogues (Cawsey, 1989, 1990; Wolz, 1990), which includes issues concerning subdialogues (Litman and Allen, 1987), follow-up questions (Moore, 1989), interruptions, and miscommunication recovery (McCoy, 1985ab). The integration of a communicative act based text planner like TEXPLAN and a communicative act based dialogue planner as in (Cawsey, 1989) could provide the foundation for a more robust cooperative dialogue system.

### 9.5.6 Multi-Media Explanation

In addition to dealing with dialogue, more natural explanation presentation strategies should reason about the most effective mode in which to present information. A natural extension of TEXPLAN's communicative plans would involve the generation of multi-media explanations. Some research has been done in presentation planning although the semantics of graphics remains an open issue. While the nature of graphical primitives is a complex issue, rhetorical predicates seem to be a natural level of abstraction for some graphical phenomena, allowing them to be handled in the same way as language ones. For example, several rhetorical predicates have graphical correlates: the attributive and comparison/contrast predicates can be represented in tabular format (e.g., attribute-value pairs), the constituency/classification predicates can be displayed as trees, and the illustration predicate correlates to providing a picture or instance of some unknown object (e.g., showing a picture of a zebra.) Similarly, many of TEXPLAN's communicative acts have correlates in other modes of presentation. For example, at the rhetorical act level narration is equivalent to temporal/spatial/causal animation and locative instructions correlate to spatial animation as in a film clip of a car following a route. Graphical primitives and aggregates require careful characterization, classification, and formalization.

Some recent work in presentation planning includes investigations into representations for media-independent communicative goals (Feiner et al., 1989), related intent-based illustration systems (Feiner, 1985; Feiner et al., 1989), the coordination of multiple modalities (Neal, 1989; Feiner and McKeown, 1990), terminology/languages for expressing multiple modalities (Fehrle, 1988; Hovy and Arens, 1990), and multimedia presentation rules (Wahlster et al., 1989; Burger, 1989). Other work is investigating the syntax and semantics of graphical primitives (Geller, 1988). Research in psychoperception can help guide this work, for example experiments examining the meaningfulness of verbal and pictorial elements (Guastello and Traut, 1989). While graphics have received a great deal of attention, the tactile and auditory senses also offer rich modes of communication and are related to communicative acts. For example, there are analogs between mediums such as textual, graphical and auditory warnings (exclaiming, flashing and beeping), graphical and auditory icons (e.g., using sirens to indicate danger), and graphical and auditory motion (e.g., using the perception of Doppler effects to indicate motion). The relation of communicative acts to text, graphics, and audition requires formalization in a media-independent representation language that a presentation planner reason about to achieve media-independent communicative goals. This remains an interesting avenue for future research.

# Appendix A

## Entity Differentia Formula

Logical definition is one of the most widely used rhetorical techniques. It consists of defining a term (species) by indicating its superclass (genus) and its distinguishing features (differentia). As discussed in Chapter 4, given an entity in a generalization hierarchy found in most knowledge based systems, it is relatively straightforward to retrieve an entity's superordinate(s) to serve as the genus (or geni) in the logical definition. Unfortunately, determining what belongs in the differentia portion of a logical definition is not as easy. This appendix develops an *entity differentia* algorithm that can automatically produce the differentia for a logical definition.

### Tversky's Object Similarity Formula

To identify the distinguishing features of an entity, we first need a numerical measure. We first consider Tversky's (1977) set-theoretic approach to *object* similarity which is based on common and unique features of objects. According to Tversky, if A and B are the feature sets of two objects, a and b, within some domain of objects $\Delta = \{a,b,c,...\}$, then we can use the ordinary Boolean set relations to define a similarity metric, s(a,b) where the similarity of two objects, a and b, "increases with addition of common features and/or deletion of distinctive features." This leads to a *contrast model* where $s(a,b) = \theta f(A \cap B) - \alpha f(A-B) - \beta f(B-A)$ for some $\theta,\alpha,\beta \geq 0$ where f reflects the relative salience or prominence of the various features as indicated by intensity, frequency, familiarity, good form, and/or informational content. $\theta$, $\alpha$, and $\beta$ indicate the relative importance of the three sets $A \cap B$, A-B, and B-A (i.e., common features versus features found only in A or B). Tversky went on to discuss a *ratio model* that normalizes the similarity of a and b, with the range [0,1]:

$$\frac{f(A \cap B)}{f(A \cap B) + \alpha f(A-B) + \beta f(B-A)} \quad \text{where } \alpha,\beta \geq 0$$

McCoy (1985ab) implemented some of Tversky's ideas and suggested that $\alpha$, and $\beta$ can be varied according to which objects are in focus (i.e., if a is in focus and b is not, then choose $\alpha > \beta$ so that "similarity is reduced more by features of object a that are not shared by object b than vice versa".) In order to encode f, the importance of each feature with respect to each other, McCoy implemented a model of perspective where a perspective filters which properties an object inherits from its superordinates. For example, when comparing a treasury bill with a money market certificate, the two objects can be viewed as savings instruments or company or organization issues. In the former case, attributes such as interest-rates and maturity-dates are highlighted. In the latter, characteristics such as issuing-company and purchase-place are highlighted. This context-sensitive metric of similarity illustrates how f, which determines the salience of different attributes, changes with context.

Tversky also discussed a set theoretic model of prototypicality and family resemblance. He defined the (degree of) prototypicality of object a with respect to some class, Λ, as

$$P(a,\Lambda) = P_n (\lambda \sum f(A \cap B) \ - \ \sum (f(A-B) + f(B-A))) \text{ with summation over all b in } \Lambda$$

Prototypicality, thus, is measured by examining which features of object a are shared or not shared with each element of Λ. $P_n$ reflects the effect of category size on prototypicality and λ determines the relative weights of common and unique features. Object a, therefore, is a *prototype* of class Λ if it maximizes P(a,Λ). This model could be used to automatically classify an object within a given generalization hierarchy. However, this indicates the prototypicality of an object with respect to a class whereas computing differentia for a logical definition requires a measure of the prototypicality of features with respect to an object (or more generally an entity).

The final notion Tversky discusses is *family resemblance*. If we let Λ be a subset of the domain objects, Δ, with cardinality n, then the category resemblance of Λ is mathematically:

$$R(\Lambda) = r_n (\lambda \sum f(A \cap B) \ - \ \sum (f(A-B) + f(B-A))) \text{ with summation over all a,b in } \Lambda$$

where $r_n$ reflects the effect of category size on category resemblance and the constant λ determines the relative weight of common versus unique features. R(Λ) accounts for the notion that family resemblance is highest for those categories which "have the most attributes common to members of the category and the least attributes shared with members of other categories" (Rosch et al., 1976, p. 435). Tversky notes that the maximization of category resemblance can explain the formation of categories. It is possible to implement the above model of family resemblance to automatically learn a classification hierarchy from a given set of objects. As with the prototypicality measure above, however, this does not aid in the selection of differentia.

## Decomposing the Similarity Metric

It is possible to take advantage of more specific knowledge, for example, to decompose notional features into attributes and their corresponding values (e.g., the attribute *color* as opposed to its specific value, *red*). It is also possible to take advantage of attribute-structure in a knowledge base (e.g., definitional versus non-definitional attributes[1] or physical versus intangible attributes) which may indicate feature saliency.

First, by decomposing features into attributes and values, we obtain a more precise definition of similarity. Instead of simply defining similarity as $s(a,b) = \theta f(A \cap B) - \alpha f(A-B) - \beta f(B-A)$, the formula below first contrasts the attributes of a and b and then, for all attributes common to a and b, it contrasts their attribute-value pairs:

$$s(a,b) = \theta f(A_a \cap B_a) - \alpha f(A_a - B_a) - \beta f(B_a - A_a)$$
$$+ \forall \in \{A_a \cap B_a\} \; (\theta_v f(A_{a,v} \cap B_{a,v}) - \alpha_v f(A_{a,v} - B_{a,v}) - \beta_v f(B_{a,v} - A_{a,v}))$$

where $A_a$, $A_{a,v}$ and $B_a$, $B_{a,v}$ indicate the sets of attributes and attribute-value pairs for objects a and b, respectively. $\theta$, $\alpha$, and $\beta$ indicate, respectively, the relative importance of attributes common to A and B, attributes found only in A, and attributes found only in B. Similarly, $\theta_v$, $\alpha_v$, and $\beta_v$ indicate, respectively, the relative importance of attribute values common to A and B, values found only in A, and values found only in B. Note that the magnitude of $\theta$, $\alpha$, and $\beta$ with respect to $\theta_v$, $\alpha_v$, and $\beta_v$ will indicate the relative importance of attributes versus their values during comparison. Like the original similarity model, $\theta,\alpha,\beta,\theta_v,\alpha_v,\beta_v \geq 0$, and f reflects the relative salience or prominence of the various characteristics.

The improved similarity metric can become arbitrarily complex where attribute and/or value structure is concerned. For instance, if the knowledge base distinguishes between definitional and non-definitional attributes, the attribute portion of the equation (i.e., $\theta_a f(A_a \cap B_a) - \alpha_a f(A_a - B_a) - \beta_a f(B_a - A_a)$ ) can be further decomposed to be sensitive to this distinction. Similarly, values may have structure. For example, the value of a quantitative attribute such as length (15 inches), volume (3 gallons), cost (2000 rubles), or speed (4 knots) can be decomposed into a measure and the units of measure. Furthermore, it may include a range or set of legal values. Similarly, the values of the "parts-of" attribute of an animate object may be structured according to the different functions that those physical components perform in the whole (e.g., sensors, manipulators, etc.).

There are many hazardous formal consequences of adopting intuitively plausible bases for measuring similarity. But as it is possible to argue for "knowledge-rich" measures, I shall work with these for demonstration purposes without making a strong claim for their objective propriety as opposed to illustrative utility.

---

[1]For example, a triangle must, by definition, consist of three sides but it can be any color.

## Discriminatory Power, Prototypicality, and Distinguishing Features

Despite this more elaborate similarity metric, we still require an algorithm that can compute the set of *distinguishing* features for a given entity. For example, consider the object o which we wish to distinguish from the context of a set of entities, E, of which o is a member. During referent identification, Dale (1989) first measures the *discriminatory power*, D, of all attribute-value pairs, <a,v>, of o with respect to E using the formula:

$$D(\langle a,v \rangle, E) = \frac{N-n}{N-1}$$

where N is the total number of entities in E and n is the number of entities in E of which <a,v> is true.[2] The range of D is the interval [0,1]. The maximum discriminatory value of 1 indicates that the attribute-value pair singles out the object from E. In contrast, a value of 0 indicates that the attribute-value pair is true of all entities in E and so it has no discriminatory power. That is, an attribute-value pair becomes less distinctive as it is more commonly held by other members of E.

We can refine Dale's formula by observing that attribute-value equality is not a binary function. While it may make sense to use a binary decision on most attributes and some values, quantitative values should be sensitive to the closeness on some relevant range. That is, if *equality* is measured on a scale of [0,1], where 0 indicates two values are opposites and 1 indicates that they are identical, then the closeness of fit of two values is defined as:[3]

$$\text{equality}(v_1, v_2) = 1 - \frac{|v_1 - v_2|}{\text{range of values}}$$

For example, if two freshwater fish have lengths 5" and 10", and the length of freshwater fish ranges from 1" to 72", then the closeness of the attributes would be:

$$1 - \frac{|5-10|}{72-1} = 1 - \frac{5}{71} = 1 - .07 = .93$$

Similarly, non quantitative values of attributes can be mapped onto a numerical scale. For example, the values of a color attribute could be arranged according to their location on a color-wheel and assigned ascending numbers. Using this equality metric, the discriminatory power of an attribute-value pair with respect to the set of entities, E, is defined as:

---

[2] Dale suggests excluding those objects from E which, in Grosz and Sidner's (1986) terms, are in closed focus spaces.

[3] The formula assumes a normal distribution over the range. It could be made sensitive to other distributions. Mike McHale (personal communication) suggests standardizing each value, $v_i$, so that $z_{v_i} = \frac{v_i - \mu}{\sigma}$ and then dividing the smaller value by the larger.

$$D(<a,v>, E) = \frac{N - \sum_{i=1}^{N} \text{equality}(<a,v>, <a,v> \text{ of } E_i)}{N-1}$$

Since attribute equality remains a binary function, we can use Dale's approach to define the discriminatory power of attribute <a> in the context of entity set E as:

$$D(<a>, E) = \frac{N-n}{N-1}$$

where n refers to the number of entities in E which have attribute <a>.

Once the discriminatory power of features are calculated, however, Dale does not indicate how features should be selected to identify o. One plausible approach is to order the object's characteristics according to their discriminatory power and take the first n elements of this set such that the resulting set of properties uniquely identifies the object in the context of E, the set of previously mentioned entities in a discourse.

When calculating the set of distinguishing features of an entity in a knowledge base, a similar approach can be taken. The situation is slightly more complex, however, because instead of E indicating those objects in the discourse history, it refers to all domain objects. Therefore the computational model presented here formalizes how the selection of distinctive features of some object, o, is influenced by the features (attributes and values) of the "relatives" of o in the generalization hierarchy. That is, an object's distinguishing features are dependent not only on the features of o's siblings (i.e., other objects within that same class as o)[4] but also on the features of o's parent(s), children, and other related objects such as cousins, uncles, and so on. The formal model of *entity differentia* suggested below is based on parent, child, and brotherhood relationships, although it could easily be extended to incorporate other classification relationships. And while this discussion primarily focuses on object differentia, the differentia algorithms for actions, events, processes, and states are analogous to that for objects where notions of classification, decomposition, and attributes/values are common to these entities.

Some important psycholinguistic evidence supports the notion that object identification is sensitive to the characteristics of the children and parents of an object. Collins and Quillian (1969) found that subjects took more time to verify statements classifying objects the farther away (in terms of semantic classes or *sememes*) the object was from its identified superclass. For example, a canary was verified slower (relatively) as an animal, faster as a bird, and fastest as a canary (see Figure A.0).

[4]Since multiple parents are possible (e.g., an apple is both a fruit and a computer), a slightly more sophisticated version of the formula distinguishes between true siblings (brothers and sisters) which have the exact same parents as object o, and types of siblings (such as step-sisters), which need only have one parent in common with o.

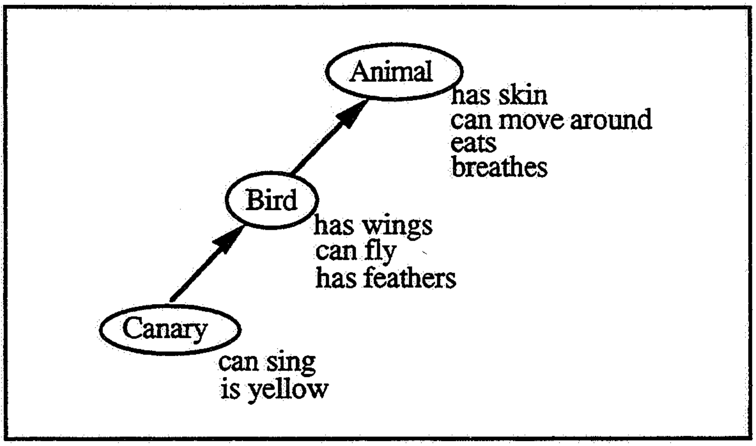



Figure A.0 Collins and Quillian's Canary Example.
Classes are circled; their associated attributes adjacent.
Directed arrows indicate class membership.

Similarly, Collins and Quillian found that when attributing characteristics to an object, the closer an attribute was to the object being characterized the faster it was verified. The subjects tested verified the statements "a canary has skin" slowly, "a canary has feathers" moderately fast, and "a canary can sing" rapidly. Later Rosch (1973) found that verification time increases as the *typicality* of an object decreases where, for example, an apple is considered more typical of the class fruit than an olive (recall Tversky's definition of prototypicality; this is its psycholinguistic motivation). Psycholinguistic studies like Collins and Quillian (1969) and Rosch (1973) suggest that entity descriptions are obtained by identifying the entity's most typical attributes and values with respect to the characteristics of the object's parents, children, and siblings in some conceptual hierarchy.

In particular, the distinguishing features of an object (or more generally of entities) should on the one hand be prototypical of o and its children, and on the other hand should differentiate o from its parent class(es) and siblings. This is indicated by the feature set $F = O \cap C - P - S$ in Figure A.1 where O is the set of features of the object (o), S is the intersection of the sets of features of o's siblings (s), P is the union of the sets of features of o's parents (p), and C the intersection of the sets of features of o's children (c). Using the subscripts $_a$ and $_{a,v}$ to distinguish between attribute and attribute-value pairs, $F = O \cap C - P - S$ can be refined to the sets $F_a = O_a \cap C_a - P_a - S_a$ and $F_{a,v} = O_{a,v} \cap C_{a,v} - P_{a,v} - S_{a,v}$ (see notation in Figure A.2). Applying this formula to Collins and Quillian's (1969) canary example yields the features listed next to the classes in their animal generalization hierarchy.

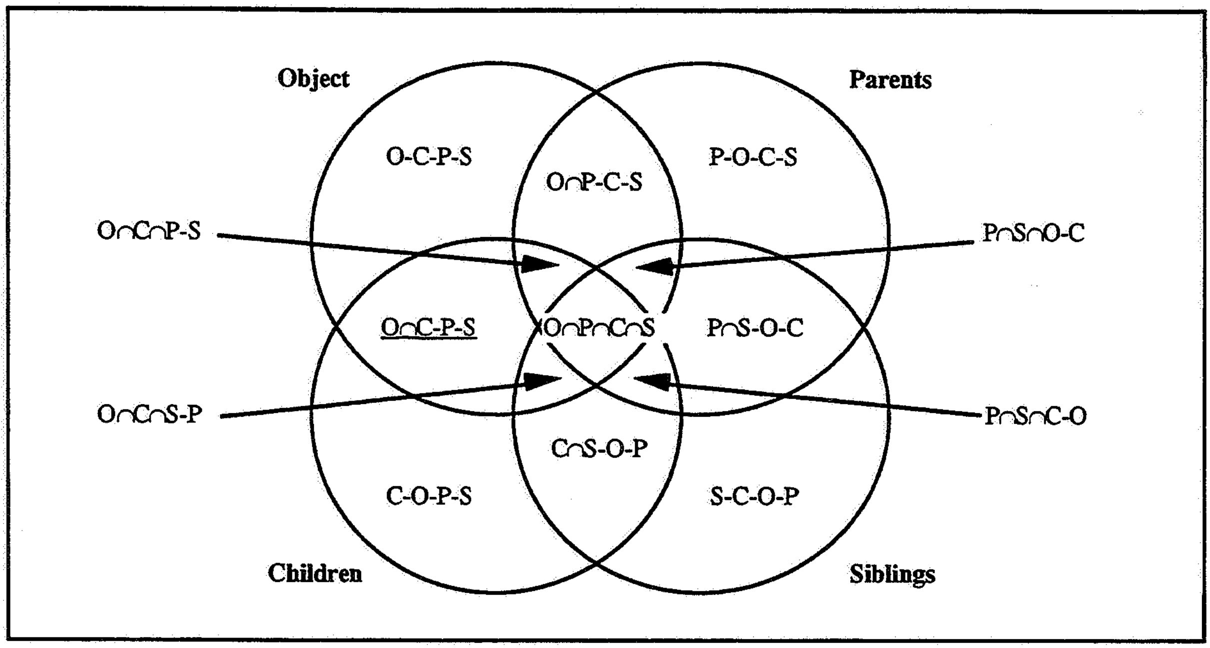



Figure A.1 Differentia in Set Theoretic Notation are $O \cap C - P - S$ (underlined in figure).

| | | | |
|---|---|---|---|
| Let | o | = | some object (class or instance) in a knowledge base |
| | p | = | {parent(s) of o} (null if o is the "root" object) |
| | c | = | {child(ren) of o} (null if o is an instance) |
| | s | = | {sibling(s) of o} (i.e., the children of o's parents) |
| | $O_a$ | = | {attributes of the object, o} |
| | $O_{a,v}$ | = | {attribute value pairs of the object, o} |
| | $S_a$ | = | {intersection of attributes of o's siblings} |
| | $S_{a,v}$ | = | {intersection of attribute value pairs of o's siblings} |
| | $P_a$ | = | {union of attributes of o's parents} |
| | $P_{a,v}$ | = | {union of attribute value pairs of o's parents} |
| | $C_a$ | = | {intersection of attributes of o's children} |
| | $C_{a,v}$ | = | {intersection of attribute value pairs of o's children} |

Figure A.2 Set Notation for Differentia

Unhappily, this characterization suffers from a weakness also found in Tversky's original formulation. In particular, attribute and attribute-value equality is viewed as a binary function. A more sensitive measurement of the typicality of a feature with respect to a class is to take all the features of the object different from those of its parent ($O_a - P_a$ and $O_{a,v} - P_{a,v}$) and then order these feature sets according to (1) the degree of *prototypicality*, P, that each attribute and attribute-value pair[5] of o manifests

[5]Note that we are characterizing the typicality of attributes and values as opposed to measuring the typicality of an object with respect to a class (see Tversky above).

with respect to o's children and (2) the degree of *uniqueness* (discriminatory power) that each attribute and attribute-value pair of o displays with respect to the object's siblings.

An attribute or attribute-value pair is prototypical of o (P has a value of 1) if it is found in each of o's children. Conversely, if the attribute or attribute-value pair is found in none of o's children, then it is not characteristic of o and P equals 0. Mathematically, the prototypicality of an attribute <a> with respect to its children c is:

$$P(<a>, c) = \frac{n}{N}$$

where N is the total number of entities (children) considered and n is the number of entities which have the attribute. Similarly, for all attributes common to all c, the prototypicality of an attribute value pair <a,v> with respect to its children c is:

$$P(<a,v>, c) = \frac{\sum_{i=1}^{N} \text{equality}(<a,v>, <a,v> \text{ of } c_i)}{N}$$

where N is the total number of children of o and $c_i$ refers the ith child of o.

Now that the attributes and attribute-value pairs of o are ordered according to their degree of prototypicality, calculate the discriminatory power of each of these attributes and attribute-value pairs with respect to the set of o's siblings, s. We use our previously defined discriminatory power metric, with a range of [0,1], redefined here with respect to o's sibling, s.

$$D(<a>, s) = \frac{N-n}{N}$$

where N is the total number of siblings and n is the number of siblings for which the indicated attribute, <a>, holds. Similarly, for all attributes of all s, the discriminatory power of a given attribute value pair, <a,v>, of o with respect to its siblings is:

$$D(<a,v>, s) = \frac{N - \sum_{i=1}^{N} \text{equality}(<a,v>, <a,v> \text{ of } s_i)}{N}$$

where N is the total number of siblings of o and $s_i$ refers the ith sibling of o. Just as the features of o are ordered according to their prototypicality, these formulas order the characteristics of o according to the degree that they discriminate the object from its siblings.

We can combine the measures of prototypicality (P) and discriminating power (D) to yield a combined metric which measures the *distinctive power* (DP) of a given attribute or attribute-value pair where $\alpha$ and $\beta$ indicate the relative importance of P and D:

$$DP(\langle a \rangle) = \frac{\alpha\, P(\langle a \rangle, c) + \beta\, D(\langle a \rangle, s)}{2}$$

$$DP(\langle a,v \rangle) = \frac{\alpha\, P(\langle a,v \rangle, c) + \beta\, D(\langle a,v \rangle, s)}{2}$$

both with a range of [0,1] where $\alpha + \beta = 1$.

In summary, to select the final set of distinguishing features for some given entity, first prune those features of the entity that it inherits from its parent(s). Next order those properties that are most typical of o (that have a large P), preferring attributes over attribute-value pairs. This set is then pruned by selecting the most differentiating features (those with a large D). This process continues until all features are exhausted or a satisfactory prototypical and discriminating set of attributes and attribute-value pairs is selected as measured by their *distinctive power* (DP).

# Appendix B

## Catalogue of Communicative Acts

### Rhetorical Acts

<u>DESCRIPTION</u>

TERSE

```
describe-by-defining
   define-by-logical-definition
   define-by-synonymic-definition
   define-by-antonymic-definition
describe-by-attribution
describe-by-indicating-purpose
describe-by-illustration
describe-by-classification
describe-by-constituency
```

EXTENDED

```
extended-description
   detail-by-attribution
   detail-by-indicating-purpose
   detail-by-illustration
   divide-by-constituency
   divide-by-classification
   divide-by-classification-and-constituency
   illustrate
   give-analogy
```

* not illustrated

** not yet implemented

COMPARISON

compare-point-by-point

compare-similarities/differences

compare-describe-in-turn

ANALOGY

describe-by-analogy

NARRATION

REPORT

narrate-report-temporally

narrate-report-topically

introduce-setting

narrate-temporal-sequence

narrate-spatial-sequence *

narrate-topical-sequence *

STORY

narrate-story

introduce-setting

narrate-event

narrate-state

tell-enablement/causation

tell-consequences

BIOGRAPHY

narrate-biography

LITERARY TECHNIQUES

narrate-event-MYSTERY

narrate-event-SUSPENSE *

narrate-event-SURPRISE *

## EXPOSITION

### OPERATIONAL INSTRUCTIONS

enable-to-do

instruct

### LOCATIONAL INSTRUCTIONS

identify-location

enable-to-get-to

### PROCESS EXPOSITION

explain-process

### PROPOSITION EXPOSITION

explain-proposition-by-description

explain-proposition-by-illustration

explain-reason-for-proposition

explain-purpose-for-proposition

explain-consequence-of-proposition

## ARGUMENT

argue-for-a-proposition

claim-proposition-by-inform

### DEDUCTION

convince-by-categorical-syllogism

convince-by-categorical-syllogism-modus-tollens

convince-by-hypothetical-syllogism **

convince-by-disjunctive-syllogism **

### INDUCTION

convince-by-evidence

convince-by-cause-and-evidence

### SUPPORTING TECHNIQUES

convince-by-illustration

convince-by-analogy

PERSUASION

```
request-enable-persuade
request
request-enable
enable-persuade *
persuade *
persuade-by-motivation
persuade-by-desirable-consequences
persuade-by-enablement
persuade-by-purpose-and-plan
```

## Illocutionary/Locutionary Acts

```
inform-by-assertion
request-by-asking
request-by-commanding
request-by-recommendation
warn-by-exclamation
concede-by-assertion
```

# Communicative Act Grammar

The following grammar is not used in the implementation but is presented here as an expository aid to indicate the correlation between the actions found in the headers and decompositions of plan operators (corresponding to the left-hand and right-hand side of the rules below). For details, readers are referred to the original definition of communicative acts in the chapters in which they appear (see index). The variables on the left-hand side of the rules are assumed to be given parameters when the plan operator is invoked. Certain variables on the right-hand side of these rules, however, may have been bound inside the body of the plan operators.

## DESCRIPTION

Describe(*S*, *H*, *entity*) --> Define(*S*, *H*, *entity*)

Define(*S*, *H*, *entity*) --> Inform(*S*, *H*, Logical-Definition(*entity*))

Define(*S*, *H*, *entity*) --> Inform(*S*, *H*, Synonymic-Definition(*entity*))

Define(*S*, *H*, *entity*) --> Inform(*S*, *H*, Antonymic-Definition(*entity*))

Describe(*S*, *H*, *entity*) --> Inform(*S*, *H*, Attribution(*entity*, *attributes*))

Describe(*S*, *H*, *entity*) --> $\forall p \in$ *purposes* Inform(*S*, *H*, Purpose(*entity*, *p*))

Describe(*S*, *H*, *entity*) --> $\forall e \in$ *examples* Inform(*S*, *H*, Illustration(*entity*, *e* ))

Describe(*S*, *H*, *entity*) --> Inform(*S*, *H*, Classification(*entity*))

Describe(*S*, *H*, *entity*) --> Inform(*S*, *H*, Constituency(*entity*))

Describe(*S*, *H*, *entity*) --> Define(*S*, *H*, *Entity*)
**optional**(Detail(*S*, *H*, *entity*))
**optional**(Divide(*S*, *H*, *entity*))
**optional**(Illustrate(*S*, *H*, *entity*)) $\vee$
Give-Analogy(*S*, *H*, *entity*))

Detail(*S*, *H*, *entity*) --> Inform(*S*, *H*, Attribution(*entity*, *attributes*))

Detail(*S*, *H*, *entity*) --> $\forall p \in$ *purposes* Inform(*S*, *H*, Purpose(*entity*, *p*))

Divide(*S*, *H*, *entity*) --> Inform(*S*, *H*, Classification(*entity*))
$\forall x \in$ subtypes(*entity*) **optional**(Describe(*S*, *H*, *x*))

Divide(*S*, *H*, *entity*) --> Inform(*S*, *H*, Constituency(*entity*))
$\forall s \in$ subparts(*entity*) **optional**(Detail(*S*, *H*, *s*))

Divide(*S*, *H*, *entity*) --> Inform(*S*, *H*, Classification(*entity*))
$\forall x \in$ subtype(*entity*, *x*) **optional**(Describe(*S*, *H*, *x*))
Inform(*S*, *H*, Constituency(*entity*))
$\forall x \in$ subparts(*entity*, *x*) **optional**(Detail(*S*, *H*, *x*))

Illustrate(*S*, *H*, *entity*) --> Inform(*S*, *H*, Illustration(*entity*, *example*))
**optional**(Describe(*S*, *H*, *example*))

Give-Analogy(*S*, *H*, *entity*) --> Inform(*S*, *H*, Analogy(*entity*, *analogue*))

Describe(*S*, *H*, *entity*) --> Inform(*S*, *H*, Analogy(*entity*, *analogue*))

## COMPARISON

Compare(*entity1*, *entity2*) --> **optional**(Inform(*S*, *H*, Inference(*entity1*, *entity2*)))
$\forall$ *attribute* $\in$ (differentia(*entity1*) $\wedge$ differentia(*entity2*))
Inform(*S*, *H*, Comparison-Contrast(*entity1*, *entity2*, *attribute*))

Compare(*entity1*, *entity2*) --> **optional**(Inform(*S*, *H*, Inference(*entity1*, *entity2*)))
Inform(*S*, *H*, Similarities(*entity1*, *entity2*))
Inform(*S*, *H*, Differences(*entity1*, *entity2*))

Compare(*entity1*, *entity2*) --> Describe(*S*, *H*, *entity1*)
Describe(*S*, *H*, *entity2*)
Inform(*S*, *H*, Comparison-Contrast(*entity1*, *entity2*))
**optional**(Inform(*S*, *H*, Inference(*entity1*, *entity2*)))

## NARRATION

Narrate(*S*, *H*, *events*) --> $\forall e \in$ *temporally-ordered-events*
Inform(S, H, Event(*e*))

Narrate(*S*, *H*, *events*) --> Introduce(*S*, *H*, *events*)
$\forall$*topic* $\in$ order-According-to-Salience(Topics(*events*))
Narrate-Sequence(*S*, *H*, Events-with-Topic(*events*, *topic*))

Introduce(*S*, *H*, *entities*) --> $\forall x \mid$ Main-Event(*x*, *entities*) $\vee$
Main-Time(*x*, *entities*) $\vee$
Main-Location(*x*, *entities*) $\vee$
Main-Agent(*x*, *entities*) $\vee$
Unknown-or-Unique-Entity(*x*, *entities*)
**optional**(Describe(*S*, *H*, *x*))

Narrate-Sequence(*S*, *H*, *events*) --> $\forall e \in$ select-and-order-temporally(*events*))
Inform(S, H, Event(*e*))

Narrate(*S*, *H*, *events+states*) --> Introduce(*S*, *H*, *events+states*)
$\forall x \in$ *chain*
Narrate-Event-or-State(*S*, *H*, *x*, *chain*)

Narrate-Event-or-State(*S*, *H*, *e*, *chain*) --> **optional**(Tell-Enablement/Causation(*S*, *H*, *e*, *chain*))
**optional**(Inform(*S*, *H*, Motivation(Agent(*e*),*e*)))

Inform(*S*, *H*, Event(*e*))
**optional**($\forall a \mid$ Agent(*a*,*e*) $\wedge \neg$ KNOW-ABOUT(*H*, *a*)
Describe(*S*, *H*, Agent(*e*)))

**optional**(Tell-Consequences(*S*, *H*, *e*, *chain*))

**optional**($\exists p$ Purpose(*e*,*p*) Inform(*S*, *H*, Purpose(*e*, *p*)))
**optional**(Inform(*S*, *H*, Narrator-Interpretation(*e*)))

Narrate-Event-or-State($S$, $H$, *state*, *chain*) --> **optional**(Tell-Enablement/Causation($S$, $H$, *state*, *chain*))

Inform($S$, $H$, State(*state*))
**optional**($\forall a$ | Agent($a$,*state*) $\wedge \neg$ KNOW-ABOUT($H$, $a$)
Describe($S$, $H$, Agent(*state*)))

**optional**(Tell-Consequences($S$, $H$, *state*))
**optional**(Inform($S$, $H$, Motivation(Agent(*state*), x)))

$\forall x$ **optional**(Inform($S$, $H$, Narrator-Interpretation(*state*, $x$)))

Tell-Enablement/Causation($S$, $H$, $e$, *chain*) --> $\forall x$ | Enablement($x$, $e$) $\wedge \neg$ Member($x$, *chain*)
**optional**(Inform($S$, $H$, Enablement($x$, $e$)))
$\forall x$ | Cause($x$,$e$) $\wedge \neg$ Member($x$, *chain*)
**optional**(Inform($S$, $H$, Cause($x$, $e$)))

Tell-consequences($S$, $H$, $e$, *chain*) --> $\forall x$ | Effect($e$, $x$) $\wedge \neg$ Member($x$, *chain*)
**optional**(Inform($S$, $H$, Effect($e$, $x$)))

Narrate($S$, $H$, *situations*, *agent*) --> **optional**(Describe($S$, $H$, *agent*))
$\forall$s | s $\in$ *ordered-situations* $\wedge$ Agent(*agent*, $s$)
Narrate-Event-or-State($S$, $H$, $s$)

Narrate-Event-or-State($S$, $H$, $e$, *chain*) --> **Do not:** Tell-Enablement/Causation($S$, $H$, $e$, *chain*))
**Do not:** **optional**(Inform($S$, $H$, Motivation(Agent($e$), $e$)))
Inform($S$, $H$, Event($e$))
**optional**($\forall a$ | Agent($a$,$e$) $\wedge \neg$ KNOW-ABOUT($H$, $a$)
Describe($S$, $H$, Agent($e$)))
**optional**($\exists x$ | Consequences($e$, $x$)
Tell-Consequences($S$, $H$, $e$, *chain*))
**Do not:** **optional**($\exists p$ Purpose($e$,$p$)
Inform($S$, $H$, Purpose($e$, $p$)))
**optional**(Inform($S$, $H$, Narrator-Interpretation($e$)))

## EXPOSITION

Enable($S$, $H$, Do($H$, *action*)) --> Inform($S$, $H$, Constraints(*action*))
$\forall x \in$ preconditions(*action*)
Request($S$, $H$, Do($H$, $x$))
Warn($S$, $H$, Danger(*action*))
$\forall$*subact* $\in$ subacts(*action*)
**optional**(Inform($S$, $H$, Constraints(*subact*)))
$\forall p \in$ preconditions(*subact*)
**optional**(Request($S$, $H$, Do($H$, $p$)))
Request($S$, $H$, Do($H$, *subact*))
$\forall y \in$ { $y$ | (Subaction(y, $x$) $\wedge \neg$ KNOW-HOW($H$, $y$)) }
Enable($S$, $H$, Do($H$, $y$))

Instruct($S$, $H$, Do($H$, *action*)) --> **optional**($\forall x$ Inform($S$, $H$, Purpose(*action*, $x$)) $\vee$
$\forall y$ Inform($S$, $H$, Motivation(*action*, $y$)) $\vee$
$\forall z$ Inform($S$, $H$, Cause(*action*, $z$)))
**optional**(Inform($S$, $H$, Constituency(result(*action*))))
Enable($S$, $H$, Do($H$, *action*))

Enable($S$, $H$, Go(*from-entity*, *to-entity*)) --> Inform($S$, $H$, Location(*to-entity*))

Enable($S$, $H$, Go(*from-entity*, *to-entity*)) --> $\forall p \in$ Path(*from-entity*, *to-entity*)
Request($S$, $H$, Do($H$, Go($p$, next-segment($p$))))
**optional**(Inform($S$, $H$, Location($p$)))
Inform($S$, $H$, Location(*to-entity*))

Explain($S$, $H$, *entity*) --> Define($S$, $H$, *entity*)
$\forall x$ **optional**(Inform($S$, $H$, Purpose(*entity*, $x$)))
Divide($S$, $H$, *entity*)
Narrate($S$, $H$, event-and-states(process(*entity*)))

Explain($S$, $H$, *proposition*) --> Describe($S$, $H$, predicate(*proposition*))
$\forall x \in$ terms(*proposition*)
Describe($S$, $H$, $x$)

Explain($S$, $H$, *proposition*) --> $\forall x \in$ examples(*proposition*)
Inform($S$, $H$, Illustration(*proposition*, $x$))

Explain-How($S$, $H$, *proposition*) --> $\forall x \in$ preconditions(*proposition*)
Inform($S$, $H$, Enablement($x$, *proposition*))
$\forall x \in$ motivations(*proposition*)
Inform($S$, $H$, Motivation($x$, *proposition*))
$\forall x \in$ causes(*proposition*)
Inform($S$, $H$, Cause($x$, *proposition*))

Explain-Why($S$, $H$, *proposition*) --> $\forall x \in$ purposes(*proposition*)
Inform($S$, $H$, Purpose(*proposition*, $x$))

Explain-Consequence($S$, $H$, *proposition*) --> $\forall x$ | Cause(*proposition*, $x$)
Inform($S$, $H$, Cause(*proposition*, $x$))

## ARGUMENT

Argue($S$, $H$, *proposition*) --> Claim($S$, $H$, *proposition*)
**optional**(Explain($S$, $H$, *proposition*))
Convince($S$, $H$, *proposition*)

Claim($S$, $H$, *proposition*) --> Inform($S$, $H$, *proposition*)

Convince($S$, $H$, *proposition*) --> Inform($S$, $H$, Universal-Definition(*superclass*, *predicate*))
Inform($S$, $H$, Logical-Definition(*entity*))
Inform($S$, $H$, Conclusion(*proposition*))

Convince($S$, $H$, *proposition*) --> Inform($S$, $H$, Universal-Definition(*predicate1*, *predicate2*))
Inform($S$, $H$, $\neg$ *predicate2*(*entity*))
Inform($S$, $H$, Conclusion(*proposition*))

Convince($S, H, proposition$) --> $\forall x \in$ order-by-importance(contra-evidence($proposition$))
Concede($S, H$, Counter-Evidence($proposition, x$))
$\forall x \in$ order-by-importance(evidence($proposition$))
Inform($S, H$, Evidence($proposition, x$))
**optional(if** BELIEVE($S$, $\neg$ BELIEVE($H, x$)) **then**
Convince($S, H, x$))

Convince($S, H, proposition$) --> Explain-How($S, H, proposition$)
$\forall x \in$ *evidence*
Inform($S, H$, Evidence($proposition, x$))
**optional**(Convince($S, H, x$))

Convince($S, H, proposition$) --> $\forall x \in$ examples($proposition$)
Inform($S, H$, Illustration($proposition, x$))

Convince($S, H, proposition$) --> Inform($S, H$, Analogy($proposition, analogue$))

Argue($S, H$, Do($H, action$)) --> Request($S, H$, Do($H, action$))
Enable($S, H$, Do($H, action$))
Persuade($S, H$, Do($H, action$))

Argue($S, H$, Do($H, action$)) --> Request($S, H$, Do($H, action$))

Argue($S, H$, Do($H, action$)) --> Request($S, H$, Do($H, action$))
Enable($S, H$, Do($H, action$))

Persuade($S, H$, Do($H, action$)) --> $\forall x$ | Motivation($x, action$)
Inform($S, H$, Motivation($x, action$))
**optional**($\forall y$ | Cause($y, x$) Inform($S, H$, Cause($y, x$)))

Persuade($S, H$, Do($H, action$)) --> $\forall x \in$ *desirable-events-or-states*
Inform($S, H$, Cause($action, x$))

Persuade($S, H$, Do($H, action$)) --> $\forall x \in$ *desirable-events-or-states*
Inform($S, H$, Enablement($action, x$))

Persuade($S, H$, Do($H, action$)) --> Inform($S, H$, Purpose($action, goal$))
Inform($S, H$, Constituent($plan, action$))

## ILLOCUTIONARY/LOCUTIONARY SPEECH ACTS ( with corresponding surface form)

| | |
|---|---|
| Inform($S, H, proposition$) --> Assert($S, H, proposition$) | (declarative syntax) |
| Request($S, H, proposition$) --> Ask($S, H, proposition$) | (interrogative syntax with question mark, "?") |
| Request($S, H, proposition$) --> Command($S, H, proposition$) | (imperative syntax) |
| Request($S, H, proposition$) --> Recommend($S, H, proposition$) | (declarative syntax with auxiliary "should") |
| Warn($S, H, proposition$) --> Exclaim($S, H, proposition$) | (imperative syntax with exclamation point, "!") |
| Concede($S, H, proposition$) --> Assert($S, H, proposition$) | (declarative syntax) |

# Appendix C

# Example Runs

## Map Display Domain

*How do I get from As to Merseburg?*

From As take Route 21 Southeast five kilometers to the intersection of Route 92 and Route 21. At that intersection take Route 92 Northwest thirty-six kilometers to Oelsnitz. From Oelsnitz continue on Route 92 Northwest thirty-eight kilometers to the intersection of Route 175 and Route 92. From that intersection take Route 2 Northeast twenty-eight kilometers to the intersection of Route 2 and Route 176. From there take Route 176 Northwest thirty-nine kilometers to the intersection of Route 91 and Route 176. At that intersection take Route 91 Northeast sixteen kilometers to Merseburg. Merseburg is a town located at 51° latitude and 12° longitude one hundred thirty-two kilometers Northwest of town As.

*How do I get from Leipzig to Friedersee?.*

Friedersee is a lake located at 52° latitude and 12° longitude forty kilometers Northeast of the town Leipzig. From Leipzig take Route 183 Northeast thirty-four kilometers to the intersection of Route 183 and Route 2. From there continue on Route 2 Northeast four kilometers to the intersection of Route 107, Route 183 and Route 2. At that intersection take Route 183 Northwest eleven kilometers to the intersection of Route 183 and Route 100. From there take Route 183 West 4 kilometers. The lake Friedersee is located two kilometers North of this point.

*Compare Freiberg and Becken.*

Both Freiberg and Becken are carto-objects. However, Freiberg is a town and Becken is a lake. Freiberg has a location of 13° longitude and 51° latitude whereas Becken has a location of 11° longitude and 51° latitude. Freiberg has a functional-status of partially-functional; Becken has a functional-status of fully-functional.

## Knowledge Replanning System

*What is an F-15?*

An F-15 is an American fighter for air-to-air-interdiction. It has several standard configuration loads (scl-d1 and scl-d2), a speed of 350 miles per hour, a communications radius of 2,220,000, an empty weight of 12,500, a maximum weight of 24,500, a maximum fuel capacity of 10,600, a turn time of 3,600, a ferrying range of 4,630,000, and a ferry package of 3,000. There are five classes of F-15s: the F-15A, the F-15B, the F-15C, the F-15D, and the F-15E.

*Why is prescanning the ATO necessary?*

The purpose of prescanning the Air Tasking Order is to test the validity of the Air Tasking Order. Prescanning the Air Tasking Order is part of planning an Air Tasking Order.

## Mission Simulation Domain (LACE)

*What is OCA100?*

OCA100 is an offensive counter air mission. It has an aircraft unit of 86-Tactical-Fighter-Wing, an aircraft type of F-16, two active aircraft, a standard configuration load of SCL-B21, and a target of Delta-Runway-08s26s.

*What happened to OCA100?*

Offensive Counter Air Mission 100 was an air strike against Delta airfield. It began mission execution at 8:41::40 Tuesday December 2, 1987. It received two F-16 aircraft from the 86-Tactical-Fighter-Wing. Seven minutes later it was flying its ingress route. Then ten minutes later it bombed its target. It began flying its egress route. Thirty-six minutes later it ended its mission. It generated its post-mission report.

*Compare a MiG-27 to a MiG-29.*

A MiG-27 and a MiG-29 are similar fighters. They both have a maximum fuel capacity of 2,000, 233.0 meters per pound of fuel, and a maximum speed of five meters per second. However, the MiG-27 fighter and the MiG-29 fighter have different nick-names (Flogger-D versus Fulcrum), different cruise-speeds (490 meters-per-second versus 660 meters-per-second), different offensive capabilities (very-weak versus very-strong), different defensive capabilities (strong versus weak), and different combat radii (1,380 kilometers versus 1,200 kilometers).

*What is a flat-face?*

A flat-face is a radar. It has a default power of 500 kilowatts, a default band designator of c, a default emplacement of ten minutes, a default vertical-beamwidth of 5°, a default horizontal-beamwidth of 2°, and an accuracy of 100 meters in range and 5° in angle.

## NEUROPSYCHOLOGIST

*What is a brain?*

A brain is an organ located in the skull. It has gray nerve tissue and white nerve fibers. It consists of two regions: the left-hemisphere and the right-hemisphere.

*Why did you diagnose Korsakoff's disorder for Patient1?*

Patient1 has Korsakoff's disorder with 75% probability. An apathetic demeanor indicates a 70% probability of Korsakoff's disorder. A poor memory and low IQ scores indicates a 80% probability of Korsakoff's disorder.

## Vertebrate Domain

*What is a canary?*

A canary is a yellow bird with a Canary Islands origin, that sings, and is domesticated.

*What is a fish?*

Fish are vertebrates that swim, have fins, have gills, are aquatic, eat vegetation and fish, have scales, and are cold-blooded.

*What is the difference between a fish and a bird?*

Fish are vertebrates that swim, have fins, have gills, are aquatic, eat vegetation and fish, have scales, and are cold-blooded. Birds, on the other hand, are vertebrates that fly, have wings, are terrestrial, eat seeds, have feathers, and are warm-blooded. Fish and birds have the same superclass, different locomotion, different propellors, different environments, different diets, different covering, and different blood-temperatures. Therefore, they are different entities.

# Bibliography

The following abbreviations are used in this bibliography:

| | | |
|---|---|---|
| COLING | — | International Conference on Computational Linguistics |
| ACL | — | Association for Computational Linguistics |
| ECACL | — | European Chapter of the Association for Computational Linguistics |
| TINLAP | — | Theoretical Issues in Natural Language Processing |
| AAAI | — | American Association for Artificial Intelligence |
| IJCAI | — | International Joint Conference on Artificial Intelligence |
| ACM | — | Association for Computing Machinery |


Allen, J. F. 1979. A Plan-based Approach to Speech Act Recognition. Ph.D. dissertation, Department of Computer Science, University of Toronto, Toronto, Canada.

Allen, J. F. 1983. "Recognizing Intentions from Natural Language Utterances." *Computational Models of Discourse*, M. Brady and R. C. Berwick, editors. 107-166. MIT Press.

Allen, J. F. 1984. "Towards a General Theory of Time and Action." *Artificial Intelligence* 23(2):123-154.

Allen, J. F. 1987. *Natural Language Understanding*. Menlo Park, CA: Benjamin/Cummings Publishing Company.

Allen, J. F. editor, June, 1988. "Special Issue on Tense and Aspect." *Computational Linguistics* 14(2).

Allen, J. F. and C. R. Perrault. 1980. "Analyzing Intention in Utterances." *Artificial Intelligence* 15(3):143-178. [reprinted in Grosz, Sparck Jones and Webber, 1986, *Readings in Natural Language Processing*, 441-458.]

Allen, J., S. Guez, L. Hoebel, E. Hindelman, K. Jackson, A. Kyburg and D. Traum. January, 1990. "The Discourse System Project." University of Rochester, NY, TR 317.

Allen, J. and E. Hinkelman. 1989. "Using structural constraints for speech act interpretation." Proceedings of the DARPA Speech and Natural Language Workshop, Cape Cod, MA, 15-18 October.

Alshawi, H. 1983. "Memory and Context Mechanisms for Automatic Text Processing." Cambridge University Computer Laboratory TR-60.

Anderson, J. M. 1971. *The Grammar of Case*. Cambridge, UK: Cambridge University Press.

Anken, C. S. October, 1989. "LACE: Land Air Combat in Eric." Rome Air Development Center TR 89-219.

Appelt, D. 1980. "Problem Solving Applied to Language Generation.", Proceedings of the 18th Annual Meeting of the ACL, Philadelphia, PA, 1980. 59-63.

Appelt, D. E. March, 1982. "Planning Natural Language Utterances to Satisfy Multiple Goals." SRI Technical Note 259.

Appelt, D. E. 1985. *Planning English Sentences*. England: Cambridge University Press.

Appelt, D. E. 1987. "Bidirectional Grammars and the Design of Natural Language Generation Systems." Position Papers for TINLAP-3, 1987. 206-212.

Aristotle. 1926. *The 'Art' of Rhetoric*. trans. J. H. Freese, Cambridge, MA: Loeb Classical Library series.

Atkins, B. T. S. 1989. "Building a Lexicon: Reconciling Anisomorphic Sense Differentiations in Machine-readable Dictionaries." Proceedings of the BBN Natural Language Symposium, BBN, Cambridge, MA, 29 November - 1 December, 1989.

Austin, J. 1962. *How to do Things with Words*. J. O. Urmson, editor. England: Oxford University Press.

Barr, A. and E. A. Feigenbaum, editors. 1981. *Handbook of Artificial Intelligence*, Volume 1. Los Altos, CA: William Kaufman.

Bateman, J. A. 1985. Utterances in Context: Towards a Systemic Theory of Intersubjective Achievement of Discourse. Ph.D. dissertation, University of Edinburgh, Edinburgh, Scotland.

Bateman, J. A. 1988. "From Systemic-Functional Grammar to Systemic-Functional Text: Escalating the Exchange." Hovy, E. H., D. D. McDonald, S. R. Young and D. E. Appelt, Proceedings of AAAI-88 Workshop in Text Planning and Realization, Saint Paul, MN, August, 1988. 123-132.

Baum, R. 1975, 1981. *Logic*. New York: Holt, Rinehart, and Winston.

Bayes, R. T. 1763. An Essay Toward Solving a Problem in the Doctrine of Chance: Philosophical Transactions of the Royal Society.

Becker, J. D. 1975. "The Phrasal Lexicon." Bolt, Beranek and Newman TR-3081.

Bench-Capon, T. J. M., D. Lowes and A. M. McEnery. 1990. "Using Toulmin's Argument Formalism to Explain Logic Programs." Proceedings of the Explanation Workshop, Department of Computer Science, University of Manchester, 25-27 April, 1990. [10 pp]

Bienkowski, M. A. September, 1986. A Computational Model of Extemperaneous Elaboration. Cognitive Science Laboratory TR 1, revised University of Connecticut Ph.D. dissertation.

Birnbaum, L. 1982. "Argument Molecules: A Functional Representation of Argument Structure." Proceedings of the 3rd National Conference on Artificial Intelligence (AAAI-82), Pittsburgh, PA, 1982. 63-65.

Birnbaum, L., M. Flowers and R. McGuire. 1980. "Towards an AI Model of Argumentation." First National Conference on Artificial Intelligence (AAAI-80), Stanford University, CA, 18-21 August, 1980. 313-315.

Black, J. B. and R. Wilensky. 1979. "An Evaluation of Story Grammars." *Cognitive Science* 3(1979)213-230.

Bobrow, D. 1977. "GUS, A Frame Driven Dialog System." *Artificial Intelligence* 8(1977)155-173, [reprinted in Grosz, Sparck Jones and Webber, 1986, *Readings in Natural Language Processing*, 595-604].

Bobrow, D. G. and T. Winograd. 1977. "An Overview of KRL, A Knowledge Representation Language." *Cognitive Science* 1.

Bock, K. 1987. "Exploring Levels of Processing in Sentence Production." *Natural Language Generation*, G. Kempen, editor. 351-363. Dordrecht: Martius Nijhoff.

Bock, K. and R. K. Warren. 1985. "Conceptual Accessibility and Syntactic Structure in Sentence Formulation." *Cognition* 21(1985):47-67.

Boguraev, B. K. 1979, August. Automatic Resolution of Linguistic Ambiguities. Cambridge University Computer Laboratory TR 11, Cambridge, England.

Boguraev, B. 1989. "Building a Lexicon: The Contribution of Computational Lexicology." BBN Natural Language Symposium, BBN, Cambridge, MA, 29 Nov - 1 Dec, 1989.

Bossie, S. 1981. "A Tactical Component for Text Generation: Sentence Generation Using a Functional Grammar." University of Pennsylvania TR MS-CIS-81-5.

Bossie, S. and I. Mani. January-February 1986. "An Overview of Research in Natural Language Generation." *TI Engineering Journal* 52-57.

Brachman, R. J. 1985. "On the Epistemological Status of Semantic Networks." *Readings in Knowledge Representation*, R. J. Brachman and H. J. Levesque, editors. 191-216. CA: Morgan Kaufman. [Originally published in Findler, N. V., editor, *Associative Networks: Representation and Use of Knowledge in a Computer*, Academic Press: NY, 1979, 3-50.]

Brady M. and R. Berwick, editors. 1983. *Computational Models of Discourse*, Cambridge, Massachusetts: MIT Press.

Brooks, S. D. and Hubbard, M. 1905. *Composition Rhetoric*. New York: American Book Company.

Brown, C. A. and Zoellner, R. 1968. *The Strategy of Composition -- A Rhetoric with Readings*. New York: Ronald Press.

Brown, G. and Yule, G. 1983. *Discourse Analysis*. Bath Press, Avon: Cambridge University Press.

Brown, J. S. and R. R. Burton. 1978. "Diagnostic Models for Procedural Bugs in Mathematical Skills." *Cognitive Science* 2(2):155-192.

Bruce, B. C. 1975. "Generation as a Social Action." Proceedings of TINLAP, 64-67 [reprinted in Grosz, Sparck Jones and Webber, 1986, *Readings in Natural Language Processing*, 419-422].

Bruder, G. A. et al. 1986. "Deictic Centers in Narrative: An Interdisciplinary Cognitive-Science Project." State University of New York at Buffalo, Computer Science Department TR 86-20, Buffalo, NY.

Buchanan, B. G. and Shortliffe, E. H. 1984. *Rule Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Reading, MA: Addisson-Wesley.

Burger, J. 1989. "User Models for Intelligent Interfaces." IJCAI-89 Workshop: A New Generation of Intelligent Interfaces, Detroit, MI, August 22, 1989. 17-20.

Carberry, S. September, 1988. "Modeling the User's Plans and Goals." *Computational Linguistics: Special Issue on User Modeling* 14(3):23-37.

Carletta, J. 1990a, March 16. "A General Architecture for Interactive Explanations." University of Edinburgh, unpublished manuscript.

Carletta, J. 1990b. "An Architecture Facilitating Repair and Replanning in Interactive Explanations." Proceedings of the Explanation Workshop, Department of Computer Science, University of Manchester, 25-27 April, 1990.

Carter, D. M. 1985. "A Shallow Processing Approach to Anaphor Resolution." University of Cambridge, Computer Laboratory TR-88.

Cawsey, A. 1989. "Explanatory Dialogues." *Interacting with Computers* 1(1):69-92.

Cawsey, A. 1990. "Generating Explanatory Discourse." In *Current Research in Natural Language Generation*, R. Dale., C. Mellish and M. Zock, editors.

Cicero. 1949. *De Inventione* and *De Optimo Genere Oratorum*. trans. H. M. Hubbell, Cambridge, MA: Loeb Classical Library series. [the latter book a brief description of 'the best kind of orator'.]

Clancey, W. J. 1979. "Dialogue Management for Rule-Based Tutorials." Proceedings of the Sixth IJCAI, Tokyo, Japan, 20-23 Aug, 1979. 155-161.

Clancey, W. J. 1983. "The Epistemology of a Rule-Based Expert System -- a Framework for Explanation." *Artificial Intelligence* 20(1983):215-251.

Clancey, W. J. July, 1986. "From GUIDON to NEOMYCIN and HERACLES in Twenty Short Lessons." Stanford University Department of Computer Science TR STAN-CS-87-1172.

Clancey, W. J. and R. Letsinger. 1981. "NEOMYCIN: Reconfiguring a Rule-Based Expert System for Application to Teaching." Proceedings of the Seventh IJCAI, University of British Columbia, Vancouver, B.C. Canada, 24-28 August, 1981. 829-835.

Cohen, P. R. 1978. "On Knowing What to Say: Planning Speech Acts." University of Toronto TR-118.

Cohen, P. R. 1981. "The Need for Referent Identification as a Planned Action." Proceedings of the Seventh IJCAI, Vancouver, B.C., Canada, 1981. 31-36.

Cohen, P. R. 1984. "The Pragmatics of Referring and the Modality of Communication." *Computational Linguistics* 10(2):97-146.

Cohen, P. R. and H. J. Levesque. 1985. "Speech Acts and Rationality." Proceedings of the 23rd Annual Meeting of the ACL, Chicago, 1985. 49-59.

Cohen, P. R. and R. C. Perrault. 1979. "Elements of a Plan-Based Theory of Speech Acts." *Cognitive Science* 3(3):177-212 [reprinted in Grosz, Sparck Jones, and Webber, 1986, *Readings in Natural Language Processing*, 423-440, and in *Readings in Artificial Intelligence*, Bonnie L. Webber and Nils J. Nilsson, editors, 1981, Tioga Publishing Company, 478-495.]

Cohen, R. 1986, September. A Computational Model for the Analysis of Arguments. University of Waterloo Department of Computer Science TR CS-86-41, Waterloo, Ontario, Canada.

Cohen, R. 1987. "Analyzing the Structure of Argumentative Discourse." *Computational Linguistics* 13(1-2):11-23.

Cole, P. and J. L. Morgan, editors. 1975. *Syntax and Semantics 3: Speech Acts*, New York: Academic Press.

Cole, P. and J. M. Sadock, editors. 1977. *Syntax and Semantics 8: Grammatical Relations*. New York: Academic Press.

Collier, J. T., M. Evens, D. Hier and P. Li. forthcoming. "Generating Case Reports for a Medical Expert Systems." *International Journal of Expert Systems*.

Collins A. M. and Quillian M. R. 1969. "Retrieval Time from Semantic Memory." *Journal of Verbal Learning and Verbal Behavior* 8(1969):240-247.

Comrie, B. 1976. *Aspect.* Cambridge, England: Cambridge University Press.

Conklin, E. J. 1983. Data-Driven Indelible Planning of Discourse Generation using Salience. Department of Computer and Information Science, University of Massachusetts, COINS TR 83-13.

Contant, C. 1986. Génération automatique de texte: application au sous-language boursier. M. A. thesis, Département de linguistique, Université de Montréal, Montréal, Canada.

Crocker, L. 1944. *Argumentation and Debate.* New York: American Book Co.

Dale, R. 1989. "Cooking up Referring Expressions." Proceedings of the 27th Annual Meeting of the ACL, Vancouver, B.C. Canada, June, 1989. 68-75.

Dale, R. 1990. "Generating Recipes: An Overview of Epicure." In *Current Research in Natural Language Generation*, R. Dale., C. Mellish and M. Zock, editors.

R. Dale., C. Mellish and M. Zock, editors. 1990. *Current Research in Natural Language Generation.* Based on Extended Abstracts from the Second European Workshop on Natural Language Generation, University of Edinburgh, Edinburgh, Scotland, 6-8 April, 1989. London: Academic Press. ISBN 0-12-200735-2, 356 pages.

Danlos, L. 1987. *The Linguistic Basis of Text Generation*, Studies in Natural Language Processing, Cambridge University Press.

Danlos, L. 1984. "Conceptual and Linguistic Decisions in Generation." Proceedings of the Tenth COLING, Stanford, CA, July, 1984. 319-325.

Davey, A. 1978. *Discourse Production: A Computer Model of Some Aspects of a Speaker.* Edinburgh University Press.

Davis, R. 1976. Applications of Meta-level Knowledge to the Construction, Maintenance, and use of Large Knowledge Bases. Stanford University Ph.D. dissertation, Stanford, CA. [reprinted in R. Davis and D. B. Lenat, editors. *Knowledge-Based Systems in Artificial Intelligence*, New York: McGraw-Hill, 1982.]

Davis, R. 1980. "Meta-rules: Reasoning about control." *Artificial Intelligence* 15(3):179-222.

Dawson, B., R. Brown, C. Kalish and S. Goldkind. May, 1987. "Knowledge-based Replanning System." Rome Air Development Center TR 87-60.

de Beaugrande, R. 1984. *Text Production: Towards a Science of Composition*, Vol. XI in series Advances in Discourse Processing, Alex Publishing Corporation.

de Joia, A. and Stenton, A. 1980. *Terms in Linguistics: A Guide to Halliday.* London: Batsford Academic and Educational Ltd.

De Smedt, K. and G. Kempen. 1987. "Incremental Sentence Production, Self-Correction, and Coordination." *Natural Language Generation*, G. Kempen, editor. 365-376. Dordrecht: Martinus Nijhoff Publishers.

Dehn, N. 1981. "Story Generation After TALE-SPIN." Proceedings of the Seventh IJCAI, University of British Columbia, Vancouver, B.C., Canada, 24-28 August, 1981. 16-18.

Dik, S. C. 1980. "Seventeen Sentences: Basic Principles and Applications of Functional Grammar." *Syntax and Semantics 13*, Moravesik and Wirth, editors. 45-76.

Dixon, P. 1987. *Rhetoric*, The Critical Idiom, London: Methuen.

Dowty, D. 1982. "Grammatical Relations and Montague Grammar." *The Nature of Syntactic Representation*, P. I. Jacobson and G. K. Pullum, editors. Holland: Reidel Publishing.

Dowty, D. editor. 1986. *Linguistics and Philosophy: Special Issue on Tense and Aspect in Discourse* 9(1).

Dyer, M. G. 1981. "Integration, Unification, Reconstruction, Modification: An Eternal Parsing Braid." Proceedings of the Seventh IJCAI, University of British Columbia, Vancouver, B.C. Canada, 24-28 August, 1981. 37-42.

Ehrich, V. 1987. "The Generation of Tense." *Natural Language Generation*, G. Kempen, editor. 423-440. Dordrecht: Martinus Nijhoff.

Ehrlich, K. and P. N. Johnson-Laird. 1982. "Spatial Descriptions and Referential Continuity." *Journal of Verbal Learning and Verbal Behavior* 21(1982)296-306.

Elhadad, M., D. D. Seligmann, S. Feiner and K. McKeown. 1989. "A Common Intentional Description Language for Interactive Multi-Media Systems." IJCAI-89 Workshop: A New Generation of Intelligent Interfaces, Detroit, MI, August 22, 1989. 46-52.

Evens, M. W., editor. 1988. Relational Models of the Lexicon: Representing Knowledge in Semantic Networks. *Studies in Natural Language Processing*, Cambridge, England: Cambridge University Press.

Fallside, F. and W. A. Woods, editors. 1985. *Computer Speech Processing*, Englewood Cliffs, NJ: Prentice Hall.

Fawcett, R. P. 1988. "Language Generation as Choice in Social Interaction." *Advances in Natural Language Generation*, Vol. 2, M. Zock and G. Sabah, editors. 27-49. London: Pinter.

Fehrle, T., T. Strothotte and M. Szardenings. 1988. "Generating Pictorial Presentations for Advice-Giving Dialog Systems." *Lecture Notes in Computer Science*, Springer Verlag.

Feiner, S. November, 1985. "APEX: An Experiment in the Automated Creation of Pictorial Explanations." *IEEE Computer Graphics and Application* 5(11):29-37.

Feiner, S., C. Beshers, N. Chin, P. Karp, D. Kurlander and D. Seligmann. 1989. "Knowledge-Based Graphics." HP Computer Graphics Symposium, June 14, 1989.

Feiner, S. and K. McKeown. 1990. "Coordinating Text and Graphics in Explanation Generation." Proceedings of the Eighth National Conference on Artificial Intelligence, (AAAI-90), Boston, MA, 29 July - 3 August, 1990. 442-449.

Fikes, R. E. and N. J. Nilsson. 1971. "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving." *Artificial Intelligence* 2(1971):189-208.

Fillmore, C. J. 1968. "The Case for Case." *Universals in Linguistic Theory*, E. Bach and R. Harms, editors. New York: Holt, Rinehart and Winston.

Fillmore, C. J. 1977. "The Case for Case Reopened." *Syntax and Semantics 8: Grammatical Relations*, P. Cole and J. M. Sadock, editors. 59-81. New York: Academic Press.

Fimbel, E., H. Gröscot, J. Lancel and N. Simonin. 1985. Proceedings of the 2nd Annual Conference of the ECACL, 226-231.

Flower, L. 1985. *Problem-Solving Strategies for Writing*. Second edition. Orlando, Florida: Harcourt Brace Jovanovich.

Franc, B. 1987. Preliminary Safety and Risk Assessment for Existing Hydraulic Structures -- An Expert Systems Approach. University of Minnesota, Department of Mechanical Engineering, Minneapolis, MN.

Freeley. May, 1969. *Argumentation and Debate: Rational Decision Making*, 2nd edition. Belmont, California: Wadsworth Publishing Co. [Toulmin description, pp 139-143.]

Frisch, A. M. and D. Perlis. 1981. "A Re-Evaluation of Story Grammars." *Cognitive Science* 5(1981):79-86.

Dik, S. C. 1978. *Functional Grammar*. North-Holland Linguistic Series, New York: North-Holland.

Galliers, J. R. July, 1989. A Theoretical Framework for Computer Models of Cooperative Dialogue, Acknowledging Multi-Agent Conflict. Cambridge University TR 172, Cambridge, England.

Garnham, A. 1983. "What's Wrong with Story Grammars." *Cognition* 15(1983):145-154.

Gazdar, G. 1979. *Pragmatics: Implicature, Presupposition, and Logical Form*. New York: Academic Press.

Gazdar, G. 1982. "Phrase Structure Grammar." *On the Nature of Syntactic Representation*, P. Jacobson and G. K. G. K. Pullum, editors. Dordrecht: Reidel.

Gazdar G., Klein K., Pullum, G. and Sag, I. 1985. *Generalized Phrase Structure Grammar*. Oxford: Basil Blackwell.

Geller, J. July, 1988. A Knowledge Representation Theory for Natural Language Graphics. State University of New York at Buffalo Ph.D. dissertation.

Georgeff, M. P. 1987. "Planning." *Annual Review of Computer Science* 2(1987):359-400.

Golden, C. J. 1985. "Computational Models of the Brain." *Computers in Human Behavior*, Vol. 1, 35-48. Pergamon Press.

Goldman, N. M. 1975. "Conceptual Generation." *Conceptual Information Processing*, R. C. Schank, editor. 289-371. Amsterdam: North-Holland.

Goodman, B. A. October-December 1986. "Reference Identification and Reference Identification Failures." *Computational Linguistics* 12(4):273-305.

Granville, R. 1984. "Controlling Lexical Substitution in Computer Text Generation." Proceedings of the 22nd Annual Meeting of the ACL, 381- 384.

Grice, H. P. 1957. "Meaning." *Philosophical Review* 66(1957):377-388.

Grice. 1975. "Logic and Conversation." *Syntax and Semantics 3: Speech Acts*, P. Cole and J. L. Morgan, editors. 45-58. New York: Academic Press.

Grimes, J. E. 1975. *The Thread of Discourse*. The Hague and Paris: Mouton.

Grimshaw, J. D. 1987. "Explanation Capabilities in Expert Systems." unpublished paper for AFIT graduate seminar in "Advanced Topics in AI."

Grishman, R. 1986. *Computational Linguistics: an Introduction*. Cambridge University Press.

Grishman, R. 1979. "Response Generation in Question-answering Systems." Proceedings of the 17th Annual Meeting of the ACL, La Jolla, CA, August, 1979. 99-102.

Grishman, R. and L. Hirschman. 1978. "Question-answering from Natural Language Medical Data Bases." *Artificial Intelligence* 11(1978)25-43.

Grosz, B. J. 1977. "The Representation and Use of Focus in a System for Understanding Dialogs." Proceedings of the Fifth IJCAI, Cambridge, MA, 1977. 67-76.

Grosz, B. J. and C. Sidner. July-September, 1986. "Attention, Intentions, and the Structure of Discourse." *Computational Linguistics* 12(3):175-204. [see also BBN TR 6097 "The Structures of Discourse Structure", November 1985].

Grosz, B. J. and C. Sidner, 1989. "Plans for Discourse." *Intentions and Communications*, P. Cohen, J. Morgan and M. Pollack, editors. MIT Press. [Harvard University TR-11-87].

Grosz, B. J., Sparck Jones, K. and Webber, B. L. 1986. *Readings in Natural Language Processing*. Los Altos, CA: Morgan Kaufman.

Guastello, S. J. and M. Traut. 1989. "Verbal versus pictorial representations of objects in a human-computer interface." *International Journal of Man-Machine Studies* 31(1989):99-120.

Haimowitz, I. J. December, 1989. "Generating Empathetic Responses with Individual User Models." MIT Laboratory for Computer Science TR-461.

Halliday, M. A. K. 1976. *System and Function in Language*. London: Oxford University Press.

Halliday, M. A. K. 1985. *An Introduction to Functional Grammar*. London: Edward Arnold.

Halliday, M. A. K. and Hasan, R. 1976. *Cohesion in English*. London: Longman.

Hasling, D. W., W. J. Clancey and G. Rennels. November, 1983. "Strategic Explanations for a Diagnostic Consultation System." Stanford University, Department of Computer Science TR STAN-CS-83-996.

Haviland, S. E. and H. H. Clark. "What's New? Acquiring new information as a processing comprehension." *Journal of Verbal Learning and Verbal Behavior* 13, 512-521.

Hayes, P., P. Anderson and S. Safier. 1985. "Semantic Caseframe Parsing and Syntactic Generality." Proceedings of the 23rd Annual Meeting of the ACL, Chicago, 153-160.

Hilton, M. L. July, 1987. "ERIC: An Object-Oriented Simulation Language." Rome Air Development Center TR-87-103.

Hilton, M. L. and C. S. Anken. February, 1990. "Map Display System: An Object-Oriented Design and Implementation." Rome Air Development Center Technical Report 90-54.

Hilton, M. L. and J. D. Grimshaw. February, 1990. "ERIC Manual." Rome Air Development Center Technical Report 90-84.

Hinkelman, E. and J. Allen. 1989. "Two Constraints on Speech Act Ambiguity." Proceedings of the 27th Annual Meeting of the ACL, University of British Columbia, Canada, 1989.

Hinrichs, E. W. June, 1988. "Tense, Quantifiers, and Contexts." *Computational Linguistics* 14(2):3-14.

Hobbs, J. R. 1979. "Coherence and Coreference." *Cognitive Science* 3(1979):67-90.

Hobbs, J. R. 1978. "Resolving Pronoun References." *Lingua* 44(1978):311-338.

Houghton, G. April, 1986. Production of Language in Dialogue: A Computational Model. Ph.D. dissertation, University of Sussex, England.

Hovy, E. 1985. "Integrating Text Planning and Production in Generation." Proceedings of the Ninth IJCAI, Los Angeles, CA, 18-23 August, 1985. 848-851.

Hovy, E. March, 1987. Generating Natural Language Under Pragmatic Constraints. Ph.D. dissertation, Yale University TR-521, Department of Computer Science.

Hovy, E. 1988a. "Planning Coherent Multisentential Text." Proceedings of the 26th Annual Meeting of the ACL, Buffalo, NY, June 7-10, 1988. 163-169.

Hovy, E. 1988b. "Two Types of Planning in Language Generation." Proceedings of the 26th Annual Meeting of the ACL, Buffalo, NY, 7-10 June, 1988. 179-186.

Hovy, E. 1988c. "Approaches to Planning of Coherent Text." Fourth International Workshop on Natural Language Generation, Los Angeles, CA, 1988, July.

Hovy, E. 1990. "Unresolved Issues in Paragraph Planning." In *Current Research in Natural Language Generation*, R. Dale., C. Mellish and M. Zock, editors, 17-45.

Hovy, E. H., D. D. McDonald, S. R. Young and D. E. Appelt, 1988. Proceedings of the AAAI-88 Workshop on Text Planning and Realization, Saint Paul, MN, August, 1988.

Hovy, E. and Y. Arens. 1990. "When is a Picture Worth a Thousand Words? -- Allocation of Modalities in Multimedia Communication." AAAI Spring Symposium on Human-Computer Communication, Stanford, CA, March, 1990.

Hovy, E. and K. McCoy. 1989. "Focusing Your RST: A Step Toward Generating Coherent Multisentential Text." Submitted to Conference of Cognitive Science Society, March 20, 1989.

Jacobs, P. S. October-December, 1985. "PHRED: A Generator for Natural Language Interfaces." *Computational Linguistics* 11(4):219-242.

Jacobs, P. S. November, 1987. "Knowledge Intensive Natural Language Generation." *Artificial Intelligence* 33(3):325-378.

Jacobs, P. S. 1988. "Achieving Bidirectionality." Proceedings of the 12th COLING, Budapest, Hungary, August, 1988. 267-269.

Johnson-Laird, P. N. 1983. *Mental Models*. Cambridge, MA: Harvard University Press.

Joshi, A. K. 1987. "The Relevance of Tree Adjoining Grammar to Generation." *Natural Language Generation*, G. Kempen, editor. 233-252. Dordrecht: Martinus Nijhoff.

Joshi, A., B. Webber and R. Weischedel. 1984. "Living up to Expectations: Computing Expert Responses." Proceedings of the 1984 National Conference on Artificial Intelligence, Dallas, TX, 1984. 169-175.

Kalish, C. September, 1987. "A Portable Natural Language Interface." Rome Air Development Center TR 87-155.

Kalita, J. 1989. "Automatically Generating Natural Language Reports." *International Journal of Man-Machine Studies* 30(1989):399-423.

Kane, T. S. and Peters, L. J. 1986 (original 1959). *Writing Prose: Techniques and Purposes*, 6th edition. Oxford, England: Oxford University Press.

Kant, I. 1934 (1787 original). *The Critique of Pure Reason*, second edition. London: Dent. Translated by J. M. D. Meiklejohn.

Kaplan, S. J. October, 1982. "Cooperative Responses From a Portable Natural Language Query System." *Artificial Intelligence* 19(2):1982.

Kaplan, S. J. 1983. "Cooperative Responses From a Portable Natural Language Query System." *Computational Models of Discourse*, M. Brady and R. C. Berwick, editors. 167-207. MIT Press.

Kass, A. and D. Leake. March, 1987. "Types of Explanations." Yale University, CSD Research Report 523.

Kass, R. and T. Finin. September, 1988. "Modeling the User in Natural Language Systems." *Computational Linguistics: Special Issue on User Modeling* 14(3):5-22.

Katz, J. J. 1977. *Propositional Structures and Illocutionary Force*. New York: Crowell.

Kay, M. 1979. "Functional Grammar." Proceedings of the 5th Annual Meeting of the Berkley Linguistic Society, 1979.

Kay, M. 1980, October. "Algorithm Schemata and Data Structures in Syntactic Processing." Xerox Corporation CSL-80-12 [reprinted in Grosz, B. J., Sparck Jones, K. and Webber, B. L. 1986. *Readings in Natural Language Processing*, 35-70].

Kay, M. 1982. "Parsing in Functional Unification Grammar." in *Natural Language Parsing*, D. R. Dowty, L. Kartunnen and A. Zwicky, editors. 251-278. Cambridge, England. [reprinted in Grosz, B. J., Sparck Jones, K. and Webber, B. L. 1986. *Readings in Natural Language Processing*, 125-138].

Kempen., G. editor, 1987. *Natural Language Generation: New Results in Artificial Intelligence, Psychology, and Linguistics*, Dordrecht: Martinus Nijhoff. NATO ASI Series.

Kittredge, R., L. Iordanskaja and A. Polguère. 1988. "Multi-Lingual Text Generation and the Meaning-Text Theory." Second International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages, Carnegie Mellon University, Center for Machine Translation, 12-14 June, 1988.

Kittredge, R., A. Polguère and E. Goldberg. 25-29 August, 1986. "Synthesizing Weather Forecasts from Formatted Data." Proceedings of the 11th COLING, University of Bonn, West Germany, 1986. 563-565.

Kobsa, A. and W. Wahlster. editors. 1988, September. *Computational Linguistics: Special Issue on User Modelling* 14(3):1-78.

Kobsa, A. and W. Wahlster. editors. 1989. *User Models in Dialogue Systems*. Berlin: Springer.

Kukich, K. 1983. "Design of a Knowledge-Based Report Generator." Proceedings of the 21st Annual Meeting of the ACL, Cambridge, MA, 1983. 145-150.

Kukich, K. 1985a. "Feasibility of Automatic Natural Language Report Generation." 18th Annual Hawaii International Conference on Systems Sciences, University of Hawaii, Honolulu, 2-4 January, 1985.

Kukich, K. 1985b. "Explanation Structures in XSEL." Proceedings of the 23rd Annual Meeting of the ACL, Chicago.

Kukich, K. 1988. "Fluency in Natural Language Reports." *Natural Language Generation Systems*, McDonald, D. and L. Bolc, editors, Springer-Verlag: New York, 280-311.

Kukich, K., J. McDermott and T. Wang. 25 July 1985. "Explaining XSEL's Reasoning." Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA, draft.

Lakoff, G. P. 1972. "Structural Complexity in Fairy Tales." *The Study of Man* 1(1972):128-190.

Lehnert, W. G. 1981. "Plot Units and Narrative Summarization." *Cognitive Science* 4(1981):293-331.

Levine, J. M. 1989. "Taking Generation Seriously in a Natural Language Question Answering System." Extended Abstract from the Second European Workshop on Natural Language Generation, University of Edinburgh, Edinburgh, Scotland, 6-8 April, 1989.

Levine, J. M. 1990. "A Flexible Bidirectional Dialogue System." Proceedings of Eighth National Conference on Artificial Intelligence (AAAI-90), Boston, MA, July 29 - August 3, 1990. 964-969.

Levine, J. forthcoming. A Flexible Bidirectional Dialogue System. Computer Laboratory, University of Cambridge Ph.D. dissertation.

Levine, J. M. and L. Fedder. October, 1989. "The Theory and Implementation of a Bidirectional Question Answering System." University of Cambridge Computer Laboratory TR 182.

Levinson, S. 1983. *Pragmatics*. Cambridge University Press.

Lewis, D. 1972. "General Semantics" in D. Davidson and G. H. Harman, editors, *Semantics of Natural Language*. Dordrecht: Reidel.

Li, P., M. Evens and D. Hier. 1986. "Generating Medical Case Reports with the Linguistic String Parser." AAAI-86, Proceedings of Fifth Annual Conference on Artificial Intelligence (AAAI-86), Philadelphia, PA, August 11-15, 1986. 1069-1073.

Linde, C. and W. Labov. 1975. "Spatial Networks as a Site for the Study of Language and Thought." *Language* 51(1975):924-939.

Litman, D. L. 1986. "Linguistic Coherence: A Plan-Based Alternative." Proceedings of the 24th Meeting of the ACL, 1986. 215-223.

Litman, D. J. and J. F. Allen. 1987. "A Plan Recognition Model for Subdialogues in Conversations." *Cognitive Science* 11(1987):163-200.

Mann, W. C. 1983. "An Overview of the PENMAN text generation system." Proceedings of the National Conference on Artificial Intelligence, Washington, D. C., August, 1983. 261-265.

Mann, W. C., M. Bates, B. J. Grosz, D. P. McDonald, K. R. McKeown and W. R. Swartout. 1981. "Text Generation: The State of the Art and the Literature." USC/ISI TR-151/RR-81-101.

Mann, W. and C. Matthiessen. 1983. "Nigel: A Systemic Grammar for Text Generation." USC/ISI TR RR-83-105.

Mann, W. C. and J. A. Moore. January-March, 1981. "Computer Generation of Multi-paragraph English Text." *Computational Linguistics* 7(1):1981.

Mann, W. C. and S. A. Thompson. 1987. "Rhetorical Structure Theory: Description and Construction of Text Structures." *Natural Language Generation*, G. Kempen, editor, 85-95. Dordrecht: Martinus Nijhoff.

Mann, W. C. and S. A. Thompson. July, 1987. "Rhetorical Structure Theory: A Theory of Text Organization." Information Sciences Institute RS-87-190. Also appears in *Text* 8(3): 243-281.

Marcus, M. 1980. A Theory of Syntactic Recognition for Natural Language. Cambridge, MA: MIT Press.

Matthiessen, C. 1981. "A Grammar and Lexicon for a Text-production System." Proceedings of the 19th Annual Meeting of the ACL, Stanford, California, 1981. 49-56.

Matthiessen, C. November, 1984. "Choosing Tense in English." University of Southern California/Information Sciences Institute TR RR-84-143.

Matthiessen, C. 1987. "Notes on the Organization of the Environment of a Text Generation Grammar." *Natural Language Generation*, G. Kempen, editor. 251-278. Dordrecht: Martinus Nijhoff Publishers.

Mauldin, M. 1984. "Semantic Rule Based Text Generation." Proceedings of the 22nd Annual Meeting of the ACL, 376-380.

Maybury, M. T. May, 1986. Artificial Intelligence: Generalized Expert Systems. Fenwick Scholar Thesis, Department of Special Studies, College of the Holy Cross, Worcester, Massachusetts.

Maybury, M. T. 1987a, February. "A Natural Language Interface to An Expert System for Neuropsychological Diagnosis." Report for Computer Speech and Language Processing (CSLP) course at Cambridge University Engineering Department (CUED).

Maybury, M. T. 1987b, September. A Report Generator. Computer Speech and Language Processing M Phil Thesis, Engineering Department, Cambridge University, Cambridge, England.

Maybury, M. T. 1988a. "GENNY: A Knowledge Based Text Generation System." Proceedings of the RAIO-88 Conference on User-Oriented Content-Based Text and Image Handling, MIT, Cambridge, MA, March 21-24, 1988.

Maybury, M. T. 1988b. "Towards a Portable Natural Language Generator." Proceedings of the 8th International Workshop on Expert Systems and Their Applications (Avignon 88), Avignon, France, 30 May - 3 June, 1988. 413-426.

Maybury, M. T. 1988c. "A Computational Model of Explanation for a Tactical Mission Planner." Proceedings of the 1988 Symposium of Command and Control Research, Naval Post Graduate School and Monterey Resort Inn, Monterey, CA, 7-9 June, 1988. 382-386.

Maybury, M. T. 1988d. "Explanation Rhetoric: the Rhetorical Progression of Justifications." Proceedings of the AAAI-88 Workshop on Explanation (also in Proceedings of the AAAI-88 Workshop on Text Planning and Realization, August 22, 1988, pp 135-140), St. Paul, MN, August 22, 1988. 16-20.

Maybury, M. T. 1988e. "Experience with Relational Grammar: A Syntactic Formalism for Multilingual Generation." Research in progress note, Computational Intelligence 88, Universitá degli Studi di Milano, Italy, September 26-30, 1988.

Maybury, M. T. 1989a. "GENNY: A Knowledge-Based Text Generation System." *International Journal of Information Processing and Management* 25(2):137-150.

Maybury, M. T. 1989b. "Enhancing Explanation Coherence with Rhetorical Strategies." Proceedings of the Fourth Conference of the ECACL, Manchester, England, April 10-12, 1989. 168-173.

Maybury, M. T. 1989c. "Rhetorical Variance in Natural Language Descriptions." Proceedings of the Second Scandinavian Conference on Artificial Intelligence, Tampere, Finland, June 13-15, 1989.

Maybury, M. T. 1989d, August. "Knowledge Based Text Generation." RADC Technical Report 89-93, 100 pp.

Maybury, M. T. 1989e. "Design Issues in Natural Language Generators." Proceedings of the American Institute of Aeronautics and Astronautics Computers in Aerospace VII, Doubletree Hotel, Monterey, CA, 1989e, October 3-5. 126-130.

Maybury, M. T. 1990a. "Custom Explanations: Exploiting User Models to Plan Multisentential Text." Proceedings of the Second International Workshop on User Models, University of Honolulu, Hawaii, 30 March - 1 April, 1990.

Maybury, M. T. 1990b. "Classifying and Reacting to User Feedback to Guide Text Generation." Proceedings of the Explanation Workshop, Department of Computer Science, University of Manchester, 25-27 April, 1990.

Maybury, M. T. 1990c. "Using Discourse Focus, Temporal Focus, and Spatial Focus to Plan Multisentential Text." 5th International Workshop on Natural Language Generation, Linden Hall, Dawson, PA, 3-6 June, 1990.

Maybury, M. T. 1990d. "The Four Forms of Explanation Presentation: Description, Narration, Exposition, and Argument." AAAI-90 Workshop on Explanation, Boston Hynes Center, Boston, MA, Monday, July 30, 1990.

Maybury, M. T. and C. Weiss. 1987. "The NEUROPSYCHOLOGIST Expert System Prototype." Intelligence Integration, Proceedings of the Canadian Information Processing Society, Edmonton, Alberta, Canada, 16-19 November, 198. 125-130.

Mays, E. 1980. "Failures in Natural Language Systems: Applications to Data Base Query Systems." Proceedings of the First National Conference on Artificial Intelligence (AAAI-80), Stanford, CA, August, 1980.

McCalla, G. I., L. Reid and P. F. Schneider. 1982. "Plan creation, plan execution and knowledge acquisition in a dynamic microworld." *International Journal of Man-machine Studies* 16(1982):89-112.

McCoy, K. F. 1985a. "The Role of Perspective in Responding to Property Misconceptions." Proceedings of the Ninth IJCAI, Los Angeles, CA, 18-23 August, 1985. 791-793.

McCoy, K. F. December, 1985b. Correcting Object-Related Misconceptions. Ph.D. dissertation, University of Pennsylvania TR MS-CIS-85-57, Philadelphia, PA.

McCoy, K. F. 1986. Automatic Enhancement of a Data Base Knowledge Representation used for Natural Language Generation. MS Thesis, University of Pennsylvania, Philadelphia, PA.

McCoy, K. F. September 1988. "Reasoning on a Highlighted User Model to Respond to Misconceptions." *Computational Linguistics: Special Issue on User Modeling* 14(3):52-63.

McCoy, K. F. and J. Cheng. 1991. "Focus of Attention: Constraining what can be said next." In *Natural Language Generation in Artificial Intelligence and Computational Linguistics,* Paris, C. L., W. R. Swartout, and W. C. Mann (eds).

McDermott, D. 1978. "Planning and Acting." *Cognitive Science* 2(1978):71-109.

McDonald, D. D. 1975. "Preliminary Report on a Program for Generating Natural Language." Advanced Papers of the Fourth IJCAI, Tbilisi, Georgia, USSR, September 3-8, 1975. 401-405.

McDonald, D. D. 1980. Natural Language Production as a Process of Decision Making under Constraints. unpublished Ph.D. dissertation, MIT Department of Electrical Engineering and Computer Science, Cambridge, MA.

McDonald, D. D. 1981. "Language Production: The Source of the Dictionary." Proceedings of the 19th Annual Meeting of the ACL, 57-62.

McDonald, D. D. 1983. "Language Generation as a Computational Problem: an Introduction." in Brady and Berwick, editors, *Computational Models of Discourse*, 1983, pp. 209-264. Originally published as University of Massachusetts at Amherst, COINS TR-81-83, December, 1983.

McDonald, D. D. 1986. "Description Directed Control: Its Implications for Natural Language Generation." *Readings in Natural Language Processing*, B. J. Grosz, K. Sparck Jones and B. L. Webber, editors. 519-537. Morgan Kaufmann. Originally appeared in *Computers and Mathematics* 9(1), 1983, 111-130, Pergamon Press.

McDonald, D. D. 1987. "Natural Language Generation: Complexities and Techniques." *Machine Translation:Theoretical and Methodological Issues*, S. Nirenburg, editor. Cambridge University Press.

McDonald, D. D. and M. W. Meteer. 1987. Notes from the Rome Air Development Center Natural Language Generation Workshop, Minnowbrook Conference Center, Blue Mountain Lake, NY, 20-31 July, 1987.

McDonald, D. D. and M. W. Meteer. 1988. "From Water to Wine: Generating Natural Language Text from Today's Applications Programs." Second ACL Conference on Applied Natural Language Processing, Austin, TX, February 9-12, 1988.

McDonald, D. D. and J. Pustejovsky. 1985a. "A Computational Theory of Prose Style for Natural Language Generation." Proceedings of the Second Conference of ECACL, 27-29 March, 1985. 187-193.

McDonald, D. D. and J. Pustejovsky. 1985b. "TAG's as a Grammatical Formalism for Generation." Proceedings of the 23rd Annual Meeting of the ACL, 94-103.

McDonald, D. D. and J. Pustejovsky. 1985c. "Description-Directed Natural Language Generation." Proceedings of the Ninth IJCAI, Los Angeles, CA, 18-23 August, 1985. 799-805.

McKeown, K. R. 1979. "Paraphrasing Using Given and New Information in a Question-Answering System." Proceedings of the 17th Annual Meeting of ACL, August, 1979. 67-72.

McKeown, K. R. 1982. Generating Natural Language Text in Response to Questions About Database Structure. University of Pennsylvania Ph.D. dissertation, TR MS-CIS-82-05, Pennsylvania.

McKeown, K. R. 1985a. *Text Generation.* England: Cambridge University Press.

McKeown, K. R. 1985b. "Discourse Strategies for Generating Natural-Language Text." *Artificial Intelligence* 27(1985):1-41.

McKeown, K. R. 1989. Presentation at Second European Workshop on Natural Language Generation, University of Edinburgh, Edinburgh, Scotland, 6-8 April, 1989.

McKeown, K. R. and C. L. Paris. 1987. "Functional Unification Grammar Revisited." Proceedings of the 25th Annual Meeting of the ACL, Stanford, CA, July, 1987.

McKeown, K. R. and W. Swartout. 1987. "Language Generation and Explanation." *Annual Review of Computer Science* 2:401-449. [reprinted in M. Zock and G. Sabah, editors, 1988. *Advances in Natural Language Generation*, Pinter Publishers: Inc, London, England, 1-54].

McKeown, K. R., M. Wish and K. Matthews. 1985. "Tailoring Explanations for the User." Proceedings of the Ninth IJCAI, Los Angeles, CA, 794-798.

Meehan, J. R. 1976. The Metanovel: Writing Stories by Computer. Ph.D. dissertation, Yale University TR 74, New Haven, CT.

Meehan, J. R. 1977. "TALE-SPIN, an Interactive Program that Writes Stories." Proceedings of the Fifth IJCAI, August, 1977. 91-98.

Mellish, C. and R. Evans. December, 1989. "Natural Language Generation from Plans." *Computational Linguistics* 15(4):233-249.

Meteer, M. W. 1988a. "The Implications of Revisions for Natural Language Generation." Fourth International Workshop on Natural Language Generation, July, 1988. extended abstract.

Meteer, M. W. 1988b. "Defining a Vocabulary for Text Planning." Proceedings of the AAAI Workshop on Text Planning and Realization, Saint Paul, MN, August, 1988. 115-122.

Meteer, M. W. July, 1989. "The SPOKESMAN Natural Language Generation System." BBN TR 7090.

Meteer, M. W., D. D. McDonald, S. D. Anderson, D. Forster, L. S. Gay, A. K. Huettner and P. Sibon. 1987, September. "MUMBLE 86: Design and Implementation." University of Massachusetts COINS TR 87-87.

Meteer, M. W. and D. D. McDonald. "The Orchestration Process in Text Planning for Generation." draft.

Meyer, B. J. 1975. "The Organization of Prose and its Effects on Memory." North-Holland Studies in Theoretical Poetics, 1, Amsterdam: North-Holland.

Miezitis, M. A. 1988, September. Generating Lexical Options by Matching in a Knowledge Base. University of Toronto Computer Systems Research Institute TR 217, Toronto, Canada.

Minsky, M., editor. 1968. "Semantic Information Processing." *Semantic Memory*, M. R. Quillian. Cambridge, MA: MIT Press.

Minsky, M. 1975. "A Framework for Representing Knowledge." *The Psychology of Computer Vision*, P. H. Winston, editor. New York: McGraw-Hill.

Moens, M. and M. Steedman. June, 1988. "Temporal Ontology and Temporal Reference." *Computational Linguistics: Special Issue on Tense and Aspect* 14(2):15-28.

Montague, J. L. 1974. *Formal Philosophy: Selected Papers*. R. H. Thomson, editor. New Haven, CT: Yale University Press.

Moore, J. D. November, 1989. A Reactive Approach to Explanation in Expert and Advice-Giving Systems. Ph.D. dissertation, University of California, Los Angeles, CA.

Moore, J. D. and C. L. Paris. 1988. "Constructing Coherent Text Using Rhetorical Relations." Tenth Annual Conference of the Cognitive Science Society, Montreal, Quebec, Canada, August 17-19, 1988.

Moore, J. D. and C. L. Paris. 1989. "Planning Text for Advisory Dialogues." Proceedings of Twenty-Seventh Annual Meeting of the ACL, Vancouver, Canada, June, 1989.

Moore, J. D. and W. R. Swartout. December 9, 1987. "Explanation in Expert Systems: A Survey." Proceedings of the First International Symposium on Expert Systems in Business, Finance, and Accounting, Newport Beach, CA, 29 September - 1 October, 1988.

Moore, J. D. and W. R. Swartout. 1988. "Planning and Reacting." Proceedings of the AAAI Workshop on Text Planning and Realization, St. Paul, MN, August 25, 1988. 30-38.

Moore, J. D. and W. R. Swartout. 1989. "A Reactive Approach to Explanation." Proceedings of the Eleventh IJCAI, Detroit, MI, 20-25 August, 1989. 1504-1510. [See also Moore, J. D. and W. R.

Swartout. 1988b. "A Reactive Approach to Explanation." Proceedings of the AAAI Workshop on Explanation, St Paul, MN, August 25, 1988. 91-94.]

Moore, J. D. and W. R. Swartout. 1991. "A Reactive Approach to Explanation." in *Natural Language Generation in Artificial Intelligence and Computational Linguistics*. Paris, C. L., W. R. Swartout, and W. C. Mann (eds).

Moore, R. C. 1980. "Reasoning About Knowledge and Action." SRI International AI Center TR 191.

Moravesik, E. A. and J. R. Wirth, editors. 1980. Syntax and Semantics 13: Current Approaches to Syntax, New York: Academic Press.

Nakhimovsky, A. June, 1988. "Aspect, Aspectual Class, and the Temporal Structure of Narrative." *Computational Linguistics: Special Issue on Tense and Aspect* 14(2):29-43.

Neal, J. 1989. "Coordination of Multi-Modal Input and Output." IJCAI-89 Workshop: A New Generation of Intelligent Interfaces, Detroit, MI, August 22, 1989. 92-95.

Neches, R., J. Moore and W. Swartout. November 1985. "Enhanced Maintenance and Explanation of Expert Systems Through Explicit Models of Their Development," *IEEE Transactions on Software Engineering* SE-11(11):1337-1351.

Nirenburg, S., editor. 1987. *Machine Translation: Theoretical and Methodological Issues*, Studies in Natural Language Processing, Cambridge, England: Cambridge University Press.

Palmer, M., T. Finin and S. M. Walter. December, 1989. "Workshop on the Evaluation of Natural Language Processing Systems." Rome Air Development Center TR 89-302.

Paris, C. L. 1987a. "Combining Discourse Strategies to Generate Descriptions to Users Along a Naive/Expert Spectrum." Proceedings of the Tenth IJCAI, Milan, Italy, 23-28 August, 1987. 626-632.

Paris, C. L. 1987b. The Use of Explicit User Models in Text Generation: Tailoring to a User's Level of Expertise. Ph.D. dissertation, Columbia University, NY.

Paris, C. L. September, 1988. "Tailoring Object Descriptions to a User's Level of Expertise." *Computational Linguistics: Special Issue on User Modeling* 14(3):64-78.

Paris, C. L., M. M. Wick and W. Thompson. 1988. "The Line of Reasoning Versus the Line of Explanation." Proceedings of the AAAI-88 Workshop on Explanation, Minneapolis-St Paul, MN, August 22, 1988.

Paris, C. L. and K. McKeown. 1987. "Discourse Strategies for Describing Complex Physical Objects." *Natural Language Generation*, G. Kempen, editor, 97-116.

Paris, C. L., W. R. Swartout, and W. C. Mann (eds). 1991. *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, Kluwer: Norwell, MA. [Proceedings of the Fourth International Workshop on Natural Language Generation, Catalina Island, Los Angeles, CA, July, 1988.]

Parisi, D. 1985. "GEMS: A Model of Sentence Production." Proceedings of the 2nd Annual Conference of the ECACL, 258-262.

Passonneau, R. J. June, 1988. "A Computational Model of the Semantics of Tense and Aspect." *Computational Linguistics: Special Issue on Tense and Aspect* 14(2):44-60.

Patten, T. editor. 1988. *Systemic Text Generation as Problem Solving*, Studies in Natural Language Processing, Cambridge, England: Cambridge University Press.

Patten, T. and G. Ritchie. 1987. "A Formal Model of Systemic Grammar." *Natural Language Generation,* G. Kempen. 279-299. Dordrecht: Martinus Nijhoff Publishers.

Pemberton, L. 1984. Story Structure: A Narrative Grammar of Nine Chansons de Geste of the Guillaume d'Orange Cycle. Ph.D. dissertation. University of Toronto, Canada.

Pemberton, L. 1989. "A Modular Approach to Story Generation." Proceedings of the Fourth Conference of the ECACL, University of Manchester, Institute of Science and Technology, 10-12 April, 1989. 217-224.

Perlmutter, D. 1980. "Relational Grammar." *Syntax and Semantics 13*, Moravesik and Wirth, editors. 195-230.

Perlmutter, D. and Rosen, C. J. G. 1984. *Studies in Relational Grammar 2.* Chicago: University of Chicago Press.

Perlmutter D. M. and P. M. Postal. 1977. "Toward a Universal Characterization of Passivization." Whistler, K., R. D. J. van Valin, C. Chiarello, J. J. Jaeger, M. Petruck, H. Thompson, R. Jarkin and A. Woodbury, Proceedings of the Third Annual Meeting of the Berkley Linguistics Society, Berkley, CA, 1977.

Perlmutter, D. and Soames, S. 1979. *Syntactic Argumentation and the Structure of English.* Berkley: University of California Press.

Perrault, C. R. and P. R. Cohen. 1977. "Overview of "Planning Speech Acts"." Proceedings of the Fifth IJCAI, MIT, Cambridge, MA, 22-25 August, 1977.

Pickett, N. and Laster, A. 1988. Technical English: Writing, Reading, and Speaking, fifth edition. New York: Harper & Row.

Plato. 1914 (original around 370 B.C). *Phaedrus*, in Works, vol I. trans. H. N. Fowler, Cambridge, MA: Loeb Classical Library series.

Plato. 1925 (original probably shortly after death of Socrates in 399 B.C.). *Gorgias*, in Works, vol V. trans. W. R. M. Lamb, Cambridge, MA: Loeb Classical Library series.

Polguère, A., L. Bourbeau and R. Kittredge. 1987. "RAREAS-2: Bilingual Synthesis of Arctic Marine Forecasts." TR, Odyssée Recherches Appliquées Inc. Montréal, Canada.

Pollack, M. 1986. Inferring Domain Plans in Question-answering. University of Pennsylvania Ph.D. dissertation, Philadelphia, PA.

Power, R. 1974. A Computer Model of Conversation. unpublished Ph.D. dissertation, University of Edinburgh, Edinburgh, Scotland.

Power, R. 1979. "The Organisation of Purposeful Dialogues." *Linguistics* 17(1979):107-152.

Propp, V. 1969. Morphology of The Folk Tale. Austin: University of Texas Press.

Pullum, G. K. 1977. "Word Order Universals and Grammatical Relations." Syntax and Semantics 8: Grammatical Relations, P. Cole and J. M. Sadock, editors. New York: Academic Press.

Pulman, S. G. Michaelmas and Lent Term, 1986-87. "Syntax and Parsing, Semantics and Inference, and Discourse Processing." lecture notes for Computer Speech and Language Processing Course, Cambridge University Engineering Department.

Quilici, A. 1989. "The Correction Machine: Formulating Explanations For User Misconceptions." Proceedings of the Eleventh IJCAI, Detroit, MI, 20-25 August, 1989. 550-555.

Quilici, A., M. Dyer and M. Flowers. September 1988. "Recognizing and Responding to Plan-Oriented Misconceptions." *Computational Linguistics: Special Issue on User Modeling* 14(3):38-51.

Quillian, M. R. 1966. *Semantic Memory*. Carnegie Institute of Carnegie-Mellon University [published as BBN Report 2, Project 8668, 1966].

Quirk, R. and Greenbaum, S. 1973. *A Concise Grammar of Contemporary Eng*lish. New York: Harcourt, Brace and Jovanovich, Inc.

Rankin, I. 1989. "Deep Generation of a Critique." Proceedings of the Second European Workshop on Natural Language Generation, Edinburgh, Scotland, 6-8 April, 1989.

Reichenbach, H. 1947. *The Elements of Symbolic Logic*. London: Macmillan.

Reichman, R. 1981a. "Modeling Informal Debates." Proceedings of the Seventh IJCAI, Vancouver, B. C., Canada, 24-28 August, 1981.

Reichman, R. 1981b. Plain Speaking: A Theory and Grammar of Spontaneous Discourse. Division of Applied Sciences, Harvard University Ph.D. dissertation, also as BBN TR 4681, Cambridge, MA.

Reichman, R. 1985. *Getting Computers to Talk Like You and Me*. Cambridge, MA: MIT Press.

Reiter, E. 1990. "Generating Descriptions that Exploit a User's Domain Knowledge." In *Current Research in Natural Language Generation*, R. Dale., C. Mellish and M. Zock, editors.

Ritchie, G. 1986. "The Computational Complexity of Sentence Derivation in Functional Unification Grammar." Proceedings of the Eleventh COLING, Bonn, West Germany, August, 1986. 584-586.

Roberts, R. and I. Goldstein. September, 1977. "The FRL Manual." MIT AI Lab Memo 409.

Rosch, E. 1973. "Natural Categories." *Cognitive Psychology* 4(1973):328-350.

Rosch, E. 1977. "Classification of Real World Objects: Origins and Representations in Cognition." Thinking: Readings in Cognitive Science, P. N. Johnson-Laird and P. C. Wason, editors. Cambridge: Cambridge University Press.

Rosch, E., C. B. Mervis, W. D. Gray, D. M. Johnson and P. Boyes-Braem. 1976. "Basic Objects in Natural Categories." *Cognitive Psychology* 8(1976):382-439.

Rubinoff, R. 1986. "Adapting MUMBLE: Experience with Natural Language Generation." Proceedings of the Fifth National Conference on Artificial Intelligence (AAAI-86), Philadelphia, PA, August, 1986. Engineering, 1063-1068.

Rumelhart, D. E. 1975. "Notes on a schema for stories." *Representation and Understanding: Studies in Cognitive Science*, D. G. Bobrow and M. Collins, editors. New York: Academic Press.

Sacerdoti, E. D. 1973. "Planning in a Hierarchy of Abstraction Spaces." Proceedings of the Third IJCAI, Palo Alto, CA.

Sacerdoti, E. D. 1975. "The Non-Linear Nature of Plans." Proceedings of the Fourth IJCAI, Tbilisi, USSR.

Sacerdoti, E. D. 1977. A Structure for Plans and Behavior. New York: Elsevier North-Holland. [Originally Stanford Research Institute TR TN-109, 1975].

Sadock, J. M. 1988. "Speech Act Distinctions in Grammar." *Linguistics: The Cambridge Survey. Vol II, Linguistic Theory: Extensions and Implications*, F. J. Newmeyer, editor. Cambridge University Press.

Sager, N. 1981. *Natural Language Information Processing: A Computer Grammar of English.* Reading, MA: Addison-Wesley.

Sanford, A. J. and Garrod, S. C. 1981. *Understanding Written Language*. Chichester: Wiley.

Sarner, M. and S. Carberry. 1990. "Tailoring Explanations Using a Multifaceted User Model." Second International Workshop on User Modelling, Honolulu, HI, 30 March - 1 April, 1990.

Schank, R. C. 1975. *Conceptual Information Processing*. Amsterdam: North-Holland/Elsevier.

Schank, R. C. 1984a, March. "The Explanation Game." Yale University CSD, Research Report 307.

Schank, R. C. 1984b, September. "Explanation: A First Pass." Yale University CSD, Research Report 330.

Schank, R. C. 1986. *Explanation Patterns: Understanding Mechanically and Creatively*. Hillsdale, NJ: Lawrence Erlbaum.

Schank, R. C. and Abelson, R. 1977. *Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Schank, R. C. and C. Riesbeck. July, 1985. "Explanation: A Second Pass." Yale University CSD, Research Report 384.

Schubert, L. K. and C. H. Hwang. 1989. "An Episodic Knowledge Representation for Narrative Texts." Proceedings First Conference on Principles of Knowledge Representation and Reasoning, Toronto, Ontario, Canada, 15-18 May, 1989.

Schuster, E., D. Chin, R. Cohen, A. Kobsa, K. Morik, K. Sparck Jones and W. Wahlster. September, 1988. "Discussion Section on the Relationship between User Models and Discourse Models." *Computational Linguistics: Special Issue on User Modeling* 14(3):79-103.

Scott, A. F. 1938. *Meaning and Style*. London: Macmillan.

Searle, J. R. 1969. *Speech Acts*. Cambridge University Press.

Searle, J. R. 1975. "Indirect Speech Acts." *Syntax and Semantics 3: Speech Acts*, P. Cole and J. L. Morgan, editors. 59-82. New York: Academic Press.

Sells, P. 1985. *Lectures on Contemporary Syntactic Theories*. Center for Study of Language and Information, Stanford, CA.

Shadbolt, N. R. 1984. Constituting Reference in Natural Language: The Problem of Referential Opacity. Edinburgh University Ph. D. dissertation, Edinburgh, Scotland.

Shieber, S. M. 1988. "A Uniform Architecture for Parsing and Generation." Proceedings of the 12th COLING, Budapest, Hungary, August, 1988, 614-619.

Shortliffe, E. 1976. *Computer-based Medical Consultations: MYCIN*. Elsevier North Holland Inc.

Shubert, L. K. and C. H. Hwong. 1989. "Episodic Knowledge Representation for Narrative Texts." Brachman, R. J., H. J. Levesque and R. Reiter, Proceedings of KR-89: First International Conference on Principles of Knowledge Representation and Reasoning, Toronto, Ontario, Canada, May 15-18, 1989. 444-458.

Sidner, C. L. 1979. Toward a Computational Theory of Definite Anaphora Comprehension in English Discourse. Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA.

Sidner, C. L. 1983. "Focusing in the Comprehension of Definite Anaphora." *Computational Models of Discourse*, M. Brady and R. Berwick, editors. 267-330. Cambridge, MA: MIT Press.

Simmons, R. and D. Chester. August 8, 1982. "Relating Sentences and Semantic Networks with Procedural Logic." *Communications of the ACM* 25(1982):527-547.

Simmons, R. and J. Slocum. 1972. "Generating English Discourse from Semantic Networks." *Communications of the ACM* 15(10).

Slocum, J. 1985. "A Survey of Machine Translation: its History, Current Status, and Future Prospects." *Computational Linguistics* 11(1).

Sparck Jones, K. December, 1984. "User Models and Expert Systems." University of Cambridge Computer Laboratory TR 61.

Sparck Jones, K. 1985. "Compound Noun Interpretation Problems." *Computer Speech Processing*, F. Fallside and W. A. Woods, editors. 363-382. Englewood Cliffs, NJ: Prentice Hall.

Sparck Jones, K. 1988. "User Models, Discourse Models, and Some Others." *Computational Linguistics* 14(3):98-100.

Sparck Jones, K. 1989. "Realism about User Modelling." *User Models in Dialogue Systems*, A. Kobsa and W. Wahlster, editors. Berlin: Springer.

Sparck Jones, K. 1990. "Tailoring Output to the User: What Does User Modelling in Generation Mean?" In *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, Paris C., W. Swartout and W. Mann., editors, 201-226.

Sparck Jones, K. and B. K. Boguraev. January-June, 1987. "A Note on a Study of Cases." *Computational Linguistics* 13(1-2):65-68.

Sparck Jones, K. and J. I. Tait. 1984. "Linguistically Motivated Descriptive Term Selection." Proceedings of the 10th COLING, Stanford, CA, 1984.

Stefik, M. 1981a. "Planning with constraints: MOLGEN Part I." *Artificial Intelligence* 16(2).

Stefik, M. 1981b. "Planning and Metaplanning MOLGEN: Part II." *Artificial Intelligence* 16(3).

Stevens, A. and C. Steinberg. March 5, 1981. "A Typology of Explanation and its Application to Intelligent Computer Aided Instruction." BBN TR-4626.

Street, M. editor, 1986. *La Cucina: The Complete Book of Italian Cooking*. Milan: Del Drago.

Strunk, W. J. and White, E. B. 1979 (original 1959). *The Elements of Style*, 3rd edition. New York: Macmillan.

Suthers, D. 1988a. "Providing Multiple Views of Reasoning for Explanation." International Conference on Intelligent Tutoring Systems, Montreal, Canada, June, 1988.

Suthers, D. 1988b. "Providing Multiple Views for Explanation." Proceedings of the AAAI-88 Workshop on Explanation, Minneapolis-St Paul, MN, August 22, 1988.

Suthers, D. D. 1989. "Perspectives in Explanation." University of Massachusetts Department of Computer and Information Science (COINS) TR 89-24.

Swartout, W. R. February, 1977. A Digitalis Therapy Advisor with Explanations. Masters thesis, MIT TR-176, Cambridge, MA.

Swartout, W. R. 1981a, January. Producing Explanations and Justifications of Expert Consulting Programs. Ph.D. dissertation, Laboratory for Computer Science, Massachusetts Institute of Technology TR-251, Cambridge, MA.

Swartout, W. R. 1981b. "Explaining and Justifying Expert Consulting Programs." Proceedings of Seventh IJCAI, University of British Columbia, 24-28 August 1981. 815-822.

Swartout, W. R. 1983a, September. "XPLAIN: A System for Creating and Explaining Expert Consulting Programs." *Artificial Intelligence* 21(3):285-325, Also ISI TS-83-4.

Swartout, W. R. 1983b. "Explainable Expert Systems." presented at MEDCOMP-83, October, 1983.

Swartout, W. R. 1986. "Knowledge Needed for Expert System Explanation." *Future Computing Systems* 1(2):99-114.

Swartout, W. R. and S. W. Smoliar. August, 1987. "On Making Expert Systems more like Experts." *Expert Systems* 3196-207.

Swartout, W. R. and S. W. Smoliar. 1988. "Explaining the Link Between Causal Reasoning and Expert Behavior." in *Topics in Medical and Artificial Intelligence*, P. L. Miller, editor. Springer-Verlag.

Sykes, J. B., editor. 1984. *Concise Oxford Dictionary of Current English*, Oxford University Press.

Szolovits, P., L. B. Hawkinson and W. A. Martin. 1977. "An Overview of OWL, A Language for Knowledge Representation." MIT Laboratory for Computer Science TM-86.

Tait, J. 1982. Automatic Summarising of English Texts. Computer Laboratory TR 47, Cambridge University.

Tait, J. I. 1985. "An English Generator for a Case-Labeled Dependency Representation." Proceedings of the Second Annual Conference of the ECACL, 194-197.

Tate, A. December, 1974. "INTERPLAN: A Plan Generation System that can deal with Interactions Between Goals." University of Edinburgh Machine Intelligence Research Unit Memo MIP-1-109, Edinburgh, Scotland.

Tedeschi, P. and Zaenen, A. 1981. *Syntax and Semantics. Volume 14: Tense and Aspect*. New York, NY: Academic Press.

Toulmin, S. 1958. *The Uses of Argument*. Cambridge, England: Cambridge University Press.

Toulmin S., R. Rieke, and A. Janik. 1979. *An Introduction to Reasoning*. New York: Macmillan.

Tulding E. and W. Donaldson, editors. 1972. "Organization of Memory." *How to Make a Language User*, Collins A. M. and Quillian M. R. Academic Press, NY.

Tversky, A. 1977. "Features of Similarity." *Psychological Review* 84(1977):327-352.

van Beek, P. A. September, 1986. "Model for User-Specific Explanation from Expert Systems." University of Waterloo TR CS-86-42, Ontario, Canada.

van Dijk, T. A. 1977. *Text and Context*. London: Longman.

van Dijk, T. A. and W. Kintsch. 1983. *Strategies of Discourse Comprehension.* New York: Academic Press.

Verdon, R. 1985. *The Enlightened Cuisine.* NY: Macmillan.

Wahlster, W., A. Jameson and W. Hoeppner. 1978. "Glancing, Referring and Explaining in the Dialogue System HAM-RPM." *Computational Linguistics* (microfiche 77):53-67.

Wahlster, W., E. André, M. Hecking and T. Rist. May, 1989. "Knowledge-based Presentation of Information (WIP)." Project Proposal.

Walter, S. M. 1986. "Natural Language Processing: A Tutorial." Rome Air Development Center TR 86-110.

Walter, S. and C. Kalish. October, 1985. "An extensible Natural Language System." USAF Electronic Security Division, Hanscom AFB, MA/MITRE Bedford Technical Objectives and Plans Report, Project 7590.

Webber, B. L. 1987a. "The Interpretation of Tense in Discourse." Proceedings of the 25th Annual Meeting of the ACL, Stanford University, Palo Alto, CA, 1987. 147-154.

Webber, B. L. 1987b. "Question Answering." *Encyclopedia of Artificial Intelligence*, S. C. Shapiro, editor. 814-822. John Wiley.

Webber, B. L. 1988a. "Tense as Discourse Anaphor." *Computational Linguistics* 14(2):61-73.

Webber, B. L. 1988b. "Discourse Deixis: Reference to Discourse Segments." Proceedings of the 26th Annual Meeting of the ACL, SUNY Buffalo, 1988, 7-10 June. 113-122.

Webber, B. L. in preparation. "Discourse Canon." Presented at the Mohonk DARPA workshop, May, 1988.

Weiner, J. 1980. "BLAH, A System Which Explains its Reasoning." *Artificial Intelligence* 15:19-48.

Weischedel, R. M. 1989. "Knowledge Acquisition for Natural Language Processors." RADC/IRDP Natural Language Processing Workshop, Rome Air Development Center, Griffiss AFB, NY, 27-27 September, 1989.

Weizenbaum, J. 1966. "ELIZA - A Computer Program for the Study of Natural Language Communication Between Man and Machine." *Communications of the ACM* 9(1):36-45.

Wick, M. R., C. L. Paris, W. R. Swartout and W. B. Thompson, 1988. Proceedings of the AAAI-88 Workshop on Explanation, Saint Paul, MN, August, 1988.

Wick, M. R., W. B. Thompson and J. R. Slagle. March, 1988. "Knowledge Based Explanation." University of Minnesota Institute of Technology TR-88-24.

Wilensky, R. 1981. "Meta-Planning: Representing and using Knowledge about Planning, Problem Solving, and Natural Language Understanding." *Cognitive Science* 5(1981)197-223.

Wilensky, R. 1983. *Planning and Understanding.* Reading, MA: Addison-Wesley.

Wilensky, R. 1983. "Story Grammars versus Story Points." *Behavioral and Brain Sciences* 6(1983):579-623.

Wilensky, R., Y. Arens and D. Chin. 1984. "Talking to UNIX in English: An Overview of Unix Consultant." *Communications of the Association for Computing Machinery* 27(6).

Wilensky, R., D. N. Chin, M. Luria, J. Martin, J. Mayfield and D. Wu. 1988, December. "The Berkeley UNIX Consultant Project." *Computational Linguistics* 14(4):35-84.

Williams, W. 1893. *Composition and Rhetoric.* Boston, MA: D. C. Heath and Company.

Winograd, T. 1971. "A Computer Program for Understanding Natural Language." Massachusetts Institute of Technology TR-17.

Winograd, T. 1972. "Understanding Natural Language." from *Cognitive Psychology* 3(1). Orlando, Florida: Academic Press.

Winograd, T. 1983. *Language as a Cognitive Process, Volume 1: Syntax.* Reading, MA: Addison-Wesley.

Winograd, T. and Flores, F. 1987. *Understanding Computers and Cognition: A New Foundation for Design.* Reading, MA: Addison-Wesley and Co.

Wolz, U. 1990. "The Impact of User Modeling on Text Generation in Task-Centered Settings." Second International Workshop on User Modelling, Honolulu, Hawaii, 30 March - 1 April, 1990.

Woods, W. A. October, 1970. "Transition Network Grammar for Natural Language Analysis." *Communications of the ACM* 13(10):591-606.

# Author Index

# Subject Index