Lexical semantics and knowledge representation in multilingual sentence generation

by

Manfred Stede

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto

Abstract

Lexical semantics and knowledge representation in multilingual sentence generation

Manfred Stede
Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto
1996

This thesis develops a new approach to automatic language generation that focuses on the need to produce a range of different *paraphrases* from the same input representation. One novelty of the system is its solidly grounding representations of word meaning in a background knowledge base, which enables the production of paraphrases stemming from certain inferences, rather than from purely lexical relationships alone.

The system is designed in such a way that the paraphrasing mechanism extends naturally to a *multilingual* generator; specifically, we will be concerned with producing English and German sentences. The focus of the system is on *lexical* paraphrases, and one of the contributions of the thesis is in identifying, analyzing and extending relevant linguistic research so that it can be used to handle the problems of lexical semantics in a language generation system. The *lexical entries* are more complex than in previous generators, and they separate the various aspects of word meaning, so that different ways of paraphrasing can be systematically related to the different motivations for saying a sentence in a particular way. One result of accounting for lexical semantics in this fashion is a formalization of a number of *verb alternations*, for which a generative treatment is given.

While the actual *choice* of one paraphrase as the best-suited utterance in a given situation is not a focal point of the thesis, two dimensions of *preferring* a variant of a sentence are discussed: that of assigning *salience* to the different elements of the sentence, and that of *connotational* or *stylistic* features of the utterance. These dimensions are integrated into the system, and it can thus determine a preferred paraphrase from a set of alternatives.

To demonstrate the feasibility of the approach, the proposed generation architecture has been implemented as a protoype, along with a domain model that serves as the background knowledge base for specifying the input to the generator. A range of generated examples is presented to show the functionality of the system.

Acknowledgements

Probably, the origin of this thesis was a bit unusual. The first half of the project was executed in Toronto, and the second half off-campus, at the Research Center for Applied Knowledge Processing (FAW) at Ulm, Germany, with the occasional trip back and forth. The whole process took a little longer than I had originally anticipated; I do suspect that the latter is, at least in part, a consequence of the former.

Thanks to my supervisor, Graeme Hirst, who directed me on the entire journey, short-distance and long-distance alike. I can't quite remember how many versions of my work he has carefully read and commented upon. Besides his competence in computational linguistics, I'd recommend him as a teacher of English writing, and as an expert on nuances of lexical meaning.

Thanks also to the other members of my thesis committee: Michael Cummings, Chrysanne DiMarco, Hector Levesque, Alberto Mendelzon, and Jeff Siskind. While there was not always agreement on how to write a thesis, they did make me think about how to present my case more clearly.

Thanks to Dietmar Rösner, whom I met—somewhat by coincidence, indeed—at my first internship at FAW Ulm, not knowing at the time what was going to follow from that 3-months trip to Germany. He brought the whole topic of language generation to my attention, and my specific thesis topic ultimately developed from my work with him on the TECHDOC project. The hours we spent discussing the problems of multilingual text generation significantly shaped my sense of how these things should be done.

Thanks to Brigitte Grote, who read everything beforehand and recommended many improvements. She also helped with drawing figures in the critical last weeks when time was short. But far more than that, she helped me to see things in proportion, thesis and "the rest of life", and she was always there when I needed to be pushed forward. (Unless the Atlantic Ocean was temporarily between us.)

Thanks to Phil Edmonds, Nils Lenke, Mick O'Donnell, and Leo Wanner, who, at various stages, read parts of earlier versions of this thesis and gave me valuable feedback.

Thanks for financial support to the University of Toronto, and to FAW Ulm for a generous Ph.D. scholarship.

Last, but certainly not least: Thanks to my parents for making this education possible.

Contents

List of Figures

Chapter 1

Introduction

1.1 Natural language generation

What exactly is the difference between *watching* and *looking*, and is it the same as that between *hearing* and *listening*? How does a translator deal with the fact that French distinguishes between *savoir* and *connaître*, and German between *wissen* and *kennen*, where English has only one word, *to know*, for both? And how can it be explained that English and German reverse the assignment of content to verb and adverb in sentence pairs like *Tom likes to swim / Tom schwimmt gern* ('Tom swims likingly')? In linguistics, questions of this kind are examined under the heading *lexical semantics*: the study of the meaning of individual words, and of the relations between different senses and between similar words. One fruitful and illuminating means of studying word meaning is *contrastive* studies, where similar words in different languages are compared with respect to their syntax and meaning. Here, one keeps running into cases where on first sight two words appear to mean the same, and any dictionary lists them as translations, yet each has subtle shades of meaning causing them to differ in certain situations—in which they should *not* be used as translation-equivalent. For instance, for the German word *Sympathie* dictionaries give the straightforward translation *sympathy*; but since the English word is ambiguous, it is a fallacy to translate *für jemanden Sympathie haben* literally as *to have sympathy for someone*, which corresponds to the German *Mitleid mit jemandem haben*. Furthermore, apparently-equivalent words can appear in different syntactic configurations: In English, the verb *to fill* can be used as in *They filled the bottle with water*, but not as in **They filled water into the bottle*. In German, with the corresponding verb *füllen*, both forms are perfectly all right.

In these examples the words are very similar and can in many situations replace each other, being almost equivalent in meaning. Other problems of semantics deal with rather unrelated words, whose replacibility is specific to the situation of use. For instance, the sentences *Remove the engine oil filler cap* and *Open the engine oil tank* are instructions to perform precisely the same action, yet they are clearly not synonymous from a purely lexical viewpoint. Similarly, the English section of an automobile manual asks the reader to *disconnect the spark plug wire*, while the corresponding German text suggests *das Zündkabel abziehen* ('pull off the spark plug wire'), and in either case the reader is enabled to act as required. These are not purely lexical phenomena anymore; instead, we are at the borderline between words and the abstract content "behind" them—we are moving into *knowledge representation*. The same thing can be said in different ways, in one or in several languages, without necessarily using synonymous or near-synonymous words.

Of the two areas of study just introduced, lexical semantics investigates the meaning of words, whereas knowledge representation is concerned with modelling aspects of the world for purposes of reasoning. One field that needs to deal with both these areas is *natural language generation* (NLG), the field whose task it is to map information represented in some particular non-linguistic form to a natural language that humans can understand. The source information that is to be verbalized could be raw data, as in systems developed for weather or stock market reports, or some structured knowledge representation written in a formal language, as in explanation facilities of expert systems. The knowledge-based one is the more standard way of approaching generation and is the one that will be pursued in this thesis.

For this kind of NLG, a crucial prerequisite is to find a level of representation that is on the one hand abstract enough to be neutral between different paraphrases in one or more languages, and that on the other hand can still be mapped systematically and in a number of steps to linguistic output. Finding this "deep" level of representation and devising the mechanisms that map it to language are difficult tasks; they involve drawing a line between the "pure" content and the linguistic packaging, i.e., the different ways of saying roughly the same thing. In many practical applications of NLG, this problem can be circumvented when no significant variety of output text is necessary. But in applications where this is not so, many truly interesting research questions arise: How do we define and delimit the range of utterances that can be produced from the same deep representation, and on what grounds do we make a sensible choice among the possible options? The study of paraphrases deals with exactly this problem.

Paraphrases If two utterances are *paraphrases* of one another, they have the same content and differ only in aspects that are somehow secondary. A more precise account of this notion will be developed in the chapters to follow. To illustrate, typical devices for deriving *syntactic* paraphrases are *topicalization* and *clefting*, which make a certain constituent of the sentence especially prominent.

> *Sandy gave the key to Dolores.*
> *To Dolores, Sandy gave the key.* (topicalization)
> *It was the key that Sandy gave to Dolores.* (it-cleft)

In a specific discourse situation, one or another of these versions may be the most appropriate to say.

Another important source of paraphrasing is *lexical* variation, which is the central theme of this thesis. Chapter 4 suggests a classification of the phenomenon; for now, consider an example:

> *This year we'll go to Texas by plane.*
> *This year we'll take the plane to Texas.*
> *This year we'll fly to Texas.*

The first variant is somewhat more formal than the others, and the third is different from the first two in that the information is "packed" in a different way: the verb incorporates the 'instrument' of the activity, which in the first two sentences is expressed separately (and differently). Again, any of these could be the most felicitous to utter under particular circumstances. Note that in this example different lexical choices have resulted in slightly different sentence structures. This is not always the case; often, a word can be replaced by a (near-) synonym of the same syntactic category, with the rest of the sentence remaining unchanged.

Multilinguality Given that language generation proceeds from an abstract representation of content, it seems natural to pursue the idea of mapping that representation not just to one language but to many. For multilingual generation, the key problem is to separate the language-specific knowledge resources (grammar, lexicon) from all the others so that as many resources as possible can be shared between the representations involved. This presupposes detailed investigations of the similarities and the differences between the target languages, and a careful design of the levels of representation within the system: It is necessary to capture the "common content" of utterances in different languages in an adequate representation, and only then to apply the knowledge of how this content is typically verbalized in each particular language.

For example, in a multilingual automobile manual, we find the English instruction *Twist the cap until it stops*, consisting of two clauses, and the corresponding German one using a single clause with modifying prepositional phrase: *Drehen Sie den Deckel bis zum Anschlag* ('Twist the cap up to the stop'). In this example, there is no felicitous way of expressing the instruction in both languages with the same syntactic construction; we are faced with a genuine cross-linguistic *divergence*. More examples will be given in chapter 4. Often, however, there is a choice as to which construction to employ, and that is why multilingual generation is very closely related to the problem of producing paraphrases within a single language. "Saying the same thing in different ways" can be done in English, or in English and German, or in English, German, and French at the same time. This idea is a central element of the research presented here: treat multilingual generation as a straightforward extension of the problem of monolingual paraphrase production, and devise representation levels and a system architecture to accomplish this unified task.

The possibility of generating text in multiple languages in fact holds some promise of making NLG economically interesting as an alternative to machine translation: When an abstract representation can be converted to multilingual output, quite a few interesting practical applications can be thought of. But curiously, this field is very young, and only few results have been achieved with multilingual generation so far. In Canada, one such system is used to produce English and French weather reports [Kittredge et al. 1988]. There are also efforts to build multilingual generators in the framework of systemic-functional grammar [Bateman et al. 1991], and a few other research projects have just recently started in Europe, where the issue of multilinguality is—understandably—seen as more pressing than in other, unilingual communities. One of these projects is TECHDOC, which will be described in the next section.

1.2 Background: the TECHDOC generator

The research presented in this thesis grew out of experiences with building the TECHDOC generator [Rösner and Stede 1992; 1994] at the Research Centre for Applied Knowledge Processing (FAW) in Ulm, Germany. That project aimed at supporting the creation of technical documentation in English, French, and German by knowledge-based text generation. The first texts dealt with were maintenance instructions from automobile manuals, and a few similar types have been added since. Instead of manufacturers manually writing instructions, translating them, and re-iterating this loop with every round of updates, the idea is to maintain a knowledge base that includes an abstract model of the product in question, and to produce documentation in multiple languages automatically from that.

The system is based on a knowledge base (KB) that encodes knowledge about the technical domain and the specific product, and also knowledge about schematic text structure. An instantiation of this general knowledge is a specific plan (in the traditional AI sense). In generation, the plan is in first mapped to a tree that captures the discourse structure of documents by means of discourse relations holding among elementary propositions or sub-trees, as has become fairly standard in text generation. This document representation is successively transformed into a sequence of *sentence plans*, which are handed over to surface sentence generation modules. For English, the PENMAN generator [Penman 1989] is used with its 'sentence planning language' for specifying input terms. To produce German text, a version of parts of the PENMAN grammar, as well as several enhancements, has been implemented, which is completed by a morphology module; a fragment of a French grammar was developed in the same style. Output is produced either for printing, with LaTeX formatting instructions included, or for the screen, where the text is "clickable", i.e., it can be interactively queried for various purposes.

A critical bottleneck of the current system is its requirement that the same semantic sentence specification be given to the language-specific sentence generators, which transform it into German, English, and French. From the perspective of system architecture, this is elegant and straightforward; but, as we have already seen, even within the fairly simple linguistic domain of technical manuals one finds cases where the languages are not parallel enough to warrant identical sentence specifications. The deeper reason for this deficiency of TECHDOC is its reliance on two basic assumptions inflicted by the PENMAN system: that a lexical item correspond to exactly one KB concept, and that the domain model (see chapter 5) be subsumed under the so-called Upper Model, a hierarchy of concepts designed to capture linguistic distinctions of English. The UM will be explained in section 2.5.2. In effect, the knowledge representation scheme underlying the system is being forced into the categories of a specific language—which puts tight restrictions on possible variety in monolingual and multilingual text output. Thus, TECHDOC, like most current NLG systems, lacks genuine *lexical choice.*

1.3 Goals of this research

In light of the problem just described, the target of this research project is a generator that can systematically "say the same thing" in different ways and in different languages, that is, produce a wide range of multilingual paraphrases for the same underlying content. The input to the system will be a language-neutral representation, which can be produced by an application program (such as an authoring tool), and the output is a range of alternative sentences in English or German. Furthermore, we will account for two dimensions of preference, so that an actual choice can be made.

Importantly, the system architecture is to be devised in such a way that multilingual output can be produced by the very same paraphrasing mechanism, so that generating English paraphrases is in principle the same as generating both English and German paraphrases. Thus, the system has to account for certain divergences between languages, i.e., phenomena where languages use different means to verbalize the same idea. But such divergences should not be seen as a nuisance or even a problem: they should "fall out" of the paraphrasing capabilities.

The project will concentrate on *lexical* variation within and between languages, and to that end thorough specifications of word meaning have to be devised. In general, a large part of the overall job is to find suitable *representations*, to distribute the kinds of information to various sources, and to make the 'right' abstractions so that language-neutral and language-specific representations can be systematically mapped to each other. And the central instrument in this mapping is to be the lexicon of the target language, which thus serves as a "bridge" between the language-neutral and the language-specific level. In contrast to the fixed concept–lexeme associations of previous generators and a correspondingly marginal semantic role of the lexicon in the generation process, the system developed here has to put the lexicon right in the middle so that with the help of flexible mappings, a wide range of verbalizations is made possible.

The focus of the research will be on verbalizing *events* and therefore on verb semantics. By using rather fine-grained representations, which break up the internal structure of events, as input to the generator, it is possible to systematically map the input to different verbalizations, one (or more) of which will be the most appropriate in a particular context. To this end, linguistic research on aspectual structure and lexical semantics needs to be extended and transferred to the realm of practical language generation.

Importantly, we will in this work concentrate on the relation between pre-linguistic knowledge and language-specific lexical semantics. Combining these two types of knowledge is an essential task for language generation, but previous NLG research has made only few contributions on interfacing with knowledge bases. Therefore, the perspective from which we will approach NLG will be a predominantly semantic one; consequently, we will have less to say about detailed syntactic phenomena. In fact, most of the syntactic realization decisions will be left to front-end generators that will be treated largely as "black boxes"—the interesting problem then will be to define an adequate interface.

1.4 Overview of the research and its results

The groundwork for developing a system along the lines sketched above is an analysis of a number of linguistic questions and a thorough examination of the relevant literature. One contribution of the thesis is thus in identifying appropriate linguistic research, and modifying and extending it for the purposes of language generation. To this end, we will in chapter 3 review a variety of work in lexical semantics that influenced the design decisions for the generator. In later chapters, we will then point out how that research relates to the various aspects of generation.

The subsequent task is to define the levels of representation of information in such a way that both fine-grained lexical variation and multilingual output are possible. To accomplish this, we will design a two-step generation process that first maps a language-neutral and paraphrase-neutral situation specification, which we call a *SitSpec*, to a language-specific semantic sentence specification, a *SemSpec*. The SemSpec can then be processed by a conventional surface generators to produce an English or German sentence.

SitSpecs The design of the SitSpec representation level is motivated by two different considerations: that it be sufficiently language-neutral to be mapped to several natural languages, and that it can act as an interface to some underlying application program (such as TECHDOC). Therefore, SitSpecs are not *linguistic* representations; rather, we will see them as *instantiated domain knowledge*. To achieve a wide range of lexical variation, it is crucial to make appropriate ontological distinctions in *modelling the domain* that the system operates in, and we will do that carefully (chapter 4). Because the SitSpecs are to be produced by an application program that will be doing reasoning, planning, or simulation, we use a standard knowledge representation language of the 'KL-ONE' family (nowadays also known as 'description logic') for defining the domain model that SitSpecs are an instantiation of (section 5.2). As a result of choosing KL-ONE, we have the instrument of *subsumption checking* available, which will prove very useful in determining the range of possible verbalizations.

We will illustrate our generation approach with the task of verbalizing *events* in various ways—an area that generation research has largely neglected so far. Therefore, we need to study the internal structure of events and how it is reflected in language, in order to build an ontology for the SitSpec level. For example, verbalizations may differ in their emphasizing

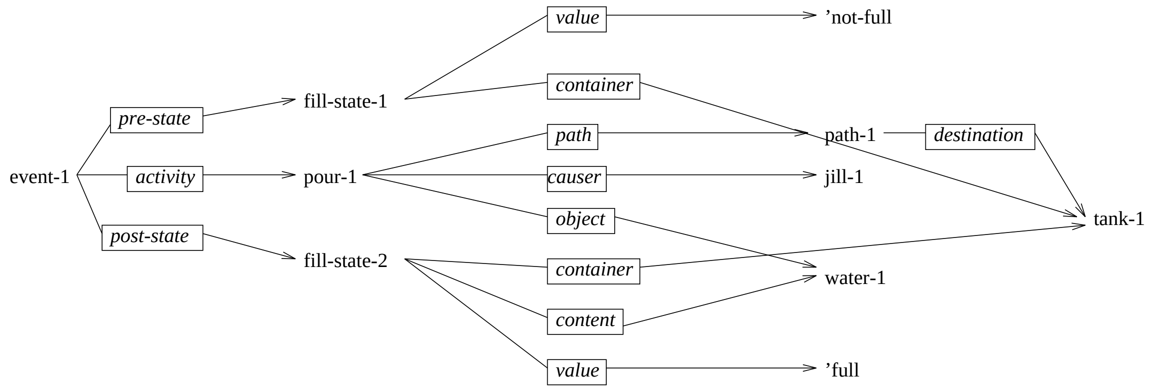

Figure 1.1: Example of SitSpec: Jill filling a tank with water

```
(x1 / anterior-extremal :domain (x2 / directed-action :lex pour_el
                                   :actor (x3 / person :name jill)
                                   :actee (x4 / substance :lex water_el)
                                   :destination (x5 / three-d-location :lex tank_el))
                         :range (x6 / nondirected-action :lex fill_el
                                   :actee x5))
```

(1) *Jill poured water into the tank until it was filled.*

```
(x1 / directed-action :lex fill_el
                      :actor (x2 / person :name jill)
                      :actee (x3 / three-d-location :lex tank_el)
                      :inclusive (x4 / substance :lex water_el))
```

(2) *Jill filled the tank with water.*

Figure 1.2: Examples of SemSpecs and corresponding English sentences

either the *result* of an event or the *activity* bringing about the result. We therefore propose to represent events, in the general case, as a tripartite structure:

- A PRE-STATE that holds before the event commences;
- A POST-STATE that is in opposition to the PRE-STATE and holds when the event has completed;
- An ACTIVITY that brings the state change about.

As an example, figure 1.1 shows a SitSpec representing a situation in which a person named Jill puts some water into a tank. We show the KL-ONE representation as a directed acyclic graph, in which the nodes are instances of concepts from the domain model (or atomic values in the case of 'full and 'not-full), and the arcs are labelled with relations holding between the instances (appearing in boxes). Our treatment of events is discussed below in section 5.3.

SemSpecs The level of SemSpecs is a linguistic level of representation, which reflects the lexical choices that have been made, but abstracts from syntactic details. To define SemSpecs, we make use of the notion of 'upper modelling' [Bateman et al. 1990], as it was introduced with the PENMAN sentence generator (described below in section 2.5.2). An Upper Model is a *language-specific* ontology that reflects the conceptual and lexical distinctions a particular language makes and guides a surface generator in making its syntactic decisions. Starting with the PENMAN Upper Model, we will argue for a small but important re-interpretation of its role, so that *lexicalization* can be seen as the central step in deriving SemSpecs from SitSpecs. Therefore, SemSpecs will be defined as a well-constrained subset of the SPL language [Kasper 1989] that is used as input to the PENMAN generator (section 6.4). In brief, a SemSpec will be composed of a variable representing the entity expressed, a type from the Upper Model, and a number of keyword/filler pairs, where the keywords can be roles akin to semantic 'deep cases' (actor, actee, etc.). For illustration, figure 1.2 shows two SemSpecs and the corresponding sentences produced by PENMAN. Our system can derive both of them (and more) from the SitSpec in figure 1.1.

Lexicalization Since the goal of this thesis requires handling monolingual and multilingual lexical phenomena in generation, we assign a prominent role to the task of *choosing words*. Selecting open-class lexical items in our framework includes both the decisions on distributing elements of the SitSpec across the words to be used, and choosing a particular verb *alternation* in order to suitably place emphasis upon an appropriate element of the sentence.

As a prerequisite, such a system requires lexical entries that are more refined than those used in previous generators. We will in chapter 7 posit lexical entries as consisting of the following:

- The *denotation* of the word: its applicability condition with respect to SitSpecs;
- The subset of SitSpec nodes actually *covered* by the word;
- A *partial SemSpec* (PSemSpec): the contribution the word can make to sentence meaning, i.e., to a SemSpec;
- The *connotations*: a list of stylistic features and values;
- For verbs only: the assignment of *salience* to the participants and circumstances;
- For verbs only: pointers to *alternation and extension rules* that apply to the verb.

To set the stage for the lexical choices, we first determine the pool of *verbalization options*: the set of words that could possibly be used to express some part of the input SitSpec. This set is found by a *matching process* that compares lexical entries to the SitSpec. Importantly, the matching does *not* check for identity of nodes, but rather for subsumption; we thus find lexical options that are more or less specific and can possibly incorporate certain units of meaning.

Since our focus is on verbalizing events, the central linguistic topic will be the semantics of verbs. For example, here are the denotation and PSemSpec of the lexical entry of the stative *to fill* (somewhat simplified for now); the denotation is a SitSpec template, where relation names and variables appear in upper-case letters and concept names in lower case. Notice the co-indexing of variables in the denotation and PSemSpec.

```
Denotation: (fill-state (CONTAINER A)
                        (CONTENT B)
                        (VALUE 'full))

PSemSpec: (x / directed-action :lex fill  :actor B  :actee A)
```

We will demonstrate how the denotation and the partial SemSpec representations can be employed to systematically derive more-complex verb configurations from simpler ones; this amounts to a new formalization of linguistic research on *verb alternations*. We will propose a set of rules that implement a number of such alternations (section 7.3). For the case of *to fill*, for instance, this means that the lexical entry of the verb need represent only its minimal configuration, which is the *stative* reading (*Water filled the tank*), and more-complex readings will be derived by productive rules. For *to fill*, the rules will derive first the *resultative* reading (*The tank filled with water*) and then the *causative* one (*Tom filled the tank with water*):

```
Denotation: (event (ACTIVITY X)
                   (POST-STATE (fill-state (CONTAINER A)
                                           (CONTENT B)
                                           (VALUE 'full))))

PSemSpec: (x / nondirected-action :lex fill  :actor A  :inclusive B)

Denotation: (event (ACTIVITY (CAUSER C))
                   (POST-STATE (fill-state (CONTAINER A)
                                           (CONTENT B)
                                           (VALUE 'full))))

PSemSpec: (x / directed-action :lex fill  :actor C  :actee A  :inclusive B)
```

By means of such derivation rules, we significantly reduce the number of lexical entries, and thereby reduce also the cost of the initial matching phase when verbalization options are determined. The second phase of the generation procedure thus consists of applying the derivation rules to those verbs that have been determined as lexical candidates in the first phase.

In the third step, the verbalization options are brought into an *order of preference* for every SitSpec node that needs to be verbalized. We will deal with two parameters here: the assignment of *salience* to the different elements of the sentence (section 7.4), and the *connotations* to be associated with the sentence (section 7.5). Then, the central task is to search the pool of verbalization options for a subset of options so that

- the denotations of the options collectively cover the entire input SitSpec,
- the PSemSpecs of the options can be combined into a well-formed SemSpec, and
- the options participating in the SemSpec are preferred (in a weak sense).

To implement the search procedure, we relax the requirement of finding the *overall* preferred verbalization. Instead, we consider the PSemSpecs in their *local* order of preference at every SitSpec node to be verbalized. Only when backtracking becomes necessary is a less-preferred option chosen at that node.

The mechanism for building a SemSpec from the SitSpec takes the preferred verbalization option for the root node of the SitSpec and tries to replace the variables therein with other SemSpecs, by calling itself recursively. Backtracking becomes necessary only if at some point the SemSpec that is supposed to replace a variable is of an incompatible type with respect to the Upper Model, or if the verbalization options do not cover the entire SitSpec. SemSpec construction therefore relies on the separation of denotation and PSemSpecs, and their being linked by shared variables. The procedure will be explained in chapter 8.

Finally, a surface generation module maps the SemSpec into a natural language sentence. For English, we use the PENMAN system, with several modifications made in the TECHDOC project at FAW Ulm, and for German a variant of PENMAN that was also developed in the TECHDOC project.

Results With the instruments of rich lexical specifications, the subsumption check in the matching phase, the coupling between lexicon and background knowledge base, and the central role of the words in finding a SemSpec, the system can produce from an input SitSpec linguistic variants of the following kinds:

- Differences in connotation: *The dog annoyed me. / The dog drove me up the wall.*
- Different incorporations: *Tom went to London by plane. / Tom flew to London.*
- Different specificity: *The dog / animal barked all day.*
- Emphasis on different aspect: *Pour water into the tank until it is full. / Fill the tank with water.*
- Situation-specific paraphrases that do not result from lexical near-synonymy: *Open the tank. / Remove the cap from the tank.*

Since lexical entries clearly distinguish their various kinds of information, parts of entries can be shared by similar words; for instance, near-synonyms like *to die* and *to perish* would share most of their lexical entries and differ only in their connotations, specifically their formality. And importantly, lexical entries can be shared across languages just as easily: the relation of 'near-synonymy' extends to a multilingual environment. For example, English *to fill* and German *füllen* have the same denotation in their stative readings (which are the ones stored in the lexicon) and the same PSemSpec, but differ in the alternation rules that apply to each.

Moreover, in our approach, certain lexical divergences between languages "fall out" as a side effect of the monolingual paraphrasing capabilities. Inter-lingual differences in incorporation, specificity, and emphasis are handled by the very same mechanism that produces the monolingual variation. In fact, language-specificity enters the generation process only at two points: at the very beginning, the lexicon for the language in which output is to be produced is used for the matching phase, and at the very end, the corresponding surface generation module is activated.

Finally, we will demonstrate the effects of the two choice dimensions accounted for in the system (salience and connotations) on the generation process. In the *fill* example in figure 1.2, the choice between sentences (1) and (2) can, for instance, result from associating a 'foreground' label in the SitSpec with either the 'activity' node (for sentence 1) or the 'post-state' node (for sentence 2). Or, more indirectly, it can result from labelling the node 'water-1' as 'foreground'. In this case, our system tries to assign a prominent role to the constituent *water* in the sentence, which cannot be accomplished when using the verb *to fill*; hence sentence (1) is preferred. In these and other ways, a desired salience assignment can direct the choice of the verb.

In chapter 10, we will compare our approach to related research by others and analyze in detail the differences between our system and a few that pursue similar goals. In general, evaluating natural language processing systems is a difficult matter, and the debate on this topic, which started in the research community several years ago, has not really resolved the issue. For language *generation*, the evaluation question is probably even more difficult than for language understanding, because there is so little agreement on what the "best" input to a generator is—it all depends on the particular purpose of the system. Therefore, there is little point in comparing I/O behavior or execution times; our arguments will instead center on the *architecture* of our system, which is designed to handle a wide range of paraphrasing phenomena and to be adaptable to different domains and generation tasks.

1.5 Organization of the thesis

Chapter 2 reviews the literature on lexicalization in natural language generation, focusing on the aspects addressed in this thesis. It determines the state of the art and identifies the central weaknesses of current systems with respect to lexicalization.

Chapter 3 is the second 'background' chapter; it introduces the topic of lexical semantics and reviews those works of linguistic research that will be used in designing our system.

Chapter 4 provides a classification of the various kinds of lexical variation we find within a language and between languages. It thus produces a map of the target phenomena to be dealt with in the thesis.

Chapter 5 takes the first step to building out generation system: modelling the domain. It gives a short introduction to the knowledge representation language chosen, and then discusses the general ontological decisions that we made for representing domain knowledge. Following this layout, the concrete model for the sample domain of our system is developed, on the basis of which the generation system will operate.

Chapter 6 discusses the levels of representation used in the generator: the language-neutral level of situation specifications, closely related to the domain model, and the language-specific level of semantic sentence specifications.

Chapter 7 develops the complex lexical entries used in our system, consisting most prominently of the interfaces to both the situation specifications and the semantic sentence specifications.

Chapter 8 combines the building blocks provided in the previous chapters and presents a novel system architecture for multilingual sentence generation. The overall generation procedure is specified in detail.

Chapter 9 shows some of the output that our system produces. Returning to the general scheme of 'situation' as developed in chapter 5, this chapter shows the possibilities of verbalizing the different kinds of situation in English and German. The last section of the chapter demonstrates how the various pieces of lexical information introduced in chapter 7 work together in deriving verbalizations.

Chapter 10 summarizes the work, compares it to related work by other researchers, states the contributions made by the thesis, and points to some promising areas of future research.

Typeface conventions Although the distinctions are not always straightforward to make, the thesis uses different typefaces to separate entities belonging to different realms of representation. *Slant* marks linguistic examples, whereas concepts and relations on the pre-linguistic level are given in SMALLCAPS. *Italics* are reserved for emphasis, some proper names of systems appear in UPPER CASE, and excerpts of actual program code or representations in `typewriter`.

Asterisk and question mark conventions Linguists have developed a tradition of marking utterances they consider ungrammatical with a preceding $*$, and those whose well-formedness they find "questionable" or "very questionable" with ? and ??, respectively. It is well-known that determining these assignments is a problematic endeavour, because the linguists' introspection is typically not the ideal tool for determining whether some utterance is acceptable or not; besides, what does it mean to be "grammatical" or "acceptable" anyway?

This thesis has no answer to the questions, but it occasionally makes use of such judgements, too. Here, the $*$, ?, and ?? simply result from the author's intuitions, and his inquiries to native speakers in the case of English data.

Chapter 2

Lexicalization in NLG

After introducing the notion of lexicalization, this chapter reviews the state of the art in natural language generation with respect to lexicalization, focusing on the issues that are immediately relevant for developing our own system in the later chapters. At the end of the chapter, the central weaknesses of current generation systems on the side of lexicalization are summarized.[1]

2.1 Introduction

In the common approach to NLG, the task is split into *strategic* and *tactical* components, the former deciding on *what* to say, and the latter determining *how* to say it.[2] The strategic component selects the content of the text and arranges it in a suitable order, represented as a *text plan*. The tactical component is then in charge of organizing the text representation into a sequence of sentences and realizing them. This thesis focuses on the second task, translating a content representation into language, and starts with the assumption that a sentence-size input has already been constructed. Now, decisions to be made involve the ordering of the information in the sentence, and the syntactic structure; for example, deciding between the use of a relative clause or an adjective. But, probably the central task in sentence generation is *lexicalization*: How do individual words find their way into sentences? This, in fact, is also a two-step process: one first chooses the *lexeme*, an abstract base form that can be realized in various grammatical forms [Bußmann 1990] (it loosely corresponds to an entry in a dictionary), and at the end produces from the lexeme a fully-inflected word. As we will not be concerned with morphological realization, our point of interest is lexeme choice. However, we will not always adhere to the technical terminology and often use 'word choice' in this sense.

With respect to this question, a common linguistic distinction is made between the selection of *open-class* and *closed-class* lexical items. The former include verbs, nouns, adjectives, and some adverbs (also called *content words*), and they are being treated as the 'interesting' part of lexical choice, usually selected by some special mechanism. On the other hand, the usage of conjunctions, prepositions, etc. is usually governed by grammatical decisions, hence not subject to a proper 'choice' process. While this distinction is not entirely without problems,[3] we adopt

[1]This chapter is a revised and shortened version of a more comprehensive overview and analysis of all the issues relating to lexicalization, which has appeared in *Artificial Intelligence Review* (Stede [1995]). Reprinting the material was kindly permitted by Kluwer Academic Publishers.

[2]The theoretical feasibility of separating these tasks has often been questioned (e.g., by Danlos [1987]), but practical generators that employ a truly *integrated* architecture have only been proposed recently (e.g., [Ward 1991; Reithinger 1992; Kantrowitz and Bates 1992]). Still, the major argument in favor of a two-step, modular design is that it keeps control flow simple and separates the different knowledge sources involved.

[3]For example, even in the seemingly innocent choice of prepositions, we notice stylistic differences like the

it here and look only at open-class items. We see the task of lexicalization as revolving around five issues, which will be discussed in turn:

- What is a lexical item? The basic unit in the dictionary of an NLP system is typically the single word, but in generation there was often an emphasis on accounting for phrasal expressions (e.g., idioms).

- What are the criteria for choosing particular lexical items? Quite often, researchers have lamented that the problem of word choice has not received sufficient attention, e.g., [Marcus 1987; Nirenburg and Nirenburg 1988; McDonald 1991]—most language generators assume that for every concept in the input expression there is exactly one associated word. Yet, when lexicalization is indeed seen as a matter of *choice*, factors determining the differences between lexical items need to be found, and taking at least some of them into consideration can enhance the expressiveness of a generator considerably.

- How are lexical items linked to concepts in the knowledge base? The input to a generator is a meaning representation that typically derives from an underlying knowledge base. To produce language, KB concepts have to be associated with lexical items, which can be done in various ways.

- When is the dictionary accessed? At what point in the overall generation process are words actually selected from the dictionary?

- How is lexicalization done in a multilingual environment? When multiple languages are to be produced, the role of lexicalization needs to be adapted to account for all of them.

2.2 The nature of lexical items in NLP

What is in a dictionary? The standard answer is "words", but language generation has often made a point of using complete phrases as lexical entries, which can account for the multi-word idiomatic expressions in language. At the same time, a 'phrasal lexicon' can be employed to reduce or even replace the need for building sentences compositionally: in certain domains it makes sense to associate fixed phrases with semantic input expressions and use only an impoverished grammar to join the phrases together (as done in ANA [Kukich 1983]).

To mention just one system, Hovy's work on the system PAULINE was strongly motivated by a quest for phrasal patterns. Hovy [1988b] states that "the lexicon should be the sole repository of the patterns that make up language—some very specific, some very general." The lexicon thus includes not only idiosyncratic forms of expression that are directly associated with concepts, but also the general formative rules of grammar, encoded as patterns. The implementational device for coordinating the information that is distributed to lexical items is a set of *syntax specialists*: procedures that are in charge of producing a certain linguistic constituent from a meaning representation. There are specialists for building noun phrases, sentences and other phrase structure entities, but also for more idiosyncratic tasks like expressing a time or a color. Likewise, phrasal templates encode specific linguistic behavior, but they have the same status as the specialists: they are merely a special case, a trivial procedure. Therefore, the

one between *on* and *upon*. More importantly, connectives can play a significant role in conveying aspects of meaning, as investigated for instance by Elhadad and McKeown [1990], and by [Grote, Lenke, Stede 1995]. Also, see the distinction between discourse-oriented and proposition-oriented closed-class items, made by Pustejovsky and Nirenburg [1987].

collection of syntax specialists—procedures and templates—constitutes the system's lexical as well as grammatical knowledge, and the generation process amounts to recursively calling more specialized procedures (or applying patterns), starting with a high-level specialist for expressing a sentence.

Approaches like these are a start for dealing with phrases and idioms, but a comprehensive and systematic treatment of the characteristics of phrasal items (nominalization, passivization, inserting extra constituents, altering word order, etc.) has not yet been accomplished in NLG. This is for the most part due to the fact that theoretical linguistics has largely ignored this matter, so that there are hardly any results to start from. There is no "off-the-shelf" classification of idiomatic phrases in terms of their syntactic behavior and their relation to grammar—presumably because idioms question the role of traditional grammar as such; they are part of the "messy" side of language that (so far, at least) resists formal description.

In this thesis, the issue of phrasal items and idioms will not be a topic of dicussion. Our system will permit single words and phrasal verbs as lexical entries, but no other phrases.

2.3 Criteria for lexical choice

When a language generator has a variety of lexical items for expressing a concept at its disposal, the task of actual lexical *choice* arises. Human beings use different words in different situations to say roughly the same thing, and the choice criteria are multifarious: Particular *genres* (e.g., sports talk) have their own special vocabulary; there are words of different *style* (e.g., formal and colloquial); words might or might not express our *attitude* towards a state or affairs, etc. The number of factors that influence lexical choices in language production and make people prefer one word over another is very large, and the interaction of these factors is complex. NLG research, in contrast, has looked at several individual choice factors in isolation, and sometimes in depth. But no attempt has been made at what Busemann [1993] called "holistic" lexical choice: an algorithmic scheme that would try to integrate all the relevant factors. That, however, is certainly not a short-term research goal. For one thing, we still do not know enough about the individual criteria. And furthermore, it is unclear how to effectively handle the interactions between the criteria, which can at times be in conflict with one another, as we have seen.

A special case of word choice is the construction of *referring expressions*, i.e. the decisions on definiteness and pronominalization, and on the specificity of the terms to use. This problem has been explored by generation research extensively[4] but will not be discussed here, because it concentrates on the particular task of identifying objects in a given context; we are instead looking at more general criteria for selecting words from sets of options.

In this thesis, the issue of choosing the most appropriate lexical item will not be solved conclusively; instead, the emphasis is on making a range of paraphrases *available* to a generator—which is a prerequisite for choice. Importantly, though, we will design the architecture in such a way that a treatment of choice criteria can be integrated into the system. And to demonstrate the range of verbalizations available, we will implement two choice factors: that of attributing *salience* to the various elements of the sentence, and a set of stylistic criteria for handling fine-grained differences between similar words. Thus, we now briefly review the research done in NLG on these two topics.

[4]See, for instance, [Appelt 1985; Novak 1988; Dale 1989] and, focusing on the notion of text cohesion and avoiding the repetition of identical noun groups, [Granville 1984; Buchberger and Horacek 1988]. A broader survey of 'discursive constraints' on lexicalization, including pronominalization decisions, can be found in [Robin 1990].

2.3.1 Salience

A number of generation systems account for the fact that different parts of the input material may have different degrees of prominence associated with them; specifically, one aspect is often said to be *in focus* as compared to the others. The decision as to which element deserves the focus role in the sentences is commonly made by the strategic component (for example, in accordance with patterns of theme development in texts), so that the sentence generator can assume that an item of the input material is already marked for being focused on. One common way to express focus is *thematization* of a constituent that would normally occur elsewhere in the sentence (*Shakespeare is the author of the book that Jim read yesterday*), but often it also influences lexical choice. For instance, Jacobs [1987] discusses the example of transfer-events that can be reported from different viewpoints, which results in sentences with the main verb being either *buy* or *sell*, depending on which participant is in focus. Pustejovsky and Nirenburg [1987] use the same example and make the point that the notion of focus ought to be differentiated further into (1) the intended perspective of the situation, (2) the emphasis of one activity rather than another, and (3) the focus being on a particular individual; however, they do not elaborate how these factors would exactly interact in sentence production and word choice.

In the GOSSiP system [Iordanskaja, Kittredge, and Polguère 1991], which is rooted in the linguistic theory of the Meaning-Text Model (MTM) [Mel'čuk 1988], the input semantic network consists of two regions marked as 'theme' and 'rheme', respectively. Theme/rheme structure is related to the focus notion; the idea is that every declarative sentence falls into these two parts—a thing that the sentence "is about" (theme, at the beginning of the sentence) and the information that is reported about it (rheme). In GOSSiP, lexicalization is influenced in two ways: When two lexemes both match the same sub-net (e.g., *send* and *receive* both match the underlying semantic structure), then the one is chosen whose first participant is in the net region marked as the theme and becomes the sentence subject. The other source of variant lexicalization results from the fact that both in the theme and rheme region one node is always marked as 'dominant', and the verbalization of the dominant node in the theme region is always to be the realized theme of the sentence. Thus, when a node labelled 'duration' is not dominant, it gives rise to an expression like *for two hours*; if it is the dominant theme node, the sentence will be akin to *the duration was two hours*.

A related approach, also rooted in the lexical functions of the MTM, is presented by Wanner and Bateman [1990]. They use a representation of abstract *situations* from which input expressions for the sentence generator are produced in accordance with a chosen *perspective* on the situation. Perspectives differ in terms of the *salience* they attribute to the different aspects of a situation, which loosely corresponds to the notion of focusing, but is more elaborate because complete configurations of salience attributions can be specified for a sentence, instead of just a single element being focused on. A *system network* (similar to a set of decision trees) implements the distinctions to be made in characterizing a perspective; traversal of the network results in the choice of appropriate lexical functions that will drive the linguistic realization of that perspective. The system network is split into four groups of decisions: (1) causality orientation—does the situation involve an active or passive causation? (2) situational orientation—is the orientation towards a described situation, a process, or the participants, and which of them? (3) temporal orientation—how is the process arranged on the temporal axis, and is it oriented towards the result of a process? (4) process stages orientation—is the emphasis on the beginning, continuation, or termination of a process? By making the necessary decisions in these four groups, associated lexical functions are selected that serve to translate the specification of an abstract situation into a concrete input expression for the sentence generator, which will produce a verbalization that reflects the chosen orientation.

2.3.2 Pragmatics and style

Hovy's [1988a] generator PAULINE was the first system to produce text in accordance with variable communicative intentions: a number of *rhetorical goals* are translated into *stylistic goals* whose realization influences lexical choice, amongst other decisions. For instance, when the purpose of the communication is to teach the hearers or activate certain goals in their mind, PAULINE can add *color* to the text by preferring idioms and frozen phrases. When *affect* is to be expressed, so-called enhancers and mitigators give rise to constructions like *X; in addition, Y* or *X; however, Y*. Adverbs like *really, extremely* and *just, only* fulfill the same function. Verb choice is a very important resource for communicating affect, too; Hovy gives the example of *tell* as a neutral word, and its synonyms *order, command* (enhancers) and *request, ask* (mitigators). Adjectives can be selected to express an opinion about a state of affairs: *wonderful, nice, nasty*, etc., and suitable noun groups can convey different attitudes: *the gentleman / that jerk*. Two more dimensions that PAULINE commands are *formality*, where the system uses or avoids popular idioms, slang terms and contractions, and *force*: to produce forceful text, simple, plain words and phrases are chosen, whereas flowery and unusual options are avoided. In earlier work [Stede 1993], we have applied a scheme along these lines to the PENMAN sentence generator [Penman 1989] and enabled it to perform a preferential word choice based on six stylistic dimensions. For example, depending on the desired stylistic color, the generator produces *Tom evicted the tenants, then he tore the building down* or *Tom threw the tenants out, then he pulverized the shed* from the same meaning specification. An open question, however, is how the settings for stylistic features are acquired for the lexicon; DiMarco *et al.* [1993] suggest formalizing existent *usage notes* in dictionaries and making them accessible for NLP purposes.

Related to the *affect* dimension, Elhadad [1991] investigated the use of adjectives and pointed out that besides their referential or attributive function, adjectives also convey *argumentative intent*. He analyzed a corpus of conversations between students and their advisors on the topic of course selection and classified adjectives with a similar meaning in terms of their argumentative features. For instance, advisors neutrally described a course as *difficult*; but when they wanted to discourage the student from taking it, they used *hard*. Therefore, lexical entries for adjectives were supplemented with features denoting the semantic scale affected by the adjective and the value that the word expresses on that scale.

The COMET system [McKeown et al. 1993] tailors word choice to the vocabulary that the user is presumed to command and employs four strategies to rephrase a message in cases where the user model indicates that some word will not be understood: choose a synonym provided by the lexicon; rephrase with a conceptual definition, e.g., give a lower-level description of a term; rephrase a referring expression (*the COMSEC cable*) with a descriptive phrase (*the cable that runs to the KY57*); use past discourse to construct a new referring expression (*the cable you just removed*). The user model relates the lexicon entries to annotations that indicate whether a stereotypical 'good' or 'poor' reader will be familiar with the term and thus establishes additional constraints for the lexical chooser module that is in charge of selecting the words.

2.4 Linking concepts to lexical items

When text generation proceeds from an internal meaning representation to natural language output, the elements of the representation need to be somehow linked to lexical items of the language. The more simple and rigid this association is, the simpler is the task of generating language—but very little output variety can be achieved. This section reviews approaches to more flexible association schemes.

2.4.1 Discrimination nets

The first invention for word-concept linking was the *discrimination net*, proposed by Goldman in the 1970s, and it proved to be highly influential for subsequent work in generation. The BABEL generator [Goldman 1975] was part of a collection of NLP programs grounded in *conceptual dependency* (CD) theory [Schank 1975]. In these systems, meaning representations are composed of *semantic primitives*, whose rule-governed combinations are supposed to capture the content of natural language sentences, and with whom the systems perform some reasoning activities (e.g., for text summarizing or translating). Actions, for example, are decomposed into a configuration of primitive acts (with their number varying between roughly one and two dozen, depending on the particular version of the theory).

BABEL, in translating a CD representation into English, has to determine which word is most appropriate to express a certain semantic primitive. These being very abstract, there naturally arises a substantial choice task, which is managed by discrimination nets, or d-nets. For every primitive, such a net is designed, which amounts to a decision tree with words on the leaves and procedures for path selection attached to the nodes. The procedures are arbitrary Lisp functions that make their decisions mostly by inspecting the context of the considered primitive in the CD formula. For example, the d-net for the primitive act INGEST, which denotes the activity of animate beings entering some sort of substance into their bodies, differentiates between the verbs *eat, drink, ingest, inhale, take (medicine),* and others on the basis of a sequence of queries regarding the substance being ingested. While this approach is not without problems (for instance, the unrestricted, hence informal, nature of the decision procedures at tree nodes has been criticized), the overall idea became quite popular: Words were considered as having a *core meaning* (in BABEL, the semantic primitive) plus some *conditions of use*, represented in the decision tree on the path from the root to a particular leaf.

Many subsequent generation systems have employed the d-net approach in one or another variant; the COMET system [McKeown et al. 1990] is one of them. The generator is based on Functional Unification Grammar (FUG) and produces text with integrated graphics through a series of unification steps. Before a meaning specification is passed to the unification grammar proper (for text production), it is enriched with lexical information and directives for grammatical structure. While this step is also controlled by the unification mechanism, a provision is made to leave the formalism and call arbitrary Lisp procedures for making more fine-grained word choices. For example [McKeown et al. 1990, p. 128], when the concept C-TURN (representing turning a knob on a radio) is lexicalized, a Lisp procedure queries the knowledge base as to whether the knob is one with discrete positions, and if so, the word *set* is chosen, otherwise *turn*.

The DIOGENES system [Nirenburg and Nirenburg 1988] uses a somewhat different representation mechanism: for every lexical item, a *frame* is defined that specifies the concept the item expresses as well as certain restrictions on particular roles of that concept. For instance, the frame for the word *boy* has its concept slot filled by 'person', and additional slots prescribe 'sex' to be 'male', 'age' between 2 and 15, etc. While the information is distributed in a different way (across the frames of the words), the result nevertheless resembles a discrimination net: The set of frames representing words that are linked to the same concept practically amounts to a net rooted in that concept, and we recognize the notion of "core meaning plus conditions". However, in a proper d-net, the *process* of selecting a word is exactly prescribed: decisions are made following the tree top-down. With the set of frames, a separate decision procedure needs to examine the slots of all the frames and filter out the inadequate ones; the search effort of finding lexical candidates can be enormous (see [McDonald 1991]). And finally, a d-net implicitly guarantees coming up with an answer, i.e., a word, because strict decisions are made at every node, and at every leaf there is a word. When the information is spread over a number of frames, on the other hand, there is no guarantee that all combinations of slot/value pairs are exhaustively covered—it might happen that a particular configuration of a concept instance does not match any of the word frames. To prevent this from happening, DIOGENES applies a numeric "meaning matching metric": on the basis of importance values that are associated with the slots, the metric computes the best match, i.e., the word whose overall slot-values come closest to the original specification. This process, called "nearest neighbor classification", restores the robustness of the lookup-process, but the assignment of numerical values and their subsequent arithmetical combination are difficult to motivate.

2.4.2 Taxonomic knowledge bases and the lexicon

As pointed out above, the discrimination net originated in the 1970s, in the context of NLP systems based on relatively few and therefore highly abstract semantic primitives. More recently, such systems have become less popular, as, for example, McDonald [1991, p. 230] observed: "Applications with this style of representation are increasingly in the minority (having been displaced by designs where the comparable generalizations are captured in class hierarchies or taxonomic lattices)." In taxonomic knowledge bases, objects (corresponding to nouns in language) as well as actions (corresponding to verbs) are organized in *is-a* hierarchies, where subordinate concepts inherit the properties of their superordinates. Depending on the representation language and on the design goals for the KB, additional relations (or *roles*) can be defined between concepts, such as *part-of*. In effect, with these hierarchies established as a de facto standard in knowledge representation, the idea of fully decomposing semantic definitions into minimal entities has been dispensed with.

As a consequence, KB designers defining an inheritance hierarchy are typically tempted to use natural language as a guide and define concepts only if there is a word for them in their own language. Thus, the problem of linking concepts to words may be reduced to a simple one-to-one mapping, which in fact happens in many systems: given a "suitably" designed KB, i.e., one oriented towards the lexicon, the lexical choice problem vanishes altogether; but with it vanishes the flexibility and expressiveness of the generator.

In principle, though, the 'grain-size' of the concepts in the KB is entirely up to the designer, and the relation between concepts and lexical items may be more elaborate. For example, there may very well be *named* and *unnamed* concepts in a knowledge base. In general, we cannot expect an isomorphism between lexical and conceptual structure [Novak 1993], and therefore a flexible link is required. In the following, we examine a few approaches where the interface between a taxonomic KB and the lexicon is more complex than a straightforward one-to-one mapping.

The 'lexical option generator' proposed by Miezitis [1988] assumes a frame-like input representing the concepts to be expressed and a taxonomically organized lexicon. Using a variant

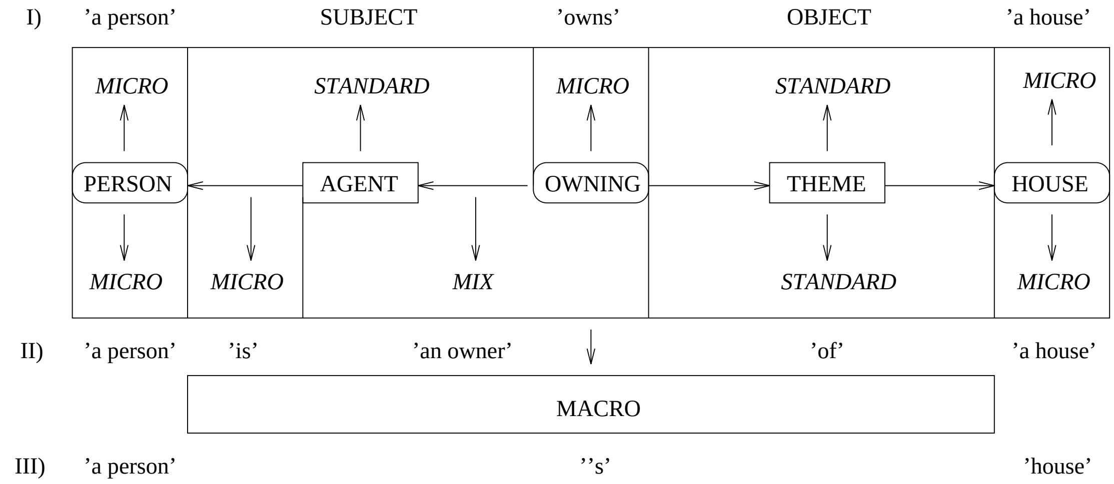

Figure 2.1: Lexicalization with 'zoom schemata' (from [Horacek 1990b])

of *marker passing*, the input is matched against the lexicon to determine the various options for expressing parts of the input. The result is a set of lexical items along with pointers to those sections of the input frame that the items cover. The next step for language production is to select pieces that together cover the complete input frame and that can be combined into a syntactically well-formed sentence. By organizing the lexicon taxonomically, it is possible to make finer distinctions in the lexicon than those made in the conceptual knowledge base underlying the system. Example [Miezitis 1988, p. 58]: if the input frame represents an INGEST action[5] and includes the slot (MANNER FAST), the lexical option generator will produce (also considering other parts of the input frame) the words *eat* and *fast*, covering different parts of the input; but it will also produce *gobble*, covering both parts together, because the INGEST-node in the lexicon has a subordinate node associated with *gobble*, which has a MANNER role pointing to FAST. In short, the knowledge base from which input frames are produced need not be aware of specific lexical items like *gobble*, but the lexicon is and can therefore propose that word as an alternative to expressing the different aspects of the input separately. Miezitis does not explicate the relation between the 'world knowledge' base and the lexical KB, but clearly all concepts residing in the former also have to exist in the latter for the process to work. This raises the issue of redundant storage, which in general ought to be avoided where possible.

A similar approach to lexicalization as pattern matching is presented by Nogier and Zock [1992], who work with the formalism of conceptual graphs [Sowa 1984]. The matcher successively replaces sub-graphs of the conceptual representation with lexical items and thereby produces a new graph representing syntactic structure; thus, the task of the lexicon is to relate concepts to syntactic entities. Since the matching sub-graphs can be more or less complex, the scheme allows for producing a variety of lexical paraphrases, for example, verbs incorporating the meaning of accompanying adverbs.

In discussing the generation component of the WISBER system, which is also based on a KL-ONE-like representation language, Horacek [1990a] examines possible relations between conceptual and lexical knowledge. He observes that the meaning of lexical items does not always correspond nicely to the meaning of KB concepts and that therefore the mapping from a conceptual representation to a set of lexical items can require restructuring work. Specifically,

[5] This INGEST does not correspond to the Schankian CD primitive mentioned earlier.

Horacek generalizes the word–concept mapping and proposes that not only lexical items but also grammatical functions (agent, instrument, etc.) and syntactic features can be mapped onto different types of concept configurations. He suggests the following four 'zoom schemata' that associate linguistic objects with various configurations (cf. figure 2.1): The MICRO schema maps a single concept or role node; the STANDARD schema applies to a concept and both its adjacent links, and the MIX schema to a concept, a role, and the link connecting the two; finally, the MACRO schema covers a concept, two associated roles, and all the links. The figure illustrates how different sentences can result from applying different combinations of schemata. In WISBER, all possible mappings are produced, and a unification-based algorithm determines a subset of lexical items and functions that together cover the complete input structure. The grammar then builds a sentence out of them.

The KING generator [Jacobs 1987] uses the knowledge representation language ACE, which was developed specifically for modelling the interactions between linguistic and conceptual knowledge, with emphasis on the use of inheritance for exploiting generalizations. KING uses a KB that taxonomizes not only concepts but also linguistic objects (e.g., various kinds of verb phrases) and associates them with one another. For example, simple events are linked to verb–object relations, with subtypes of both also being in more specific correspondences: transfer-events are associated with verb–indirect object relations, where the recipient of the transfer maps onto the indirect object. The association of lexical items and concepts is one special case of this general scheme. Generation proceeds by first *mapping* from conceptual to linguistic structures (according to the specified relations in the KB), then *selecting patterns* that govern constituent order, and finally *restricting* patterns to enforce syntactic constraints. The first mapping stage may also involve mapping conceptual structures onto others, corresponding to different *views* expressing the same event. Thus, while the concept–word link is fairly simple (single lexical items are attached to a subset of the concepts), the generator is nonetheless capable of producing a range of textual variations by means of conceptual mappings in the fine-grained representation of event structures.

2.5 Placing lexicalization in the generation process

A generator has to make decisions of various kinds, like ordering and structuring the material, or selecting grammatical constructions. Naturally, lexicalization has to occur at some point or another in the overall process; deciding on this point also implies a decision on its possible inter-dependencies with other generation decisions.

2.5.1 Lexical and other choices

The common role of lexical choice is to serve as a link between sentence-size input to the generator and the grammatical decision-making.[6] A conceptual structure is mapped onto lexical items: verbs are chosen to express events, and as a consequence, semantic roles used in the knowledge representation are mapped onto corresponding syntactic functions (e.g., an agent is usually realized as the subject). Thereby, the properties of lexical items come to constrain the syntactic realization of the sentence[7]—roughly speaking, the generator first selects the words and then figures out how to put them together. Quite obviously, this procedure presupposes that

[6] Cumming [1986, p. 11] concludes this in her survey as do McDonald [1991, p. 229] and Matthiessen [1991, p. 277] in their analyses of the role of lexicalization.

[7] For a detailed discussion of the interaction between lexical and syntactic decisions with specific English and German examples, see Mehl [1995].

the words can be combined *at all*; usually, generators implicitly assume things will work out: the range of possible input specifications is a sufficiently restricted type of predicate/argument structure so that it corresponds closely enough to linguistic realizations. If one seeks a more elaborate treatment of the relations between lexicon and grammar, some provisions for backtracking from earlier word choices have to be made. To rephrase the issue a little, the point of accessing the lexicon depends on how much *formative* information is encoded therein; Hovy [1988b], for example, argues the extremist view of placing *all* such information in the dictionary, thereby eliminating the need for a separate grammar.

2.5.2 PENMAN

One of the most successful sentence generators nowadays is the PENMAN system [Penman 1989], which uses as input an expression formulated in the Sentence Plan Language (SPL) [Kasper 1989] and produces an English sentence corresponding to that specification. Penman is built around the systemic-functional grammar NIGEL, which is organized as a large network of choice points, the so-called 'system network'. When generating a sentence, the network is traversed for every 'rank' to be realized, from higher-level clauses to lower-level groups and phrases, and during the traversal, features are collected that collectively determine the properties of the utterance to be constructed.

Here, lexical choice is related to the grammar as follows: At the end of every traversal of the grammar, a word is looked up that is associated with the concept given in the input SPL expression and at the same time matches the set of features. Again we have the underlying assumption that "things will work", i.e. that there will be a suitable word available. But in the NIGEL case, the underlying theory does in fact warrant the procedure, because both lexical and grammatical decisions are made with respect to the same semantic 'upper model', a semantic ontology that we will describe below. The decisions made in the grammar are largely based on this ontology.

At first glance, the lexicalization scheme employed by PENMAN appears to actually interleave grammatical decisions and lexical choice, but in fact there is not much of a *choice*: words are directly associated with concepts that appear in the SPL expression, and selection is governed solely by the grammar, where the required syntactic/functional features are the only criterion. Although words are determined at the end of every pass through the grammar, and hence there is a *temporal* interleaving, a word selected on a higher rank cannot influence later decisions on lower ranks. This crucial limitation follows directly from the viewpoint of systemic theory, as mentioned above: lexical decisions are not granted a distinct status; thus they have no way to exercise influence on other decisions. In theory, the lexicogrammar is an elegant idea, but in practice the diminished role of the lexicon reduces the expressiveness of the generator.

The system to be developed in this thesis will use PENMAN as a front-end generation module but make an important change to the place of lexicalization: We will choose words before activating PENMAN, specifically: in the process of *building* the input expressions to PENMAN from a more abstract specification.

The Upper Model Since our system will make use of the idea of the Upper Model, we here discuss in more detail its purpose and function. The Upper Model (UM) [Bateman et al. 1990] is an ontology rooted in systemic-functional linguistics [Halliday 1985] and was first applied to text generation in PENMAN.

The central requirement for SPL expressions, i.e., the input to PENMAN, is that each entity in that expression needs to be associated with a UM type. To this end, the domain

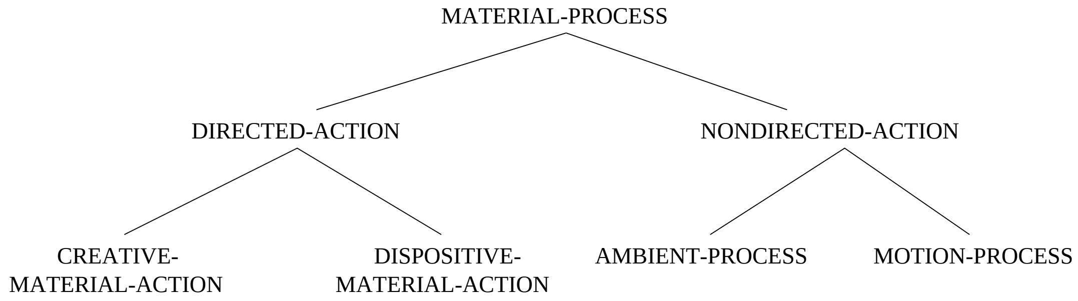

Figure 2.2: Small excerpt from Upper Model

model concepts, which are in practice used in an SPL, need to be linked to appropriate UM concepts. On the basis of the UM type of the entity, the grammar knows how to verbalize that entity (some other sources of information also play a role, but the UM is clearly the central engine). Hence, the UM can be characterized as mirroring the distinctions made in surface linguistic realizations: Typically, any two distinct UM types correspond to some difference in English sentences.[8] Or in other words, any UM concept is associated with clearly specifiable *lexicogrammatical*[9] consequences. The idea is to define a level of abstraction midway between linguistic realizations and conceptual representations—something that is very useful to text generation.

A glimpse of the UM Linguistic theory (or rather, any of various linguistic theories) declares the verb as the most prominent constituent of a sentence, around which the other elements are assembled. Correspondingly, the central element of an SPL expression is a *process*, with which certain *participants* and *circumstances* are associated.[10] Participants are considered as essential to performing the process, whereas circumstances give additional information like temporal or spatial location, the manner of performing the process, etc.

Processes are characterized by typical verbalization patterns, and the knowledge about these regularities resides within PENMAN's grammar. Given an input SPL, PENMAN inspects the UM types of the main process and the participants and circumstances, and derives the possibilities of realizing that particular configuration in language. At the heart of PENMAN's operation is thus a thorough classification of processes that reflects exactly the distinctions made by the target language. The processes form an important sub-hierarchy of the UM, which altogether consists of several hundred concepts that are encoded in LOOM.

The original UM, as developed for PENMAN, is thoroughly documented by Bateman et al. [1990]. To illustrate some of the categories, we give an example from that paper. Figure 2.2 shows a small fragment of the PROCESS hierarchy, namely the subtree of MATERIAL PROCESSES, which our generation examples given later will make use of. This family of processes can be

[8] More accurately, realizations *with different meaning* stem from different UM types. Henschel [1993] points out that "disjoint concepts in the UM do not necessarily correspond to disjoint sets of surface sentences—only to disjoint semantic perspectives on them. The interface between the UM and the grammar should be written in such a way that it is possible in some cases to generate the same sentence from different semantic input."

[9] In systemic-functional linguistics, there is in theory no separation between lexicon and grammar; both are intertwined in the network of choice points ('systems'), the lexicogrammar.

[10] The distinction between participants and circumstances is made in one way or another in any linguistic theory, where the realizations as surface constituents are, for example, called *complements* and *adjuncts*. The former are seen as being *subcategorized* for by the verb, whereas the latter are not. We will discuss these notions in section 3.7.

characterized by the fact that English verbalizations of them in present tense typically use the progressive form, as in *the house is collapsing* (unmarked) as opposed to *the house collapses* (marked). They typically involve the participant roles ACTOR and ACTEE but differ in terms of constraints on the types of the role fillers, and with respect to their realization in language.

NON-DIRECTED ACTIONS do not involve external agency and are mostly intransitive. If they are transitive, though, then the object is not created or affected by the process, as in *I am playing the piano*. With such processes, the ACTEE is not a genuine participant, but rather an elaboration of the process. Verbs falling into this category are those of movement, of expressing skills, as well as support verbs like *to take* as in *take a shower*. The UM explicitly represents MOTION-PROCESSES and AMBIENT-PROCESSES, which express weather conditions, and acknowledges that more classes would be needed here.

DIRECTED-ACTIONS, on the other hand, are always transitive, and they involve an external agent of the process. CREATIVE-MATERIAL-ACTIONS *create* their actee, as in *Mary baked a cake*. They can always be paraphrased by the verbs *to create* or *to make*. DISPOSITIVE-MATERIAL-ACTIONS, on the other hand, *affect* an already-existing actee in some way, as in *Eunice ate the cake*.

The idea of using Upper Models for language generation originated with PENMAN and has since been used in several applications based on it, e.g., in DRAFTER [Vander Linden and Scott 1995]. And independently of PENMAN, UMs are used in other generation systems as well, for instance in SAGE [Meteer 1994] or in PROVERB [Fiedler and Huang 1995]. Also, in an evaluation of IBM Germany's LILOG project, Novak [1991] pointed out the importance of separating linguistic from non-linguistic knowledge taxonomies (which had not been done in LILOG) and advocated employing UMs as a solution.

2.6 Multilingual generation

As mentioned in chapter 1, multilingual generation (MLG) is, surprisingly, a line of research that developed only quite recently. Probably the "oldest" working system is FOG, which produces English and French weather forecasts in Canada [Goldberg et al. 1994]. Due to the limited domain and restricted vocabulary, though, lexical choice is only a minor issue in that system; specifically, the requirement of multilinguality does not pose additional problems—the lexical selections for English and French are almost exactly parallel, except for the different syntactic environments.

The idea of using Upper Models to abstract over language-specific realizations has been extended to multilingual environments (e.g., [Bateman et al. 1991, Bateman et al. 1994]), but this work does not concentrate specifically on lexical matters.

In the absence of "lexical results" from MLG, we turn to *interlingual machine translation*, where the problems are similar. Dorr [1993], for example, systematically discusses different cases of divergences between languages, which have to be handled in an interlingual MT framework in much the same way as in MLG. We will return to the notion of divergences in chapter 4, and compare our own approach to multilinguality with that of Dorr in section 10.2.

2.7 Conclusions: making progress on lexicalization

Criteria for choice We pointed out that the range of factors influencing lexical choice are far from being well-understood, and characterizing their various interactions is, correspondingly, a wide-open question. NLG research has investigated a number of isolated choice criteria but did not account for their interactions in lexicalization; an exception was PAULINE [Hovy 1988a], but this system masks all the decision-making process in an array of interacting procedures.

This thesis will not address the question of selecting the most appropriate paraphrase in a specific situation of utterance, i.e., that of *tailoring* an utterance; rather, we will investigate two choice criteria and demonstrate that their treatment can be integrated into our overall system architecture. Thus, we follow the path suggested by Cumming [1986, p. 26]:

> We're a long way from having natural language generators that have the degree of control over any level of linguistic choice, grammatical or lexical, that a serious treatment of these considerations would entail; but we can design our systems so that such distinctions will be able to be accommodated when we have the analyses to support them.

To these ends, we will start from the observation that while the factors for lexical choice are commonly labelled as *constraints*, most of them should rather be seen as *preferences*: as soon as connotations are represented in the lexicon and some sort of pragmatic goals are part of the input to the generator, conflicts are likely to arise. For one thing, particular stylistic goals might or might not be achievable, therefore the generator can *try* to fulfill these goals, but there is no guarantee (e.g., for producing a formal sentence that refers to the concept MAN, the system can choose the formal word *gentleman*, but for LASER-PRINTER there are no such options to convey the stylistic tone). And in addition, lexical choices as well as other linguistic decisions are made with a number of different viewpoints in mind. When a generator is confronted with a number of simultaneous goals, the task is to satisfy all the requirements as far as possible; hence individual choices become a matter of preferring one option over the others, under the influence of a range of parameters that may well be in conflict. For example, if one goal is to produce concise text, and another one calls for formal words, then the generator will have to compromise occasionally, because formal words and phrases tend to be more lengthy than informal or slang vocabulary.

Linking concepts to words A typical scenario for language generation nowadays is a conceptual representation, based on some taxonomic knowledge base, which has to be verbalized. The easiest way to associate lexical items to concepts is a one-to-one mapping, and many systems follow that path. But moving away from a strict one-to-one mapping between concepts and words is an absolute necessity for any generator that is expected to permit variety in text output that needs to be tailored to different purposes or to different audiences.

In general, the task of the word-concept link is to mediate between the granularities of the KB and the lexicon: the problem is trivial when they are identical, but typically there are good reasons to make distinctions in the lexicon finer than are required for reasoning purposes in the KB. The discrimination network (in whatever implementational variant) is an instrument for making such fine-grained word choices, but it has crucial limitations: It does not offer a way of finding more or less specific words to express the concept, because there is no knowledge about subsumption relationships between concepts and words, and between different words. Encoding such relationships in a decision tree (which a discrimination net amounts to) would be an extremely cumbersome task. Furthermore, the discrimination net is always attached to a single concept, hence it cannot account for the need to map whole *configurations* of concepts and roles to lexical items.

As a step into this direction, we have described Horacek's [1990a] four 'zoom schemata', and similarly Miezitis's [1988] 'lexical option generator' that worked with spreading activation to find the best match. Neither are concerned with the subsumption relationships, and Miezitis's matching mechanism lacks the declarative flavor that computational linguistics has come to value. Besides, both these approaches as well as that of Nogier and Zock [1992] (which is similar to Horacek's) map the cnceptual units directly to syntactic objects; hence there is no account of lexical semantics, and the approaches do not lend themselves to multilingual generation, because the complete mapping from concepts to surface sentences would need to be duplicated for every target language.

NLG has traditionally treated the concept associated with a word as its sole "meaning" and neglected to account for other aspects of lexical semantics. In addition to the word–concept link, we will see the contribution that a word can make to sentence meaning as a separate entity; by dividing it from the conceptual content, it is possible to posit lexical rules that derive certain readings of words (in particular verbs) from others, and thereby to capture generalizations about the behavior of lexical classes.

The point of lexicalization in the generation procedure The majority of language generators have taken lexicalization as the first step, and grammatical decisions for linking these words into well-formed utterances follow behind. We subscribe to this view, too, but it needs to be ensured that all the lexical choices, which have been made independently of each other, *can* in fact be syntactically combined. Many systems have made this assumption just implicitly.[11] To this end, we will employ a level of semantic representation that is built up in the lexicalization stage, and whose "expressibility" is guaranteed by grammatical knowledge.

A different path has been taken by Elhadad [1993], who shared several of our motivations, predominantly the goal of increasing the lexical variety that generators can produce. He put all the additional efforts into the surface grammar, and thereby gained elegance of description; on the other hand, modularity is lost: when multiple target languages are to be generated, all the work has to be re-done in each grammar module. For this reason, we opt for a different approach that separates language-neutral from language-specific levels of representation and leaves specific grammatical decisions to the end of the process.

Designing a new architecture for NLG In conclusion, our goal is to design an architecture that combines the strengths of some earlier ideas and at the same time overcomes at least some of their shortcomings. Specifically, we need a system architecture that

- is based on a domain model that is suitably structured to allow for producing a range of lexical paraphrases;
- makes a clear transition from a non-linguistic input representation to a linguistic level of representation;
- determines the pool of lexical *options* very early in the process, so that other decisions can be based on it;
- can account for various dimensions of lexical choice, i.e. translate generation parameters into lexical decisions;
- allows for flexible word–concept mapping and accounts for subsumption relationships;
- uses lexical entries that are rich in information and separate the various realms that information belongs to;

[11] This problem is, so to speak, the sentence-planning version of the "generation gap" that Meteer [1992] has dealt with on the level of text planning.

- operates on declarative representations and does not hide decisions in procedures;
- lends itself to multilinguality, in fact does not need any special machinery for producing multlingual output.

To achieve such a design, it is important to strengthen the role of lexical semantics in NLG. In the next chapter, we review some contributions from the linguistics literature, which will be used later in developing our generation system.

Chapter 3

Lexical semantics

This chapter introduces the topic of lexical semantics and then reviews a range of contributions from linguistic research regarding that topic, which will in later chapters be used to motivate the design decisions made in building our generation system.

3.1 Introduction

> The academic name for the study of meaning is semantics. It is not an easy subject, and beginning students can be misled because two different intellectual enterprises go by that name. One is philosophical semantics, dignified and inscrutable; its goal is to formulate a general theory of meaning. The second is lexical semantics, grungy and laborious; its goal is to record meanings that have been lexicalized in particular languages. [Miller 1991, p. 148]

Assuming that Miller's partitioning of the semantic arena is correct, the work presented in this thesis clearly falls into the second, 'grungy', camp. In this chapter, we will be analyzing the meaning of words and trying to uncover the differences and commonalities between similar words—within a language, and across languages.

There ought to be a word about philosophical semantics, though. We are not brushing it aside because of dislike or lack of interest; the subject is merely outside the realm of this thesis. When the task is to generate language from an underlying knowledge base, then anything to be said about semantics is anchored within a fixed representational system—namely that of the knowledge base. In our terms here, the domain model that will be introduced in chapter 5 defines the playing field for semantic analysis: word meaning has to be defined first and foremost with respect to that domain model. Philosophical semantics, seeking general theories of *meaning*, would have to do rather with investigating the relationship between the domain model and the "real world", and these issues are beyond our present concern.

Concentrating then on word meaning, there are two, complementary, lines of thought for dealing with it: one can aim at *defining* word meaning exhaustively, that is, in terms of a fixed set of primitive elements, or one can collect similar words and investigate merely the differences among them, without striving for complete decomposition. The latter leads to *relational* theories of lexical semantics, often also called *structural*.

3.2 Relational theories of word meaning

The extreme structuralist view, brought to popularity by Saussure [1915/1966], is that the meaning of a linguistic unit cannot be determined by looking at that unit in isolation, but only by scrutinizing its relationships to other units. In this way, the vocabulary of a language is seen as a system that defines each individual word in terms of the relations it has to other words. And indeed, investigating these relations is at the center of much work in lexical semantics (e.g., [Cruse 1986], [Evens 1988]). The four most widely accepted relations are the following:

- **Synonymy** Most authors agree that true synonymy between two words of a language does not exist. However, as soon as we extend this relation beyond its traditional boundaries and apply it across languages, we can call translation-equivalent words like the English *bear* and the German *Bär* synonyms. Cruse [1986, p. 265] suggests that a language exhibits different 'degrees of synonymy': "*Settee* and *sofa* are more synonymous than *die* and *kick the bucket*, which in turn are more synonymous than *boundary* and *frontier*."

- **Antonymy** Often, this relation is treated as a general term for lexical opposites, as for instance *man / woman*. Some, like Cruse [1986], use it in a restricted sense, as applying only to gradable adjectives (*large / small*) and some adverbs (*quickly / slowly*), where antonyms denote degrees of some variable properties such as length, speed, or weight.

- **Hyponymy** The relation of class inclusion is difficult to define precisely. Least problematic are nouns, where X is a hyponym of Y iff the sentence *This is a X* entails *This is a Y*, but not vice versa. Class inclusion can be investigated for verbs as well, but is hard to diagnose with general methods. Typically (and vaguely), if verb X is a hyponym of verb Y, then doing X is a specific manner of doing Y; *staring* is one particular way of *watching*. But there are differentiae other than manner. For instance, whenever there is a *murdering* going on, there is also a *killing*, and the additional information conveyed by the former is that of volitional agency.

- **Meronymy** Equally difficult to specify is the part–whole relationship. For instance, Cruse [1986] devotes several pages to the distinction between *parts* of something and *pieces* of something; the parts constitute some ordered arrangement of the whole, whereas pieces do not. A typewriter, for example, can be regularly disassembled into its parts, or it can be arbitrarily sawn into pieces. In linguistics, meronymy is of interest because of, amongst others, its role in choosing determiners: when an object is within the focus of discourse, its parts can be referred to with a definite article, even upon their first mention.

A number of other relationships are discussed in the literature, and the reader of, say, Evens [1988] begins to wonder whether the line between philosophical and lexical semantics can indeed be drawn as clearly as Miller [1991] suggests. It seems that the branch of linguistics concerned with relational theories of the lexicon is aiming to explain the world as such, for every possible relationship between entities in the world is seen as a *lexical* relationship.

From the perspective of knowledge-based NLG, this approach is of limited help. Here, our goal is to separate the language-neutral facts from the language-specific idiosyncrasies—or in other words, the general concepts from the specific words. Under this view, the facts that wolves are animals and automobiles consist of certain parts should be represented on the conceptual level, insofar as they hold for speakers of different languages and are thus independent of lexical items. In a multilingual system knowing only lexical relations, we would have to replicate the hyponymy relationship between *mammal* and *wolf* as also holding between *Säugetier* and *Wolf*, and so forth for other languages. And the same would happen with meronymic and other relations; but duplicating all this information would clearly miss the point.[1]

Where exactly the line is to be drawn between concepts and words can only be decided empirically, by comparing the distinctions that different languages make—or even distinctions within a single language, when paraphrases distribute the units of meaning differently. Notwithstanding our critical remarks, there certainly are some genuine lexical relations: The 'collocational' phenomena we have discussed in chapter 2 are affinities between specific lexemes and have to be represented as such. However, we concentrate here on the interface between the concept taxonomy and the lexicon and thus neglect the collocations.

3.3 Decomposition

The idea of systematically decomposing words into elementary units of meaning was promoted notably by Katz and Fodor [1963], who suggested dividing these units into semantic *markers* and *distinguishers*. The markers were supposed to be the units that recur in the definitions of many words, and that constitute the 'systematic' part of word meaning, whereas the distinguishers were names for the remaining differences that are supposed to be idiosyncratic to some particular group of words. The theoretical feasibility of separating markers and distinguishers has been questioned many times (for an overview, see [Lyons 1977]), but still, the notorious example "bachelor = man + unmarried" has been taught to countless students of linguistics. The idea of decomposing word meaning into primitives found rather radical formulations in the theory of Wierzbicka [1980] and, limited to verbs, in the Conceptual Dependency theory of Schank [1975], but ultimately, these and other approaches never got beyond explaining quite simple examples. The 'distinguishers' of Katz and Fodor were meant as idiosyncratic and exceptional, but their significance was underestimated: the goal of explaining as much as possible only with systematic 'markers' was not accomplished to an extent that would warrant describing the idea as successful. And, in parallel, a second line of attack on meaning decomposition gained strength; it posed the question of how one could actually justify the existence of semantic primitives, beyond just *postulating* them to be mental objects. If, so goes the argument, the meaning of words, which are symbols, are explained solely with a number of so-called primitives, which are also symbols, then what has been gained? After all, the 'primitive' symbols in turn need to be explained. According to this view, the meaning of primitives ultimately needs to be accounted for by jumping out of the symbolic system. These matters are nowadays discussed as the *symbol grounding problem* (e.g., Harnad [1990]).

However, this does not imply that there is no point at all in decomposing word meaning. Identifying non-idiosyncratic meaning components is desirable for reasons of linguistic description: When some semantic feature can be shown to correlate with some particular syntactic behavior of a class of words, then there is support for the assumption that syntactic behavior is not arbitrary but follows from some semantic commonalities. This is an important line of research, for instance, in explaining the *alternation* patterns of verbs (see section 3.8).

[1] There are two theoretical positions compatible with rejecting the "all is lexical" view. One is that of conceptual realism: Taxonomic, meronymic, and other relations hold in *the world*, and the different languages merely mirror them; the conceptual representation in the KB then literally represents the world. The other is a cognitive position: The mammal-wolf relationship or the fact that we tend to divide things into certain parts are due to principles of cognition, i.e., the way in which we *perceive the world*, and these are assumed to be largely shared between human beings belonging to different cultures and speaking different languages. As an example for a disagreement between similar cultures, note that in English, *potato* is a hyponym of *vegetable*, whereas in German, the corresponding *Kartoffel* is excluded from the category GEMÜSE. For reasons of this kind, we lean towards the cognitive position, but this does not really make a difference for the thesis.

A well-known approach aiming at explaining certain aspects of the semantic behavior of words and their correlations with syntactic features is that of Jackendoff [1983, 1990]. He developed *lexical-conceptual structures* (LCSs) as a scheme of semantic representations that are systematically linked to syntactic structure. LCSs have gained quite some popularity, especially in North American linguistics (e.g., [Rappaport and B. Levin 1988]) and computational linguistics (e.g., [Nirenburg and L. Levin 1992], [Dorr 1993]). The central theme of the LCS approach is a commitment to decompose word meanings in a principled manner: If a primitive is recurrent in such a way that it appears to be responsible for some specific semantic and/or syntactic behavior of a class of words, then it can be accepted into the system. To give just two examples for primitives, the existence of CAUSE can be motivated on the grounds that many verbs can occur in two different configurations—one where an event takes place by itself, and one where it is caused by an external agent. Accordingly, the presence of this agent is syntactically realized as subject of the sentence. Similarly, the primitive INCH (for 'inchoative') works as a function that is applied to a state and yields the event of something gradually moving into that state. Again, many verbs have an inchoative as well as a non-inchoative reading, which appears to warrant the acceptance of the primitive.

At the same time, in an NLG framework, linguistic representations do not exist for their own sake but are typically linked to some conceptual representation, which is used by a system that performs reasoning. This can impose additional requirements on decomposition: a feature that is relevant for a reasoning operation on the 'conceptual' level is to be introduced as an entity on this level, and the representations of lexical meaning then have to respect its existence, since they have to be linked to the conceptual representations. In short, the role of decomposition should not be that of trying to build a complete ontological symbol system, but that of introducing a primitive precisely at those points where it is relevant either for reasoning purposes, or for achieving differences (if they are desired) in monolingual or multilingual verbalization.

To uncover such differences between verbalizations, we return to the method of systematically comparing similar words, as in the relational accounts explained above. While many of the results of their research are of little use for us, the method should not be dismissed. Thorough comparisons can lead not only to an inventory of lexical relations, but also to sets of features that distinguish similar words. This approach was first systematically undertaken in the *Wortfeld* ('lexical field') analyses by Trier [1931], and later by Weisgerber [1950], who emphasized that such lexical fields have a significant impact on how an individual language structures the way of perceiving the world. A lexical field is a set of words that demarcate each other and collectively cover some 'semantic area'. The method of *componential analysis* has developed this notion and followed the idea of characterizing the lexicon of a language with a limited inventory of semantic features. As an example, James [1980] analyzes a number of English cooking verbs (*cook, boil, simmer, fry, roast, toast, bake*) in terms of the features *with water, with fat, in oven, contact with flame*, and *gentle*. Also, the method of lexical-field analysis can be straightforwardly extended to cross-linguistic comparisons, the so-called *contrastive analyses*. There is no principled difference between examining (near-) synonymy within languages and between languages. James [1980], for instance, goes on to compare the English cooking verbs to the German *kochen* (three different senses), *braten, rösten*, and *backen* using the same set of features.

Lexical-field analysis, both intra-lingual and contrastive, has traditionally been applied to content words, but can in fact be extended to function words as well. In [Stede 1994], a contrastive analysis of German and English *discourse markers* that signal a 'substitution' relationship in English is given; Grote, Lenke, and Stede [1995] do the same for markers signalling 'concession' in both languages.

Crucially for lexical-field analysis, the next step (which Trier or Weisgerber did not undertake) has to be sorting the features into the different realms to which they belong, for instance into those of *denotation* and *connotation*.

3.4 Denotation versus connotation

The distinction to be discussed now is that between semantic and stylistic features, or, equivalently, *denotation* and *connotation*. It has been made in semantic theory at least since the Middle Ages, and in a wide variety of ways.[2] In a linguistics dictionary, Bussmann [1983] defines the denotation as the constant, basic meaning ('Grundbedeutung') of a word that is the same over all possible contexts and utterance situations. Connotations, on the other hand, vary from speaker to speaker: emotive, stylistic overtones that can be superimposed upon the basic meaning and tend to resist a context-independent definition. Part of the task of linguistics and computational linguistics, though, is to overcome this rather "pessimistic" explanation of connotation and to identify at least some of the features belonging to that cloudy realm, so that they *can* be characterized independently of specific contexts.

The division between denotation and connotation can be stated in terms of truth conditions, as by DiMarco, Hirst, and Stede [1993]: "If two words differ semantically (e.g., *mist, fog*), then substituting one for the other in a sentence or discourse will not necessarily preserve truth conditions; the denotations are not identical. If two words differ (solely) in stylistic features (e.g., *frugal, stingy*), then intersubstitution does preserve truth conditions, but the connotation—the stylistic and interpersonal effect of the sentence—is changed."

If a lexical substitution does not preserve truth conditions, then there is a change in denotation; this much can be said. While this condition is necessary, it is not sufficient, because often the border between denotation and connotation is not clear-cut. DiMarco, Hirst, and Stede [1993] consider the example *He* {*arranged* | *organized*} *the books on the shelves* and state that "both choices mean 'to put things into their proper place', but *arrange* emphasizes the correctness or pleasingness of the scheme, while *organize* emphasizes its completeness or functionality [*OALD* 1989]. Variations in emphasis such as these seem to sit on the boundary between variation in denotation and variation in connotation; in the example sentence, intersubstitution seems to preserve truth-conditions—the two forms of the sentence could describe the exact same situation—but this need not be true in general: the *arrangement* might be incomplete, or the *organization* not pleasing."

Nonetheless, some classifications can be made. For one thing, certain lexical properties can be isolated as recurrent stylistic features. Standard dictionaries often list *formality* as a dimension along which similar words differ, and sometimes also note how "up to date" a word is, whether it is archaic or modern, maybe even trendy. With closer scrutiny it is possible to identify more dimensions of this kind, using the method of carefully comparing near-synonyms. While this area of research has barely begun to be explored (at least in computational linguistics), some preliminary results can be stated; section 7.5 will suggest a set of stylistic features that in many cases are useful for discriminating similar words with identical denotations.

A number of the semantic, or denotational, features can be encoded with the instruments of a taxonomic knowledge base (which will be introduced in chapter 5), by means of carefully defining roles for concepts and stating constraints on their fillers. But these methods have their limitations. For instance, the German word *ausbessern* (similar to *to mend*) applies to inanimate objects *except for* engines and machines [Schwarze 1979, p. 322]. There are, generally,

[2] For a comprehensive historical overview, see [Garza-Cuarón 1991].

three ways of dealing with this kind of situation. First, one could introduce a new level into the concept hierarchy below INANIMATE-OBJECT and separate MACHINE from OTHER-INANIMATE-OBJECT. This step has an ad-hoc flavor to it; but the reluctance to take it can be overcome if other words turn out to make the same distinction. If not, the specific idiosyncrasy can be dealt with either on the conceptual level by barring the general verb (here, *ausbessern*) from percolating downwards to one particular branch (here, MACHINE), or—if the idiosyncrasy does not pertain to semantic traits—on the word level by stating a collocational constraint, thereby leaving the word–concept mapping unaffected.

Another problem is that semantic distinctions are often not categorial at all (as with *ausbessern*) but deal with fuzzy boundaries. The difference between *forest*, *wood*, and *copse* is similar, but not quite identical to that between the German *Wald*, *Gehölz* and *Wäldchen* [DiMarco, Hirst, Stede 1993]. Representing such differences in a concept taxonomy would lead to an inflation of quite awkward concepts like SMALLISH-TRACT-OF-TREES or BIGGER-TRACT-OF-TREES. And finally, much lexical differentiation lies in emphasis rather than conceptual denotation; recall the example *organize / arrange*.

Being aware of such limitations, we will in this thesis explore the taxonomic approach and see how much it can do. For the other, non-taxonomic parts of lexical semantics, which we leave aside here, DiMarco and Hirst [1993] suggest an approach based on a study of dictionary *usage notes*.

3.5 Two-level semantics

The question of where to represent what differences between similar words leads us to consider the number of representation levels needed to account for lexico-semantic phenomena. One view is exemplified by Jackendoff's [1990] lexical-conceptual structures. Although we will borrow some of his representational decisions, we do not in fact share the basic assumption of his approach: that there be only one level of semantic description. Jackendoff insists that conceptual structure is essentially the same as semantic structure (more precisely, that the latter is a subset of the former), and posits that besides processing language, other cognitive operations can be explained on the very same level.

We think otherwise for the reason that several interesting semantic questions can best be dealt with when two separate levels are assumed. In support, here is a brief outline of the *Zwei-Stufen Semantik* ('two-level semantics') advocated by Bierwisch [1983].

The central motivation behind the work of Bierwisch is to explain certain kinds of 'regular polysemy' exhibited by lexical items. Consider his example (translated from German) *Faulkner is hard to understand*, which can be interpreted in the following ways:
(a) Faulkner's articulation (speech) is hard to understand,
(b) Faulkner's way of behaving is hard to understand,
(c) Faulkner's books are hard to understand.
Essentially, we are faced with different readings of the proper name *Faulkner* and the verb *to understand* and need to explain how coherent interpretations of the sentences come about. Bierwisch argues that the answer cannot lie in assuming a variety of separate dictionary entries for 'ambiguous' words like those above, or for a noun like *school*, which can be used in various related senses:
(a) *The school is located beside the playing field* – building
(b) *The school is supported by the community* – organization
(c) *School is boring to him only occasionally* – occupation
(d) *School is a central element of European history* – idea

Note that this polysemy has nothing to do with metaphor; Bierwisch is concerned solely with 'literal' meanings that are closely related, and not with far-fetched sense-extensions, nor with ambiguity between totally unrelated word senses like in *bank*.

Now, in a sentence like *The school is of great concern to him* it is rather unclear which of the senses is being intended. Bierwisch presents a series of arguments, which shall not be reviewed here, in support of the thesis that such a sentence should not be considered syntactically nor semantically ambiguous, and draws the conclusion that a separate level of representation, beyond semantics, is needed to capture the differences between the senses. He calls it the 'conceptual' level and argues that its structure need not be identical with the structure of natural language—which amounts to a flat rejection of Jackendoff's thesis on the identity of semantic and conceptual structure.

The semantic representation is thus assumed to be potentially underspecified, and the meaning of a word on this level is seen as a specification of the information that the word contributes to sentence meaning (in accordance with traditional compositional analysis). Yet it can receive multiple interpretations on the conceptual level. Bierwisch says that a word's meaning defines a "family" of conceptual units, from which an interpretation function in a given context selects the most appropriate one for constructing the conceptual representation. The example *school*, like *book* or *sonata*, is a word that can undergo a *conceptual shift* toward any of the four interpretations listed above. In a sentence, it is possible that the same word can undergo two different conceptual shifts: *This book, which John wrote, weighs five pounds*. But not all combinations are possible: ∗*The sonata that is lying on the piano is the most important genre of the Viennese classical music*. And in the following example the interpretations of the two clauses are dependent on one another: *Hans left school and went to the theater*. Both nouns can be interpreted as institutions (Hans changed his career) or as buildings (Hans spent an afternoon), but they have to be the same.

In summary, the two-level approach moves a lot of work out of the linguistic realm and into a 'conceptual' one. Lexical entries are often underspecified, thereby their overall number is reduced, and contextual parameters are supposed to aid an interpretation function in constructing the right conceptual representation.

Similar, at least in spirit, to Bierwisch's work is Pustejovsky's [1991b] conception of the 'generative lexicon', which also aims to reduce regular polysemy in the lexicon. Also, Nirenburg and L. Levin [1992] made a proposal to distinguish 'ontology-driven' from 'syntax-driven' lexical semantics, and blend two complementary perspectives from AI/ontological modelling and linguistics/syntax together. Nirenburg and L. Levin argue that semantics needs to be approached from *both* the syntactic and the ontological/knowledge level and that there is no point in fighting over which one is "better". On the syntax-driven side, they employ lexical-conceptual structures (LCSs) in Jackendoffian style, where the central task is in linking semantic participants to syntactic positions. "Syntax-driven lexical semantics helps *distinguish* meanings, but not represent them." [Nirenburg and L. Levin 1992, p. 10] Any task beyond explaining the mapping between surface sentences and LCSs falls into the realm of 'ontology-driven' lexical semantics, which is responsible for building up representations of texts that can be reasoned with. One task for this kind of semantic processing is to support disambiguation in language understanding, in cases where more knowledge is needed than that for argument-linking rules. Also, ontology-driven lexical semantics would be responsible for explaining synonymy, antonymy, and hyponymy between lexical items. Giving a number of examples of 'divergences' between languages (see chapter 4), Nirenburg and L. Levin conclude that the 'deep' representations of meaning should not follow the syntax nor the lexis of any particular language, since the same event can be expressed in rather different ways in different languages. As one consequence, the authors note that a verb hierarchy for an individual language need not coincide with the concept hierarchy needed to encode the underlying knowledge—a point that we will stress later in the thesis.

3.6 Aspect and Aktionsart

Since we will focus on generating verbalizations of *events*, we need to examine the linguistic means for describing them. The "branch" of linguistics most interesting to this aim is that of studying *aspect*. Here, the goal is to uncover the inherent temporal structure of occurrences[3], as found in the meaning of verbs. The central distinction, sometimes called *imperfective* versus *perfective*, is that between continuous occurrences without an internal structure (*to walk, to sleep*) on the one hand, and occurrences that develop towards some 'culmination' on the other; for example, *to destroy* denotes that there is an occurrence at the end of which something has changed—here, the integrity of the object in question.

Besides such verb-inherent features, there is another, slightly different, meaning of *aspect* more closely related to grammatical form, where for example the distinction between *progressive* and *non-progressive* in English is concerned. From this angle, there are indeed great differences between languages: German does not have a progressive form corresponding to the English; Slavic languages have a much richer grammatical aspectual system than either English or German. Thus, the term *aspect* covers a somewhat heterogeneous range of phenomena. Dorr and Gaasterland [1995], for instance, point out that aspect is traditionally taken to have two components, the non-inherent features (that define, for instance, the perspective such as simple, progressive and perfective) and the inherent features (that distinguish, for instance, between states and events). To help clear the ground, we suggest labeling the inherent features as the *Aktionsart*, a term from German linguistics, which is sometimes, but not regularly, used in Anglo-American research as well. While the *exact* difference between aspect and Aktionsart is an unresolved issue in linguistics, the latter clearly has to do with inherent features of the verb that characterize facets of the situation denoted by the verb. Aspect, in contrast, can then be confined to grammaticalized distinctions, i.e., those that are visible in the surface sentence and subject to choice; the fact that English verbs can occur in simple or progressive form (*Sally swims / Sally is swimming*) is largely independent of the verb's Aktionsart.

Presumably, the notion of Aktionsart originated in German linguistics because in this language information about the temporal structure of occurrences can be morphologically encoded in the verb with some regularity. The prefix *ent-*, for instance, can indicate the beginning of an occurrence, and *ver-* its successful culmination.[4] The latter is in English sometimes denoted by phrasal verbs with the preposition *up*. Thus, *entbrennen* means 'to start burning', and *verbrennen* means 'to burn up'.

In the reviews to follow here, however, we use the term 'aspect', because it is so common in Anglo-American linguistics. While the studies to be reviewed here all deal with English as object language, this is not problematic, because the categories being discussed apply equally well to German and many other languages. A source frequently cited as 'pioneering' for work on aspect is Vendler [1965][5], who posited that verbs fall into the four categories *state, process,*

[3]In this section, it will turn out somewhat difficult not to be confused by the various terminologies for situations and their subtypes. Before defining our own categories are defined in section 5.3, we will use *occurrences* as a generic, theory-neutral term referring to the things that can be "going on" in the world—exactly those that need to be classified here.

[4]These are by no means strict implications, though; *ver-*, in particular is a highly multifunctional prefix.

[5]Parsons [1990], however, reminds us that work on verb classification had indeed started several centuries

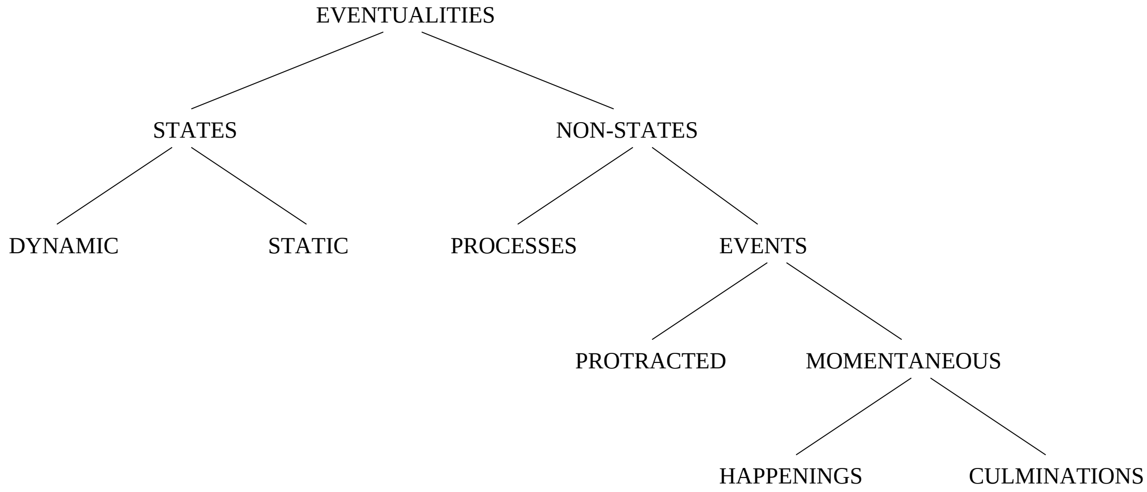

Figure 3.1: Taxonomy of eventualities from Bach [1986]

achievement, accomplishment. Later work has pointed out that, significantly, these categories cannot strictly be associated with verbs, as they can change when building phrases and sentences from them—this is the problem of *aspectual composition*, which will be explained below.

The Vendlerian proposals have been developed further, amongst others, by Bach [1986]. He sought a minimum classification of categories for dealing with syntactic and semantic phenomena of English, and to this end suggested a taxonomy of *eventualities*, which is reproduced in figure 3.1 (it is in turn based on work by Carlson [1981]).

The distinctions between the categories can, to a certain extent, be motivated with linguistic tests: the kinds of modifiers that can be added to an expression give an indication to that expression's category. *States* hold continuously over time. Bach distinguishes them further into *dynamic* (*sit, stand*) or *static* (*be drunk, be in New York, own a car*). Non-states are either *processes* (*walk, dream*) or *events*. The former can be diagnosed by adding an adverbial phrase expressing duration, e.g., *for an hour*, which can also apply to any *state*.

The subtree rooted in *events* represents the perfective occurrences, as opposed to the imperfective states and processes. Event descriptions accept the addition of a 'frame adverbial' like *in an hour*. Bach goes on to separate them into *protracted* (*paint a picture, walk to Boston*) and *momentaneous* ones. The latter are either *happenings* (*recognize, notice*) or *culminations* (*die, reach the top*).

Momentaneous activities can be diagnosed by adding a *point adverbial* like *at noon*. Also, Dowty [1979] suggests another test for distinguishing the two: *Protracted* events have two readings when modified by *almost*, as in *John almost painted a picture*—it is not clear whether he started painting and did not finish, or the activity never commenced at all. With *momentaneous* events, on the other hand, only one reading is possible, because the event is point-like and therefore cannot be executed half way through; it either occurs or it does not.

The correspondence to Vendler's categories seems to be the following: his states and processes have their counterparts in the Bach taxonomy. Achievements are Bach's momentaneous events, and accomplishments map to the protracted events.

Similar classifications along these lines, with minor variations, have been used widely in

before Christ, and that in our century, amongst others, Russell and Ryle have investigated some of the distinctions later elaborated by Vendler.

work on aspect, e.g. by Pustejovsky [1991], and White [1994]. Bennett *et al.* [1990] as well as Dorr and Gaasterland [1995] take three binary features to characterize different eventualities. Adapted to Bach's terminology, they are: ± dynamic (state vs. non-state), ± telic (processes vs. other, 'culminative', events), ± atomic (protracted event vs. momentaneous event).

Equipped with these distinctions, we can now illustrate the problem of aspectual composition. In a nutshell, the aspectual category of a verb can differ, depending on which of its semantic roles are present in the sentence. *Tom walked* denotes a process, as the *for*-diagnostic demonstrates: To further characterize the occurrence, we can add an 'unbounded path' to the clause without a change in aspect: *Tom walked along the river for an hour.* But as soon as the path is 'bounded', the end of the occurrence is already implicitly communicated: *Tom walked to the river* is an accomplishment, and now the duration of the event needs to be expressed as *in an hour.*

With some verbs, this shift toward including the completion of the event works by adding not an oblique phrase, but a direct object: *Sally read* is a process, but *Sally read the book* entails that she actually finished it. Compare: *Sally read the book in an hour.* If she had not finished it, one would say *Sally read <u>in</u> the book.* Further, the (in-) definiteness of the direct object can play a role: *Water drained from the tank* is a process, but *The water drained from the tank* can be read as an accomplishment, because the definite determiner converts the substance *water* to a fixed amount of that substance, which here acts as a discrete object. Consequently, in an event of movement, a bounded path is not enough to warrant an accomplishment; the object undergoing the motion also needs to be bounded. For extensive discussion of these problems, and a computational treatment, see [White 1994].

How do these linguistic considerations relate to our tasks of building a language generator and modelling the domain in which it operates? If the generator is to produce descriptions of occurrences, then categories of the kind just discussed are highly relevant. Consider the example of Sally's reading; if the generator is expected to verbalize that event and its duration, say two hours, then the realization of the temporal adverbial depends on the aspectual category of the event: *Sally read <u>for</u> two hours*, but *Sally read the book <u>in</u> two hours.* Or, slightly more elaborate, if the event is that of Sally pouring oil into the engine of her car until the level reaches a particular mark, it would be desirable to have at least two alternative verbalizations available; the one just used (*Sally did x until y*), or a shorthand comprising both the process and its result: *Sally filled the engine to the second mark with oil.*

A generator thus needs to know about aspectual categories and internal event structure if its capabilities are to move beyond dealing with simplistic input like `read(sally,book)`. Consequently, the domain model from which the generator receives its input needs to be rich enough to provide the information required.

3.7 Valency and case frames

In this short section, we can barely scratch the surface of the research field *valency*, which is notorious for widely heterogeneous terminology and approaches.[6] What seems to be uncontroversial, though, is that historically the notion of valency is due to Tesnière [1959], who

[6] Kunze [1987, p. 302], a senior valency researcher, started a paper with the following sentences: "I will not enter into the terminological discussion on deep cases, case relations etc., and [will instead] subsume all these variants under the label 'case relation'. This is justified by the obvious fact that there are more proposals and systems than authors. So one will not overcome this chaos by neat terminological distinctions." But while the situation is bad, there are nonetheless some good overviews of this situation, notably those of Somers [1987] and, in German, Storrer [1992].

developed a highly verb-centered approach to grammar and made the fundamental distinction between *actants* and *circumstances*; the former are taken as central participants in the process denoted by the verb, while circumstantials express temporal, local, or other circumstances that are less closely tied to the verb. While the existence of such a distinction is accepted across a wide variety of linguistic theories, the trouble starts when it needs to be made precise. German linguistics, for instance, always emphasized the syntactic side of the problem: how many actants are obligatory with a verb, and which surface case is assigned to them? This is not surprising, because German has a much richer case system than, say, English. Consequently, entire 'valency dictionaries' have been compiled for German (e.g., [Helbig and Schenkel 1973]). On the other hand, there have also been approaches to solving valency on the semantic level, with 'deep cases' (agent, patient, theme, instrument, etc.); these started with Gruber [1965] and Fillmore [1968]. The fact that semantic accounts give rise to heated debate is not really surprising, because selecting the 'right' set of deep cases, and afterwards assigning them 'correctly' to every verb in question, is a matter where the truth can hardly be proven. But even for purely syntactic accounts, it can be problematic to come to conclusions. In particular, grammaticality or acceptability judgements on the obligatoriness or optionality of constituents can at times be very difficult to justify.

We will make no further attempt at overviewing the scene. Instead, let us just illustrate some of the problems with examples. In chapter 7, we will return tho these issues and discuss how our generation system is to deal with valency and case.

To see that verbs can differ significantly in their valency requirements, consider first the verbs that typically require a direct object but in the right context—and only there—it can be elided. The sentence *I missed* is meaningful only in a situation where the identity of the target that was not reached is obvious to the hearer. Conversely, some verbs obligatorily need all their actants in the clause, no matter how specific the context is. *I put the book in the closet* can only occur in this complete form, never as ∗*I put the book* or ∗*I put in the closet*. This is also an example of the situation where an adverbial phrase that typically denotes a circumstance, *in the closet*, is in fact an obligatory actant; hence, actants can syntactically be more than only direct or indirect objects.

Then, there are verbs that exercise a strong semantic influence on their environment in the clause. Some can appear both transitively and intransitively: *Sally ate the pizza* is a perfect sentence, but *Sally ate* is also all right—we can *infer* that the missing direct object must belong to a particular category of things, here: edible things. Specifically verbs of *consuming* and *creating* can appear in both these configurations. With verbs of this kind, it is not possible to associate one fixed valency requirement. Different from these are verbs that also exercise strong semantic influence on their direct object, but the object cannot be deleted. The German verb *dauern*, for instance, expresses that something takes a certain amount of time, or up to a certain point of time—and the temporal object is strictly required: *Es dauert zwei Stunden* ('it takes two hours').

It is obvious that a language generator needs to know about the valency of verbs in order to generate correct sentences. Specifically, the lexical entries need to have the information on what different valency patterns a verb can appear in; this is one aspect of the *alternation* behavior of a verb.

3.8 Verb alternations

One reason for verbs being so popular in linguistics is that they can potentially occur in a number of different configurations in the sentence: *The door opened. Tom opened the door.*

The door was opened by Tom. The door opens easily. The challenge is to explain which of these *alternations* (or *diatheses*) are possible for particular verbs but not for others, and a central question is: How does the syntactic behavior relate to semantics? Or, is it possible to predict from the meaning of the verb which alternations it can undergo?

Not surprisingly, alternations have been investigated widely in linguistics. The most comprehensive result todate is the compilation by Levin [1993], who gives a catalogue of alternations, lists the English verbs that can undergo them, and proposes verb classes on the basis of the alternation behavior.

For our purposes, a central distinction to be made is between alternations that involve the denotation of the verb, i.e., alter its semantics, and those that do not. The latter merely affect the participant linking and thus the surface ordering of the constituents; they can shift emphasis from one participant to another, or help establishing the focus in discourse, but the central meaning (here in particular: the denotation) is left unchanged.[7] Examples are the *passive*, and also the *dative* alternation: *I read the book to her / I read her the book.*

Among those alternations that affect meaning, we distinguish two groups. The first is that of alternations that relate similar verb readings with a systematic change in meaning. A case in point is the *causative* alternation, which can be illustrated with the example *The tank filled / Tom filled the tank.* For another example, consider the *conative* alternation: *The butcher cut the meat* says that the butcher did something to the meat so that in the end it was cut in pieces. *To cut* can also be used with the proposition *at*; in this case, however, the result of the event is no longer implied: *The butcher cut at the meat* does not convey that the meat actually ended up in pieces. Another well-known case of an alternation that affects meaning is the *locative* alternation, which states that a verb like *to spray* can be used as *Sally sprayed paint onto the wall* and also as *Sally sprayed the wall with paint.* The difference is that in the second but not in the first sentence we have the aspect of 'complete' or 'holistic' covering of the wall with paint.

In the second group of 'semantic alternations', the meanings of the two forms are unconnected: A verb can occur in different configurations, which are, however, not related to each other. There is, for example, the *middle* alternation, which can be illustrated with *The butcher cuts the meat / The meat cuts easily*; the alternation says that the causative verb *to cut* can also be used to characterize an attribute of an object. Other verbs do not allow this: *Terry touched the cat / ∗Cats touch easily.*

Levin [1993] presents dozens of alternations, along with groups of verbs that do or do not undergo them, and then verb classes are proposed on the basis of their overall alternation behavior. Levin and other researchers working with her propose as their central claim that (non-)alternating behavior is indeed determined by the meaning of a verb: classes of verbs undergoing the same alternations are supposed to share one or more common semantic features, or *meaning components.* These are then taken to determine the range of possible syntactic argument structures for the verb. While this strong thesis is yet unproven and requires a lot of further investigation, Levin cites considerable evidence in its support. To give but two examples, the above-mentioned middle construction appears to be available only to verbs that involve causing a change of state, and the conative alternation applies only to verbs involving both motion and contact between two objects.

For lexical semantics, probably the most interesting group is the one including the *causative, conative*, and *locative* alternations mentioned above, for they pose the challenge of systematically relating not only the syntactic changes but also the shifts in meaning. Particularly, it is desirable

[7] From the perspective of truth-functional semantics one would say that these alternations do not change the truth conditions of the sentence.

to *derive* one verb reading from the other instead of positing a distinct lexical entry for each of them. Jackendoff [1990] is concerned with this problem for a number of alternations; specifically, in his LCS framework he seeks to explain the relationships between stative, inchoative, and causative readings of a verb. Applying this to the verb *to fill* yields: *Water filled the tank* (stative), *The tank filled with water* (inchoative/resultative), *Tom filled the tank with water* (causative). In Jackendoff's analysis, the forms are derived sequentially by embedding in the primitives INCH and CAUSE, respectively:

- stative: $\mathrm{BE}([_{Thing}\ ]_{\langle A\rangle}, [\mathrm{IN}_d\ [_{Thing}\ ]_A\ ])$
- inchoative: $\mathrm{INCH}\ [\mathrm{BE}([_{Thing}\ ]_{\langle A\rangle}, [\mathrm{IN}_d\ [_{Thing}\ ]_A\ ])]$
- causative: $\mathrm{CAUSE}([_{Thing}\ ]_A, \mathrm{INCH}\ [\mathrm{BE}([_{Thing}\ ]_{\langle A\rangle}, [\mathrm{IN}_d\ [_{Thing}\ ]_A\ ])]$

It is not necessary to introduce the LCS notation here in detail; only a few things need to be known. Individual entities are enclosed in square brackets, and the subscript at opening square brackets (here always THING) denotes the type of the entity. Here, the positions are all empty, though; the A-subscript at closing brackets indicates an argument position, which is optional if it appears in angle brackets. IN_d is a function that maps an entity to the PLACE within that entity, BE is a function mapping a THING and a PLACE to a STATE. In turn, the function INCH maps a STATE to an EVENT in which the STATE comes about. Finally, CAUSE maps a THING and an EVENT into another EVENT, which has the THING as causer.

While Jackendoff succeeds in deriving one reading from the other systematically, his solution with primitives like INCH is not satisfactory for our purposes. We have stressed the importance of making the internal structure of events explicitly visible, whereas an INCH primitive masks the relationship between an ACTIVITY and its resulting POST-STATE. In chapter 7, we will therefore propose a new mechanism for making derivations of this kind.

3.9 Salience

Consider the following examples of paraphrases:

I let the water run out of the tank.
I emptied the tank.

I bought the book from her.
She sold the book to me.

Jill is older than Jane.
Jane is younger than Jill.

Sentence pairs like these differ in terms of emphasizing a particular aspect of the situation or making one or another participant of the situation more prominent than the other(s). The general term for such phenomena is that they differ in attributing *salience* to the elements; this is characterized by Pattabhiraman and Cercone [1991] as "a measure of the degree to which the entity stands out from the rest, and 'gets the attention' of the speaker."

In order to determine how certain sentences render certain elements more salient than others, there has to be a sense of how these sentences differ from an 'unmarked' version that would not have that salience effect. In other words, salience can be discussed thoroughly only when comparing it to non-salience: For a sentence being able to make some element stand out from the rest, there must be a paraphrase of that sentence that is salience-neutral—in which nothing has special prominence over anything else. To find this neutrality, psycholinguists are trying to relate linguistic forms of expression to general patterns of cognition (e.g., [Osgood 1980]). They also point out that producing and recognizing salience in utterances involves complex interactions between world knowledge (what is *usual* and what is *unusual*) and knowledge about linguistic means for signalling salience.

Salience is a far-reaching issue (for an overview of the role of salience in NLG, see [Pattabhiraman and Cercone 1991], and some approaches were already summarized in section 2.3), and we will be dealing here only with two related aspects: the influence of salience parameters on distributing the semantic units across the lexemes, and on choosing the verb and its alternation. As background to this topic, we briefly review two contributions from linguistics.

Verbs distribute emphasis One group of salience phenomena has to do with verbs assigning emphasis to different aspects of a situation. To formalize this behavior, Kunze [1991] presents a system of decomposing verb meaning into semantic primitives that is designed to explain how the case role assignment for a verb can be calculated compositionally from its primitives. He introduces a base assignment of case roles to the primitive predicates, and then defines functions that project the assignment to lexemes and higher-level constituents.

Similar verbs share the same semantic base form, and the traditional selectional restrictions are now attached to these base forms, and no longer to the verbs. For instance, *to give* and *to receive* would be assigned the same base form for which selectional restrictions are specified; the verbs merely assign different roles to the same "deep participants" (the person giving away, the person receiving, the thing given), so specifying the selectional restrictions for each verb would in fact be redundant.

What, then, are "similar" verbs? Kunze lists three kinds of differences between verbs with the same base form:

- surface appearance of the participants (including obligatory and optional roles),
- specializations of the general base form (respecting subsumption), and
- fine-grained distinctions that cannot be handled by the formal representation system.

He concentrates on the first group and states that different mappings to surface sentences can be derived from the combination of the base form and three parameters, the most interesting of which is the distribution of emphasis. In the example above, *to give* places the person giving something away into the foreground, whereas *to receive* assigns the most prominent position to the recipient. Both verbs, however, are similar in their emphasizing the fact that at the end of the event the recipient is in possession of the object. Other verbs concentrate on the opposite fact: that the 'giver' loses possession of the object. In German, these are often morphologically marked by specific prefixes, as the *ab* in *abtreten* ('to give away') or the *ver* in *verkaufen* ('to sell'); English sometimes adds particles and creates phrasal verbs: *Tom gave the dress away to the Salvation Army* emphasizes that the dress is no longer Tom's; the new owner is of only secondary interest. Note that this distribution of emphasis is less apparent when the particle is removed: *Tom gave the dress to the Salvation Army*.

Different verbs can thus emphasize different *partial propositions* and thereby create a 'perspective' on the common semantic base form: they add different kinds of information to it; here, the emphasis distribution. The other two kinds of additional information will not be discussed here, nor the wealth of German examples Kunze presents to illustrate the similarities and differences between verbs, especially those of change of possession, with respect to emphasis distribution.

We will in chapter 7 take up Kunze's suggestion to split the representation of word meaning into different parts, which can be shared among similar verbs, and our representations will allow for representing the differences in emphasis assignment that he is concerned with.

Distribution of attention comparisons across languages Investigating the same phenomenon under a different label, Talmy [1988] is interested in linguistic means of distributing *attention* across the various units in a sentence. He lists the following:

- Topicalization. Fronting a constituent assigns a high a degree of attention to it: *To Hawaii he went last month.*

- Grammatical categories can be placed in a hierarchy of assigning attention: noun > verb > closed-class form. E.g., *I went by plane to Hawaii last month* places higher emphasis on aeronautic conveyance than *I flew to Hawaii last month.*

- Similarly, grammatical relations form an attention hierarchy: subject > direct object > indirect object > oblique. This corresponds closely to Kunze's analyses of verbs that assign these relations differently and thereby shift emphasis.

- Head vs. non-head constituency: It makes a difference whether a constituent is the head of a phrase, as the *bricks* in *The bricks in the pyramid came crashing down* or it is not, as in *The pyramid of bricks came crashing down.*

- Morphological autonomy. When information is expressed in a separate constituent, as the negation in *This is not relevant*, it receives more attention than in constructions that conflate it with another constituent: *This is irrelevant.*

- More generally, by means of conflation the information can be distributed across the constituents in different ways: *We went across the field* expresses the direction of movement separately, whereas *We crossed the field* incorporates it into the verb.

In other work, Talmy [1985] looked into the conflation behavior of verbs in more detail. He compared a number of different languages and discovered tendencies for incorporation, which we we will state in chapter 4. In effect, the conflations lead to foregrounding or backgrounding of minimal units of meaning, as opposed to standard notions of 'focusing' that have to do with complete linguistic constituents expressing participants of the clause. Talmy [1985, p. 122] says: "A semantic element is backgrounded by expression in the main verb root or in any closed-class element. Elsewhere it is foregrounded." When discussing salience in our own framework (section 7.4), we will sketch how Talmy's observations can be employed in a generator that makes lexical decisions, *inter alia*, on the grounds of a target distribution of salience across the elements of the sentence.

3.10 Conclusions: word meaning in NLG

In knowledge-based NLG, the specific role of lexemes is to "carry" the meaning from the underlying domain model over to utterances in natural language. To accomplish this step, the lexicon of a generation system needs to have two basic components: it has to explain what words mean with respect to the domain model, and it must explain how they can be combined into well-formed sentences. We have argued that lexical semantics is in charge of systematically relating these two tasks.

As pointed out in chapter 2, though, NLG has a history of largely neglecting the topic of lexical semantics, due to the fact that words and concepts were often conveniently put into a one-to-one correspondence. The cornerstone for incorporating lexical semantics into NLG is breaking up this tight correspondence and arranging for more flexible mappings. As soon as entities in the knowledge representation scheme do not correspond to words in a direct manner, or more precisely, if it is not postulated that these entities *represent words*, then the relationship between word meaning and entities in the KB needs to be specified in some more elaborate manner. Now, lexical semantics has to supply the interface between knowledge and words: it has to specify what words can be used to express what parts of what KB entities, and, possibly, under what circumstances. In short, there is room, and need, for applying more lexical semantics to NLG; this thesis will in the following chapters build on the linguistic research reviewed here, extend it, adapt it for generation purposes, and implement it in a working system.

Tasks The step of flexibly mapping from domain model to language opens up the door towards variety in language generation, towards being able to say roughly the same thing in several ways. Obviously, only an indirect association between KB entities and words gives rise to the possibility that several words, or combinations of words, can express the same content, so that a meaningful choice can be made. The features that distinguish similar words can be responsible for selecting the most suitable word from a set of near-synonyms, or a 'lexical field'. In turn, lexical semantics has to constrain this choice by determining exactly the right range of candidate lexical items that all share a common "kernel meaning" and differ in additional aspects.

As noted just above, lexical semantics also has the duty of accounting for the combinations of words in sentences, and correspondingly for the combination of meaning. Focusing on verbs, we will investigate the role of alternations that change the syntactic configuration as well as the meaning—if not the denotational meaning, then possibly another, more subtle one: that of assigning different degrees of salience to the elements of the situation. For example, *The picture was painted by Tom* makes the picture more salient than it is in the unmarked, active form: *Tom painted the picture*.

In summary, we have identified four central tasks for lexical semantics in natural language generation:

- Mediate between pre-linguistic and linguistic representations.
- Provide factors for well-motivated word choice.
- Provide constraints for combining words in sentences, and explain the composition of meaning.
- Explain how different verbalizations can distribute varying degrees of salience across the elements of the situation.

A final remark: Our discussions will convey the view that studying word meaning is largely the same as studying *verb* meaning. In fact, verbs have received by far the most attention in lexical semantics (and as well in syntax), since they are seen as the central constituents of clauses, where they arrange other parts of speech around themselves. We also focus most of our attention on verbs, as this perspective appears to be particularly helpful in generation. However, we want to point out that recent work in computational linguistics has developed ideas of spreading the "semantic load" of the lexicon more evenly across the various parts of speech. For example, both Bierwisch [1983] and Pustejovsky [1991b], as mentioned above, have proposed to explain the multiplicity of senses of verbs by viewing compositionality not as a simple application of a functor to an argument (as traditionally done), but rather as an *interaction* between the two, shifting some semantic weight to the arguments and thereby reducing polysemy in the lexicon. This kind of work on the so-called "generative lexicon" should ideally be complementary to our approach.

Chapter 4

Classifying lexical variation

As the goal of this thesis is to generate a wide range of lexical variants from the same underlying representation, we need to investigate more closely the notion of 'paraphrase', and then to delimit the range of variation to be produced in our system. A central idea of the project is to see multilingual generation as a mere extension of the monolingual paraphrase task and to devise the system architecture accordingly: Rephrasing an English sentence with different English words is not in principle different from rephrasing that same sentence with German words. Obviously, this stance is more difficult to defend when looking at less closely related languages, but for our purposes here we stick to English and German, occasionally looking at French for interesting examples. Therefore, we develop the following overview of lexical variation by looking at both paraphrases within a single language (section 4.1) and differences between languages, the so-called *divergences* (section 4.2). Then, section 4.3 points out the commonalities between the two and argues that from the viewpoint of NLG there should not be a fundamental difference between them.

4.1 Intra-lingual paraphrases

Lexicologists have studied the *synonymy* relation between words extensively, and it is the subject of thesauri and many dictionaries. But what exactly the conditions are for two lexemes to be synonyms is still unclear. Moving from the lexical level up to that of complete utterances, the corresponding notion is that of *paraphrase*. And again, the question of when two utterances are paraphrases of one another is a notoriously difficult one (see e.g., [Lenke 1994])—probably more difficult than the synonymy question, because a larger number of factors contribute to utterance meaning than to lexical meaning.

One possible view is that the *truth conditions* of the utterance are the same, but that does not help much with 'pragmatic' paraphrases of the kind to be given below. Another explanation posits that paraphrases logically imply one another, which is again insufficient, because one way of paraphrasing an utterance consists of moving to a higher or lower level of generality; thus, implication can only hold in one direction (examples will be given below). The very similar claim that paraphrases have the same set of implied propositions suffers from the same problem, of course. At any rate, without a clearly defined notion of *meaning*, it is hardly useful to discuss the idea of paraphrase at all, as it obviously has to do with some kind of meaning equivalence.

The following account, proposed by Naess [1975], does not attempt to rigidly explain the nature of paraphrases, but it may be helpful in just charting the territory. He distinguishes four cases:

(i) Utterances A and B mean the same to all hearers in all situations.

(ii) Utterances A and B mean the same to all hearers in some situations.

(iii) Utterances A and B mean the same to some hearers in all situations.

(iv) Utterances A and B mean the same to some hearers in some situations.

It has often been argued that 'true synonymy' does not exist, because of language's tendency to admit a new lexeme only if there is at least one tiny distinction it introduces into its overall system—be it a slight shift in connotation, collocation, or whatever. Correspondingly, class (i) of paraphrases may be thought to be empty—provided one assumes that even sentences using the same words and differing only minimally in word order do have a difference in meaning, for example a shift in focus. Class (ii) is the one that will be of most interest to us: paraphrases that work only in some situations, because they arise from a combination of linguistic meaning and non-linguistic knowledge. Class (iii) weakens class (i) to the effect that hearers can have individual linguistic preferences, as in dialects or sub-languages (see below). The vast majority of paraphrases probably belongs to class (iv). First, they are situation-dependent, and second, even if the "gist" of the message comes across all right, hearers tend to understand different additional shades of meaning, of which the speaker may or may not be aware.

In short, giving a general definition of *paraphrase* is a difficult and maybe not even very fruitful task. Taking the perspective of an actual language generation system, however, we are in a better position, because one criterion is quite clear: At some level of description, the paraphrases have the same representation, and only subsequent steps in mapping it to language bring about the differences, due to, for instance, slightly different pragmatic goals, while the propositional content remains unchanged. Before examining this more closely, though, we need to have the representations of the domain model (chapter 5) and of word meaning (chapter 7) in place; we will therefore resume this discussion at the end of chapter 7. For now, the goal is to categorize the various dimensions along which verbalizations can differ. The following groups are not meant to be mutually exclusive so that every paraphrase found in language would fit into exactly one; on the contrary, there is overlap because the categories look at paraphrases on different levels of description.

Pragmatics The same speech act can be conveyed more or less explicitly, or in different ways. Well-known examples are *I'm hungry / Can I have something to eat*, or *It's cold in here / Can you please close the window*. They are heavily situation-dependent and not always understood as intended and thus belong to class (iv). We will not be concerned with this kind of variation, though.

Connotation Words can have the same *denotation* but different *connotations*. This distinction has been discussed in section 3.4; for our generation purposes, we take the denotation to be the set of conditions necessary for using a lexeme, with respect to the underlying domain model. Connotations are of secondary nature and may be a reason for *preferring* one word over. Certain aspects of connotation, especially the *stylistic* dimensions, can be formalized as distinct features (see, for instance, [DiMarco et al. 1993]), such as 'class': A person can have a *job* as a cleaner, or an *appointment* as a professor; exchanging the combinations would lead to a rather ironic formulation.

Very often, *idiomatic* variation is also a matter of changing connotations, because many idioms tend to convey a colloquial or maybe vulgar tone, like the notorious *to kick the bucket*.

Dialect, Sub-language In different dialects, things can have different names; for instance, what North Americans typically call a *lightning rod* is a *lightning conductor* in Britain. In this rather systematic case, the different nouns denote different aspects of the object (shape versus function), but often the differences result from more arbitrary naming conventions. Similar to dialects, different *genres* or *sub-languages* can develop their own vocabulary; numerous examples can be found for instance in legal language, or in sports talk. Utterances from such genres are often not readily understood by "outsiders"; hence this kind of paraphrase belongs to class (iii) of the list given above.

Incorporation Words can distribute the pieces of information across the sentence in different ways, by means of *incorporation*. An entity or an aspect of meaning can either be expressed separately, or "swallowed" by another word, as in *to affect adversely / to impair*. Sometimes, a verb can be used in different alternations that do or do not express a chunk of meaning: *Keith hit the horse* says that Keith successfully performed a hitting action, whereas the so-called *conative* version *Keith hit at the horse* leaves open whether he actually succeeded.

Incorporation is also a means for *defining* terms. A word can often be replaced with a less specific term and appropriate modifiers: *to rise / to move upward*. We can view this as "incorporation via subsumption". A classical example is the incorporation of an INSTRUMENT into the verb, as in the example *to go by plane / to fly*.

Specificity A related dimension of choice is the specificity of the word one uses to refer to an object (*poodle, dog, animal*) or an event (*to darn the socks, to mend the socks*). The more general word has less information, in the sense that it can denote a wider class of entities; yet there can be good reasons for using it, for example when certain connotations are to be expressed. If one does not like the neighbor's dog, the derogatory attitude can be conveyed by referring to it as *that animal*.

Selection of an aspect Verbs can denote certain aspects of an event and leave other aspects to be inferred by the hearer. Linguistically, such verbs differ in terms of their *Aktionsart*, which was introduced in section 3.6. Example: *I let the water run out of the tank* describes the process that went on, whereas *I emptied the tank* characterizes its result. In a context where the relevant circumstances are known to the hearer (here: there was water in the tank, and it could be let out), both sentences are candidate verbalizations. Whether such decisions affect meaning, should again not be debated without a particular notion of 'meaning' to base the argument on.

Syntax As mentioned in chapter 1, there are purely syntactic decisions to be made in generation, such as constituent ordering, thematizing, clefting, or passivizing. Also, morphosyntactic processes like nominalization produce syntactic paraphrases: *It angered him that the Romans destroyed Carthage / The destruction of Carthage by the Romans angered him*. Most of these will not be of central interest in this thesis, but, obviously, many lexical decisions also have syntactic consequences; for instance, nearly synonymous verbs can map their argument roles to different surface positions.

Role assignment When certain alternations are applied to a verb, the assignment of semantic roles to grammatical functions (subject, direct or indirect object, adjunct) changes. Consider, for example, the dative alternation: *I gave the books to Mary / I gave Mary the books*. The same can happen when the verb is replaced with a near-synonym or a more general verb that exchanges some roles; with *to donate*, for instance, only one configuration is possible: *I donated the books to the museum / ∗I donated the museum the books.*

Incorporation variants often have different constraints on role assignment. Consider the correspondence of 'driving something to somebody' and 'bringing something to somebody by car': *Tom drove the books to Sally / ∗Tom drove Sally the books — Tom brought the books to Sally by car / Tom brought Sally the books by car.* Such alternations or verb exchanges can be used to assign different degrees of prominence to the various elements of the situation.

Perspective When a different role assignment results, on the other hand, from choosing non-synonymous verbs, a different perspective is taken on the same event: *I bought the book from her / She sold the book to me.* This relates to lexical *antonymy* as a source of paraphrase: *Jim is older than Jane / Jane is younger than Jim.*

4.2 Inter-lingual divergences

A translation from one language into another can sometimes be a literal, that is word-by-word, equivalent of the original. Usually, however, this is not possible, in which case we face a *divergence* between the two languages: they use different means to communicate the same meaning. Or, it might be just impossible to communicate exactly the same meaning, and one has to be content with an approximation. In machine translation, such cases where the meaning has to be slightly changed in the target language are often called translation *mismatches*. It seems, though, that an exact division between the two is hard to define and that there is, rather, a continuum between them. Therefore, we suggest using the notion of paraphrase as encompassing both; then we can view the translation as a paraphrase of the original. Or, from the generation perspective: verbalizations in both languages are all paraphrases of the same underlying content. Again, these notions need to be defined more precisely once we have explained the domain model and the representations of word meaning. For now, focusing on lexical matters, we can distinguish the following cases of inter-lingual phenomena.

Morphology As is well known, German has a strong tendency to form compound nouns in order to refer to specific objects. In French, this is hardly possible, and one has to construct phrases with *de* ('of') instead. English does not facilitate morphological compounding, but several nouns can appear in a row. An example from the car manual quoted earlier: The *engine oil filler cap* is in German an *Öleinfülldeckel* and in French a *bouchon de remplissage.*

Lexical grain-size Sometimes, languages exhibit different grain-sizes in their vocabulary; that is, one language makes a distinction where another does not. French *savoir* and *connaître* correspond to German *wissen* and *kennen*, but in English both are covered by *to know*. The phenomenon can be generalized to observing different lexical taxonomies. A notorious example is *to put*, where German typically uses one of the verbs *setzen, stellen, legen* ('to sit', 'to stand', 'to lay'), which add information about the shape of the objects in question, and their relative positioning. Similarly, Kameyama et al. [1991] show that Japanese and English make different distinctions when describing kinds of pictures, and events of giving. Wunderlich [1991] notes that the German verbs *nehmen, kaufen, bekommen* ('to take', 'to buy', 'to get') all map to the same lexeme *almak* in Turkish, and the differences (agency and the return of money) are to be supplied by the context.

Conventions in lexical specificity The absence of some specific word from a language is one thing; a different matter is a tendency to use specific words less often. In the TECHDOC corpus studies [Rösner and Stede 1991], we noticed that English prefers to use abstract and less specific verbs where German has a concrete and quite specific verb. In bilingual manuals, for example, we found *to remove* corresponding to numerous German verbs that specifically characterize the physical activity and the topological properties of the objects involved.[1] These verbs are not absent in English, but it seems to be more common to describe the abstract effect of the action. Hawkins [1986, p. 28] sees a generalization here: "It is often observed ... that German regularly forces a semantic distinction within a lexical field where English uses an undifferentiated and broader term."

Different incorporation The seminal work by Talmy [1985] demonstrated that different languages (or language families) exhibit different tendencies for incorporating information, what he called "lexicalization patterns". English motion verbs, for example, tend to incorporate the MANNER of motion into the verb, whereas Romance languages prefer to incorporate the PATH: *The bottle floated into the cave / La botella entró a la cueva flotando* ('The bottle entered the cave floating'), or *He swam across the river / Il a traversé la rivière à la nage* ('He crossed the river swimming').

Different role assignment Verbs that correspond to one another can assign surface roles differently. This has been called "thematic divergence" [Dorr 1993]: *I like Mary* / Spanish *Me gusta Maria* ('Mary pleases me'). Here, the person who likes Mary is once expressed as subject, and once as direct object with dative case. And, obviously, corresponding verbs can undergo different alternations. *I gave the students some books* is as well-formed as *I gave some books to the students*, but the first cannot be translated literally into Russian [Nirenburg and Levin 1992], and the literal translation of the second into German sounds at least odd.

Different construction More dramatically than switching, for instance, subject and object, corresponding verbs can give rise to entirely different sentence structures. Recall the example from a car manual, given in chapter 1: *Twist the cap until it stops* (two clauses linked by a connective) / *Drehen Sie den Deckel bis zum Anschlag* (lit. 'turn the cap up to the stop', one clause and prepositional phrase).

Head switching From a syntactic viewpoint, the "worst case" of different constructions are those of *head switching*, as they are labelled in the machine translation literature. Some part of meaning is expressed by the verb in one language, but as either an argument or an adverb in the other language; we have described Dorr's [1993] treatment of *demotional* and *promotional* divergences in machine translation. An example for the former is *Peter likes to play chess / Peter spielt gern Schach* ('Peter plays likingly chess').

Different aspect In the previous section, we discussed the possibility of choosing different aspects of an event when verbalizing it. Between languages, this is occasionally not a matter of choice; in our car manual, for instance, we find the instruction *disconnect the spark plug wire* and the German version *ziehen Sie das Zündkabel ab* ('pull off the spark plug wire'). The English version thus characterizes the technical effect of the event, whereas the German one describes the physical activity leading to it. There are two reasons why the literal translation

[1] Schmitt [1986] made the same point in his study of various technical manuals.

of the English sentence is not used in German: *trennen* ('to disconnect') requires the explicit verbalization of the entity that something is disconnected *from*, which goes against the rule of giving short instructions. And, more seriously, the use of *trennen* or *abtrennen* would actually be misleading, suggesting not an ordinary unplugging, but a forceful cutting off or something similar.

Another example, from Wunderlich [1991]: A roll-up shutter, as sometimes used on shop windows and doors, can be *opened* in English, but hardly ??*drawn up* or ??*pulled up*. In German, however, the word corresponding to *to open* (*öffnen*) is quite uncommon in this context; it is more natural to use *hochziehen* ('to draw up') or *aufziehen*. Morphologically, the latter can be either an amalgamation of drawing and opening, or a shorthand of *heraufziehen*, which is a synonym of *hochziehen*. Similarly, in French one uses *tirer en haut* ('to draw upwards'). Thus, again, English prefers to verbalize the result of the action, while German and French characterize the action itself.

But verbs are not the only words that invite cross-linguistic comparison. In much the same way as they can incorporate different aspects of a situation, we occasionally do find compound nouns emphasizing different aspects of an object. As mentioned above, what in British English is a *lightning conductor* the Americans prefer to call a *lightning rod*—one word focuses on function, the other on shape. The German *Blitzableiter* is close to the British version, but the morpheme *ab* adds the aspect that the lightning is being led *away*. In Polish, *piorunochron* means *lightning protector*, thus emphasizing not its physical function but rather its utility.

4.3 Divergences as paraphrases

When the goal is to do bilingual generation in very much the same way as monolingual generation, it needs to be demonstrated that handling the inter-lingual divergences requires no special machinery beyond that needed for monolingual paraphrasing. Or, equivalently, it needs to be shown that the variation possible between two languages is in principle also possible within a single language. For English and German, we thus briefly examine the divergences again.

- Morphology typically has to do with lexicalizing single units of meaning; hence the different conventions can be encoded in the respective lexicons.

- Differences in lexical grain-size between languages are resolved by resorting to a more general word and possibly adding an appropriate modifier. The very same technique is to be used in monolingual paraphrasing, as we have seen in the paragraph on incorporation.

- Conventions in lexical specificity are not a specifically inter-lingual problem, either. When generating one language, there might be a preference for a less specific item than that to be used in another language. But the variety of more or less specific words has to be available anyway, and the knowledge about specificity conventions needs to be made a choice factor in lexical selection.

- Incorporation variants between languages do not rely on any mechanism other than that for incorporation variants within a language. It is just the case that the options for distributing chunks of meaning across the words are different.

- Different role assignments are possible in a single language, too, when a verb is replaced by a near-synonym. We have seen examples above.

- Different syntactic constructions can of course also be employed in a single language. The range of options can differ, though.

- Head switching can also occur within a language: English *to like* can, as we have seen, be rendered by the German adverb *gern*; but there is also the verb *mögen*, which is often nearly synonymous with *gern tun*: Both *Ich fahre gern Rad* and *Ich mag radfahren* are possible translations of *I like cycling*.

- Different aspects of events can also be chosen in one as well as in two languages, as we have seen.

To summarize, when we make a range of verbalization options available for a single language, then getting that range for the other language is not an additional problem. But it is not the case that every option in one language necessarily has its exact counterpart in the other: these are the instances of divergences.

Chapter 5

Modelling the domain

In NLG, there has so far been little emphasis on domain modelling; but in order to arrive at the lexical variation in verbalizations that we have set out to achieve, the decisions for domain modelling need to be made carefully. Thus, chapter 5 develops the domain model for the generator: the taxonomy of concepts and relations that, in later chapters, the generation system will be based on.

5.1 Building domain models for NLG

A language generator requires a model of the domain that it is expected to "talk about": a specification of how objects are related to one another, how actions change states of affairs, etc. In our case, the sample domain grew out of the TECHDOC application domain. We will be dealing with a small world in which various substances are moved in and out of containers in different ways; the containers vary in their openings and their means for measuring the fluid level inside. In an automobile, this scenario comes in several variants: engine oil, coolant, transmission fluid, brake fluid, power steering fluid, windshield wiper fluid. See figure 5.1 for an excerpt from a bilingual car manual, which illustrates part of the world we need to model. While this is a fairly narrow *conceptual* domain, we will see in later chapters that the range of possible *verbalizations* of events occurring in this domain is not narrow at all; in fact, it holds a range of interesting cross-linguistic lexico-semantic phenomena that pose some challenges to natural language generation. And this is the second reason for choosing this particular sub-domain for our work.

For any modelling task, the first thing needed is a suitable representation language. Section 5.2 introduces LOOM, a language of the widely popular KL-ONE family, nowadays often called 'description logic'. LOOM offers representation and reasoning facilities that have proven very useful in TECHDOC, and similar languages are used in many similar systems. In short, it is almost a 'standard', and therefore the introduction will be fairly brief.

The other prerequisite for building a model is less technical and more intellectual: determining the basic ontological distinctions for the model. These decisions are typically difficult to motivate and almost impossible to prove 'correct': we just do not have any agreed-upon system of basic categories that would explain how the world is put together. There are tendencies, though, and some of them can be derived from language—given that we accept the view that language, in whatever specific sense, mirrors the structure of the world. The ontological decisions outlined in section 5.3 below are largely based on contemporary work on aspectual structure, which was introduced in section 3.6. Recall that aspectual structure is concerned with distinctions like that between *states* and *events*, and this phenomenon builds a bridge to

Changing oil

(i) Warm up the engine.

(ii) Remove the engine oil filler cap and drain bolt, and drain the oil.

CAUTION: A warmed-up engine and the oil in it are hot; be careful not to burn yourself.

(iii) Reinstall the drain bolt with a new washer and refill the engine with the recommended oil to the upper mark on the dipstick.

ENGINE OIL CHANGE CAPACITY: 3.5 liters

(iv) Start the engine and make sure oil is not leaking from the drain bolt.

Ölwechsel

(i) Motor warmlaufen lassen.

(ii) Motoröleinfülldeckel und Ablaßschraube entfernen und Öl ablassen.

VORSICHT: Bei heißem Motor ist auch das Öl heiß; geben Sie acht, damit Sie sich nicht verbrennen.

(iii) Die Ablaßschraube mit einer neuen Dichtungsscheibe anbringen, den Motor mit dem empfohlenen Öl auffüllen, bis der Stand die obere Marke am Tauchmeßstab erreicht.

MOTORÖLWECHSELMENGE: 3.5 Liter

(iv) Den Motor anlassen und sichergehen, daß kein Öl an der Ablaßschraube ausläuft.

Figure 5.1: Sample text from a Honda car manual

the second factor in making ontological decisions: the kind of *reasoning* that is to be done with the knowledge base. The scenario that artificial intelligence has employed traditionally in *planning* is one where particular *operators* are used to transform *states* into others, and where the sequence of steps towards achieving a desired goal state can be computed by applying knowledge about the effects of operators. Our domain model (and the TECHDOC knowledge base in general) is designed to support an actual *simulation* of the events for which instructions (such as those in figure 5.1) are to be generated. Therefore, the states of objects and the operations that change them are central categories that shape our ontology.

With these two foundation stones in place we can then turn to the domain model itself. Section 5.4 introduces our 'microworld': it provides a taxonomy of the objects and relations that are relevant in the sample domain. We will illustrate the LOOM representations with examples and explain some of LOOM's reasoning facilities that are useful for our purposes. The idea is, obviously, that the sample domain is *just one* domain that can be modelled using the ontological system given in section 5.3, which is intended to be general enough to cover a wide range of similar areas.

It should be emphasized that the approach to domain modelling presented in this chapter is seen as merely one among many possibilities—certainly not as "the one and only" correct approach. Choosing the particular way of constructing a model is always to be decided on the basis of what is intended to be *done* with the model. Or in other words: It is the nature of any *representation* that it abstracts from certain aspects of the entity represented; which ones these are depends entirely on the purpose of the system that makes use of the representations. Only the purpose renders things either relevant or irrelevant for the model.

5.2 Background: knowledge representation in LOOM

In the early 1980s, a new strand of research in the field of knowledge representation was initiated by the development of a language called KL-ONE [Brachman and Schmolze 1985]. Basically, the rationale was to provide a formalization for the ideas of 'frame' languages that were popular at the time. Frames consist of various 'slots' and their 'fillers', which relate frames to one another, thereby incorporating the ideas of 'semantic networks'.

Within a KL-ONE language, one can define a taxonomy of concepts and of relations holding between them; these *roles* can be restricted in various ways, e.g., with respect to the concepts that fillers must belong to, or to the number of fillers that a role can have. Importantly, any definition written in KL-ONE can be assigned an *interpretation* in terms of a denotational semantics: the idea of frames, slots, and fillers is put on solid ground. The KL-ONE proposals gave rise to a wealth of theoretical investigations regarding the tradeoff between language expressivity and computational tractability, and also to a variety of specific implementations of such languages, which opted for different solutions with respect to that question. The one we are using here is LOOM [MacGregor and Bates 1987], a system that is under continuing development at the Information Sciences Institute of the University of Southern California.

Following a general characteristic of the KL-ONE family, LOOM offers two languages complementing each other: a *terminological* language for defining concepts and their relations (TBOX), and an *assertional* language for representing specific *instances* of concepts (ABOX). Roughly speaking, the TBOX defines the categories of things or events (e.g., DOG or SLEEPING), and their relations. A concept `C1` that is subsumed by a more general concept `C0` inherits all the information from `C0`. Importantly, a concept can be subsumed by more than one other concept, in which case it inherits the properties of all its subsumers (multiple inheritance). The

ABOX represents actual entities (e.g., FIDO or BRUCE'S-SLEEPING-LAST-NIGHT).[1] In database terms, the TBOX defines the structure of the database (the *schemata*), and the instances are the actual data. Consequently, the ABOX language contains operations not only to assert data, but also to construct queries for retrieving instances.

As its central service, LOOM offers a *classifier* that computes subsumption relationships between both TBOX and ABOX expressions. That is, one can describe a concept solely in terms of its properties (in what relations it stands to other concepts), and the classifier will determine its position in the concept taxonomy. Similarly, a new instance can be described, and the classifier will automatically compute its type, i.e., the most specific concept whose definition subsumes that of the instance. And, vice versa, a retrieval operation returns those instances that the classifier has determined to be subsumed by the query expression. With the help of the classifier, LOOM also maintains the consistency of the represented information and alerts the user when a new concept definition or instance assertion is in conflict with the current state of the KB.

The most important well-formedness condition enforced by the classifier (for our purposes, anyway) is the type restriction on role fillers, mentioned above. This restriction is typically another concept name, and fillers of the role have to be instances of that concept. But fillers can also be numbers or uninterpreted symbols, and the range of the relation is then defined as a numeric interval, or a set of symbols, respectively. We will see examples later. The final feature of LOOM to be mentioned here is the *context* mechanism, which allows one to divide the data into separate 'bags' (technically: name spaces). This will prove useful, because we will define instances that act as interfaces between a conceptual structure and the lexemes, and these 'lexical instances' can be kept in different contexts (one for the English lexicon, one for German) so that the task of finding candidate lexemes for a specific target language can be implemented straightforwardly (to be explained in chapter 9).

In summary, the typical way of using a system like LOOM is as follows: One first constructs the domain model by defining the concept and relation taxonomy. Next, instances of concepts are created, representing specific entities. These instances can then be retrieved using the query language, which provides facilities similar to those of a database.

5.3 Ontological categories in our system

We now begin to design our domain model and proceed in a 'top-down' fashion from the very general categories to the specific domain concepts. The top level of our ontology reflects basic distinctions that are commonly made in domain modelling: it distinguishes *objects*, *qualities*, and *situations*. The last of these are by far the most interesting for our purposes, and thus will be analyzed in detail. Figure 5.2 depicts the top of the overall ontology, which will be expanded throughout the chapter, sometimes with their actual LOOM definitions to illustrate the use of the language.

Objects are mostly things that are physically tangible (e.g., dipstick, cap) or are in a broader sense a part of a tangible object (e.g., a mark on a dipstick). There are also abstract objects, things that can be talked about but that do not exist physically; e.g., the liquid level in a tank.

[1] Both concepts and instances are pre-linguistic entities and thus appear in SMALLCAPS. To distinguish them, we will adopt the convention of forming instance names by adding numbers to the name of the concept they are instantiating. For example, DOG-1 and DOG-2 would stand for instances of DOG.

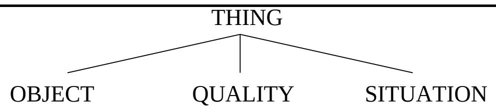

Figure 5.2: The top level of our ontology

Qualities are those attributes of objects or situations that are taken to remain constant, for example, shape and color of an object, or a manner in which an action is performed (say, quickly or slowly). "Constant" is, of course, meant with respect to the domain model—if in the particular domain it is not relevant that an attribute of an object could possibly change, then we treat it as a quality. On the other hand, a *state* (see below) of an object is prone to change, especially as a result of actions. The distinction between states and qualities is primarily motivated by the choice of events that are needed in the domain model; it is not meant to be a "natural" one.

Situations are occurrences that can relate objects and qualities. Deciding on their ontology is a complex matter and is the subject of the rest of this section.

As pointed out above in section 5.1, a central guideline for our modelling of occurrences is to account for the notion of *states* and their changes, so that some basic reasoning capabilities for the KB are supported. States change because things happen in the world, and these things we need to model. Rather than representing such changes with simple predicates, we are interested in breaking up the internal structure of *events* and in exploring different ways of generating linguistic descriptions of such events. As stated in chapter 1, this is a focal point of the thesis, because previous NLG research has largely worked with input structures too simple to account for interesting variation in verbalizing events.

When classifying the kinds of occurrences in the world, it is natural language that provides clues to at least the basic distinctions; there is no point in demanding representations to be independent of language. The three categories suggested above also correspond to linguistic ones: objects are typically referred to with nouns, qualities with adjectives and adverbs, and situations with verbs. In designing ontologies, we cannot liberate ourselves from the linguistic categories that we use every day; however, we can try to abstract from certain peculiarities of any particular natural language and broaden the perspective by working with different ones. In this thesis, we follow this guideline to some extent—English and German are quite closely related and do not exhibit any "deep" categorial differences.

In short, for domain modelling it is by no means illegitimate to resort to linguistic research when determining the basic categories. And the "branch" of linguistics most interesting to this aim is that of studying *aspect*, which we have discussed in section 3.6. Specifically, we will build on the ontology given by Bach [1986], shown in figure 3.1. The category distinctions we make for our purposes here are in a few respects less fine-grained than those of Bach, but at the same time go beyond his account at one central point. We call the general class of occurrences SITUATIONS and distinguish three different kinds: STATES, ACTIVITIES, and EVENTS (see figure 5.3). Activities were called *processes* by Bach, but we will need this term on a different level of description, to be introduced in the next chapter.

States are seen much in the same way as Bach sees them: Something is attributed to an object for some period of time (possibly indefinitely), and the object involved is not perceived as

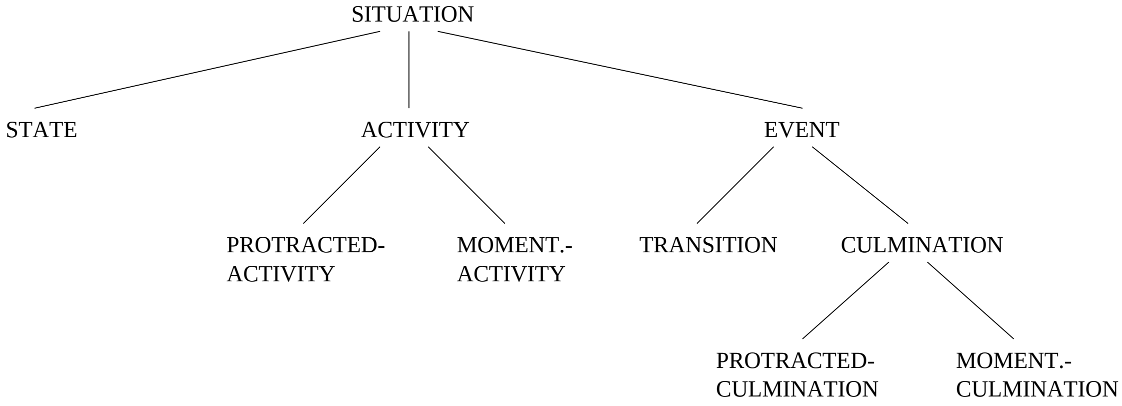

Figure 5.3: Our classification of situation types

"doing" anything. *The bottle is empty* is true for the bottle without it doing anything about it, and the same holds for affairs like *Jeremy is ill.* We do not make further distinctions among states here.

Activities are quite similar to states, but there is always something "going on". This can be related to volitional action by some agent, but it need not: The water in the lake being calm is a state, but the water in the river flowing towards the sea is an activity, although it has nothing to do with volition.

Activities can be distinguished with regard to their *(un-)boundedness* (cf. Jackendoff [1993]). Some are by their nature limited in duration (*Jill knocked on the door*), while others appear as essentially unbounded. In English, at least three different linguistic cases can be distinguished. The verb can inherently signal unboundedness (*Sally slept*); other verbs achieve the same effect when used in the progressive form (*His heart was beating*), or when combined with a verb that projects an otherwise bounded occurrence into an unbounded one (*He kept hitting the wall*).

Normally, the boundedness feature is taken to distinguish activities from events. Thus our distinction between unbounded and bounded activities may be controversial; Bach does not make it, and others, e.g. White [1994], would treat a momentaneous activity as an event. Our notion of EVENT, however, will always involve some change of state, which is quite unnatural to assume in the case of, say, someone knocking on a door. To label the two kinds of activities, we refrain from using the loaded term *bounded* and instead borrow from Bach's terminology here: we call the bounded activities MOMENTANEOUS and the others PROTRACTED. A linguistic test to distinguish the two is the 'point adverbial': *Jill knocked at noon.* Although *Jill slept at noon* is grammatically well-formed, too, the difference is that this sentence does not entail that Jill did not sleep immediately before and immediately after noon. Also, it is interesting to note that the standard diagnosis for activities, adding an adverbial like *for an hour* always produces an *iterative* reading when applied to a momentaneous activity: *to knock for an hour* does not mean that a single knock lasted that long, but that the activity was performed repetitively.

Events are occurrences that have a structure to them; in particular, their result, or their coming to an end is included in them: *to destroy a building, to write a book.* As their central feature we take them to always involve some change of state: the building loses its integrity, the book comes into existence, or gets finished.

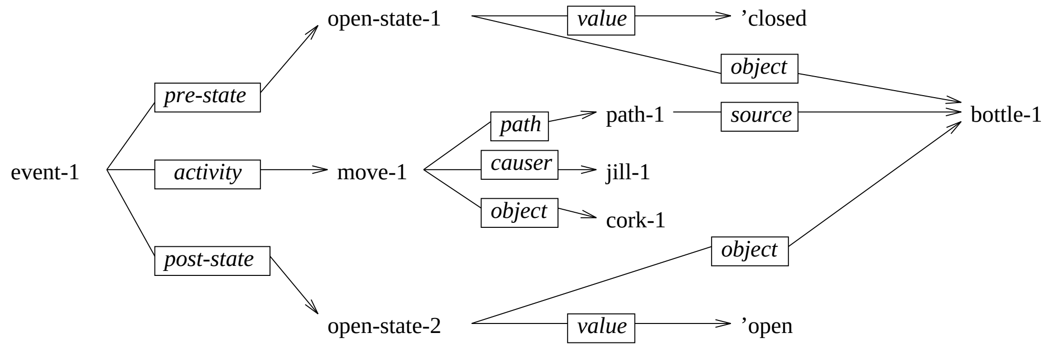

Figure 5.4: Event representation for Jill opening a wine bottle

Bach distinguished two kinds of events, momentaneous and protracted ones, but did not look at their internal structure. Others suggested, however, that this needs to be done; Moens and Steedman [1988] described an event as consisting of a *preparatory process*, a *culmination*, and a *consequent state*. Parsons [1990] posited a *development* and a *culmination* portion of an event. Pustejovsky [1991] treated Vendlerian accomplishments and achievements as transitions from a state Q(y) to NOT-Q(y), and suggested that accomplishments in addition have an intrinsic agent performing an activity that brings about the state change. The activity predicate is added to the "prior" state.

We follow the suggestion of combining activity and transition in an event representation, but will modify it in some important ways. Basically, we see any event as involving a state change; an activity responsible for the change can *optionally* be present. A plain transition is necessarily momentaneous (*The room lit up*), whereas a transition-with-activity inherits its protracted/momentaneous feature from the embedded activity. In other words, we see an event as consisting of three parts: a state that holds before the event commences, a state that holds after the event has completed, and optionally an activity bringing about the transition. The relationship between the activity and the transition is not one of *causation*; rather, the state change is inevitably entailed by the transition. For example, removing the cork from the bottle does not cause the bottle to be open, but is the *very act* of opening it. As a linguistic diagnosis, sentences describing the relationship as one of purpose sound slightly odd: ?*Jill pulled the cork out of the bottle in order to open it.* Causation, in our framework, can only hold between *different* events (and is thus not discussed further here). For it to hold, there is always a set of conditions that need to be true; when flipping the switch in order to turn the lights on, one of the conditions is that there be no power-out. On the other hand, when the activity of an event brings about a transition, that set is always empty. We call the events with an embedded activity *culminations* (but not in exactly the same sense meant by Bach), and they fall into protracted and momentaneous ones, just like the activities.

As an example, figure 5.4 shows our representation of an event where Jill opens a wine bottle. She performs the activity of moving the cork out of the bottle (for our model of PATHS, see the next subsection), which causes the OPEN-STATE of the bottle to change its value. In the figure, the quotation mark in front of 'open and 'closed distinguishes uninterpreted symbols from instance names.

Generalizing from Pustejovsky's [1991] proposal, we take state transitions to be more than merely oppositions of Q(y) and NOT-Q(y); they can also amount to a gradual change on some scale, or involve other values. In a world of containers and liquids, for instance, it makes sense to model the FILL-STATE of containers with a value on a numeric scale, or a few discrete points, or whatever is convenient. So, *the tank filled* could be represented as a transition from NOT-FULL(tank) to FULL(tank), but *the tank filled to the second mark* needs a different value assignment.

There is another point of contrast to Pustejovsky [1991]. He treats *agentivity* as a central feature that is taken to distinguish between Vendlerian accomplishments and achievements. In our view, the presence of a volitional agent is not responsible for any of the category distinctions; rather, that feature cuts across the aspectual categories. Except for STATES, which are always agentless, any situation might or might not have an agent, which we here call CAUSER—again, to distinguish it from the linguistic level of description, which will be explained in the next chapter.

Figure 5.3 above shows the complete ontology of situations; it directly expands on the SITUATION node at the top of our ontology shown in figure 5.2. Let us again illustrate the situation types with linguistic examples:

- State: *Tom was ill.*
- Protracted activity: *Tom studied the menu.*
- Momentaneous activity: *Tom blinked.*
- Transition: *The lights turned green.*
- Protracted culmination: *Tom filled the gas tank.*
- Momentaneous culmination: *Tom flipped the switch.*

To illustrate the usage of LOOM, figure 5.5 shows the LOOM concept definitions for the top of the SITUATION ontology. To enhance readability, some of the associated relations as well as a few details have been omitted. For instance, the commands defining EVENT, STATE, and ACTIVITY as an 'exhaustive partition' of SITUATION are not shown; they ensure that every instance of type SITUATION is an instance of exactly one of the three (the three are disjoint). Note that the restrictions on the number of role fillers (`exactly, at-most`) are used to encode the difference between the optional and obligatory parts of an EVENT, and thereby the difference between TRANSITION and CULMINATION.

In summary, what we are proposing is a synthesis of a "traditional" ontology like that of Bach [1986] with the representation of the *internal structure* of events, as called for by Pustejovsky [1991], but with several modifications. To facilitate comparisons, our ontological system can be related to the Vendlerian categories as follows. Vendler's *state* and *activity* have their counterparts in our system, but we make an additional distinction among the activities. Vendler's *accomplishment* corresponds to our protracted culmination; his *achievement* is split into three groups here: the transitions, and both the momentaneous activities and culminations. Similarly, Bach's categories as well as the three binary features used by Bennett *et al.* [1991] (recall section 3.6) can be related to our scheme, as shown in table 5.1.

5.4 A domain model for containers and liquids

We can now begin constructing our domain model along the basic ontological distinctions just explained. As mentioned earlier, the sample domain for this thesis originally arose from studying automobile manuals and exploring the possibilities of generating maintenance instructions

```
(defconcept situation :is-primitive thing)

   (defconcept state :is-primitive situation)

   (defconcept activity :is-primitive situation)

      (defconcept protracted-activity :is-primitive activity)
      (defconcept momentaneous-activity :is-primitive activity)

   (defconcept event :is-primitive (:and situation
                                         (:exactly 1 has-ev-pre-state)
                                         (:exactly 1 has-ev-post-state)
                                         (:at-most 1 has-ev-activity)))

      (defconcept transition :is (:and event (:at-most 0 has-ev-activity)))

      (defconcept culmination :is (:and event (:exactly 1 has-ev-activity)))

         (defconcept protracted-culmination
           :is (:and culmination
                     (:the has-ev-activity protracted-activity)))
         (defconcept momentaneous-culmination
           :is (:and culmination
                     (:the has-ev-activity momentaneous-activity)))
```

Figure 5.5: LOOM definitions for basic ontological categories

Bennett *et al.* 1991	Bach 1986	Our ontology
+ dynamic	non-states	activity ∪ event
− dynamic	states	state
+ atomic	momentaneous	momentaneous-activity ∪ transition ∪ momentaneous-culmination
− atomic	states ∪ processes ∪ protracted	protracted-activity ∪ protracted-culmination
+ telic	events	event
− telic	processes ∪ states	activity ∪ state

Table 5.1: Correspondences between ontological categories

automatically. Around the engine of a car, a variety of liquids and liquid containers with different properties play an important role. Actions of filling and emptying such containers are part of the domain, as well as accompanying actions like opening and closing them in various ways. When building the domain model, an important design goal is to keep general knowledge that can be transferred across domains separate from those parts that pertain only to the domain in question. That is, in the taxonomy of concepts, the more general ones are supposed to be transferable to similar domains, whereas only the specific concepts at and around the leaf nodes of the taxonomy should be confined to the automobile world. We will now briefly describe the various branches of the model, as far as they are relevant for the language generation examples discussed in the next chapters.

5.4.1 Objects

The OBJECT taxonomy for our application distinguishes four subtypes: PERSON, ABSTRACTION, SUBSTANCE (here, various kinds of LIQUIDS), and MECHANICAL-OBJECT. The last of these subsumes the various tangible objects found in the domain:

- CONNECT-PART is the class of those objects that serve to connect with other objects: PLUG and SOCKET, SCREW and THREADED-PART, etc.
- CONTAINER heads a taxonomy of containers for liquids. They all have an opening for letting liquid in, some also have one for letting it out (e.g., for draining engine oil). Via a HAS-PART relation, they are connected to these openings (see below). There are also different ways of measuring the fluid level inside the tanks; clear plastic tanks (e.g., for windshield wiper fluid) can have a scale imprinted on them, whereas the engine oil is to be measured indirectly with a dipstick. The gasoline tank, on the other hand, resists inspection; the level inside is determined with an electronic instrument.
- CONTAINER-PART: Three important parts of containers are MARKS for reading off fluid levels (only on some tanks), OPENING, and CAP. Openings and associated caps vary: we find simple plastic caps that can be pulled off, caps with threaded joints, or twist-off caps for tanks that are under pressure. Using multiple inheritance, these objects are also subtypes of the respective CONNECT-PART; for example, a THREADED-CAP is a CAP and also a THREADED-OBJECT.
- Various other objects like SPARK-PLUG-WIRE, etc.

5.4.2 Qualities

In the approach to domain modelling explored here, QUALITIES are of comparatively little relevance. Any attribute that plays a role in reasoning, because it can change, is defined as a STATE; what remains is a rather unstructured list of attributes that are not amenable to taxonomization; for instance, whether an action is performed in a QUICK or SLOW manner is taken as a quality. But only few qualities will be needed in the sentence generation examples to follow, so we do not discuss them further here.

5.4.3 States

A taxonomy of STATES (shown in figure 5.6, which extends the STATE node of the SITUATION taxonomy in figure 5.3) captures the temporary attributes of objects, whose transitions are the

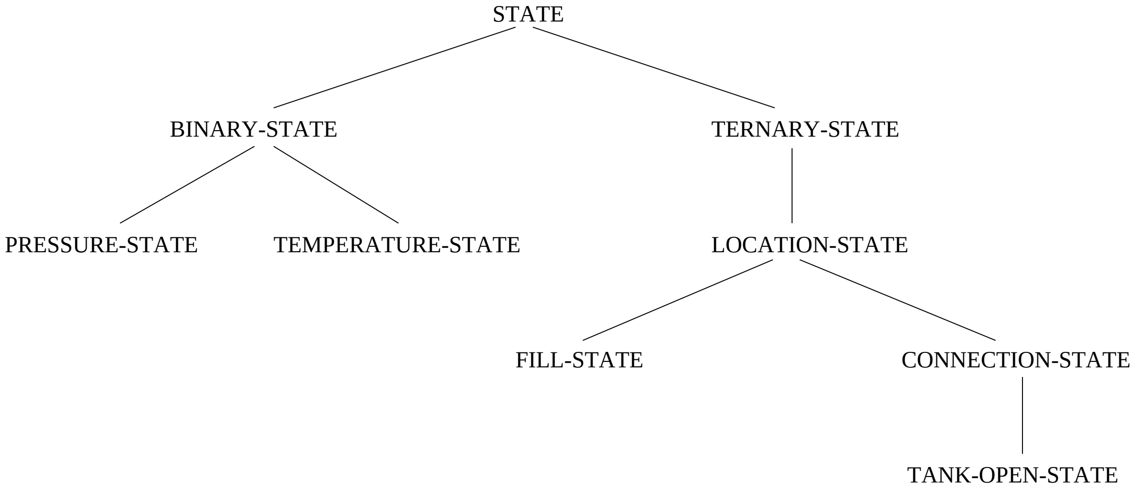

Figure 5.6: Taxonomy of STATES

basic means of expressing change in the model. Those that are relevant for our domain are the location of objects, the temperature of objects, the pressure in a tank, the fluid level in a tank, whether a tank is open or closed, and the state of the connection of a joint, e.g., whether a tank cap is disconnected from or connected to the opening, and whether the connection is loose or tight.

Our STATES relate either two or three entities to one another and thus group into BINARY-STATE and TERNARY-STATE. While discussing them, we can illustrate the various kinds of relation-ranges that LOOM offers; some of the actual definitions (slightly abridged) are shown in figure 5.7. Every BINARY-STATE relates some object to some value, and accordingly has two roles associated with it. The corresponding relations are also shown in the figure; note that for HAS-STATE-VALUE, the range is left unrestricted—this is because no "general" range of all possible state values can be defined. We then distinguish two BINARY-STATES:

- PRESSURE-STATE relates objects of type CONTAINER to appropriate values; how these are modelled depends on the granularity intended, thus on the reasoning to be done with the representation. Here, we are content with a discrete set of four symbolic values, shown in the definition of PRESSURE-STATE-VALUE-SET. Note that the two relations associated with the concept are sub-types of the general HAS-STATE-OBJECT and HAS-STATE-VALUE.

- TEMPERATURE-STATE can relate any object to a temperature value, and here we opt for modelling it with a numeric interval.[2]

As for ternary states, we are dealing with the location of objects, the state of connections, the fill-state of containers, and the state of a tank being open or being closed by a cap, which all relate three different entities. We define LOCATION-STATE as relating a LOCATUM to a LOCATION, which can both be arbitrary OBJECTS. Optionally, we admit a LOCALIZER, which is a symbolic value representing the spatial relationship between the two: INSIDE, ON-TOP-OF,

[2] In fact, LOOM offers some operators for reasoning with such intervals; we could, for instance, define a subtype like COOL-TEMPERATURE-STATE, where the range might be `(:through 0 10)`, and whenever some TEMPERATURE-STATE is created, the classifier would assign either the specific or the more general concept to it, depending on the particular value.

```
(defconcept binary-state :is (:and state
                                   (:exactly 1 has-state-object)
                                   (:exactly 1 has-state-value)))

   (defrelation has-state-object :domain binary-state :range object)
   (defrelation has-state-value  :domain binary-state)
; --------------------------------------------------------------------
(defconcept pressure-state :is (:and binary-state
                                     (:exactly 1 has-press-state-container)
                                     (:exactly 1 has-press-state-value)))

  (defrelation has-press-state-container :is-primitive has-state-object
    :domain pressure-state
    :range container)
  (defrelation has-press-state-value :is-primitive has-state-value
    :domain pressure-state
    :range press-state-value-set)

  (defset press-state-value-set :is (:one-of 'high 'medium 'low 'zero))
; ---------------------------------------------------------------------
(defconcept temperature-state :is (:and binary-state
                                        (:exactly 1 has-tempst-object)
                                        (:exactly 1 has-tempst-value)))

  (defrelation has-tempst-object :is-primitive has-state-object
    :domain temperature-state
    :range object)
  (defrelation has-tempst-value :is-primitive has-state-value
    :domain temperature-state
    :range temp-state-value)

  (defconcept temp-state-value :is (:through 0 200))
```

Figure 5.7: LOOM definitions of BINARY-STATES

```
(defconcept location-state :is-primitive (:and ternary-state
                                               (:exactly 1 has-locst-locatum)
                                               (:exactly 1 has-locst-location)
                                               (:at-most 1 has-locst-localizer)))

(create 'LOC-STATE1 'location-state)
(tell (has-locst-locatum LOC-STATE1 PAPER-SHEET1)
      (has-locst-location LOC-STATE1 CARDBOARD-BOX1)
      (has-locst-localizer LOC-STATE1 'inside))
```

Figure 5.8: LOOM definition of LOCATION-STATE

etc. For example, a sheet of paper can be inside a cardboard box or on top of it; the objects involved are the same, and we mark the difference between the two LOCATION-STATES only with the LOCALIZER. This is one among several ways of modelling LOCATION-STATES, and we choose it because it will facilitate some interesting cases of inheritance. See figure 5.8 for the concept definition and the instantiation of the first cardboard-box example (instance names here given in upper-case letters). Similarly, a FILL-STATE relates a CONTAINER, its CONTENT, and the value representing the extent to which the container is filled; a CONNECTION-STATE relates the two objects connected and the degree to which they are connected.

Now, we can observe that the three ternary states are not quite independent of one another: whenever there is information about the state of a connection, we also know something about the location of the two parts: when the connection between cap and opening is tight, it is also clear that the cap is located on the opening. Similarly, when a tank is closed by a cap, then the cap is connected to the opening, and in turn located on the opening. Also, the fill-state of the tank relates the tank, the substance therein, and the value of the level; and whenever we instantiate a fill-state, we should know as well that the substance is located in the container.

To capture these inferences, both CONNECTION-STATE and FILL-STATE have to be subtypes of LOCATION-STATE, and TANK-OPEN-STATE a subtype of CONNECTION-STATE. We also need subsumption between the relations and their filler-constraints. A LOCATION-STATE relates general OBJECTS—basically, anything can be located anywhere. A FILL-STATE, on the other hand, is restricted to hold between CONTAINER and SUBSTANCE, both subtypes of OBJECT, and a CONNECTION-STATE similarly relates CONNECT-PARTS. A TANK-OPEN-STATE relates a TANK to a CAP; a TANK is also a CONNECT-PART, which is precisely what distinguishes it from other CONTAINERS.[3] Only the respective VALUE-relations cannot subsume one another, as we need different symbolic fillers for them. Figure 5.9 shows all these inter-connections: straight lines denote subsumption between concepts, dashed lines stand for the filler-constraints of the associated relations. For instance, the two roles attached to FILL-STATE are restricted to be filled by a SUBSTANCE and a CONTAINER, respectively.

What is the result of these definitions? Whenever we define a FILL-STATE involving a CONTAINER and a SUBSTANCE, LOOM will also classify it as a LOCATION-STATE involving a LOCATUM and a LOCATION. And, consequently, when our system *verbalizes* a FILL-STATE, then there is always the option to also express it as a LOCATION-STATE. Thus, whenever the generator can say *The tank is full of water* it can also say *Water is in the tank*. This is a case

[3] This is a shortcut; a more sophisticated model would treat only the TANK-OPENING as a CONNECT-PART, but not the entire TANK.

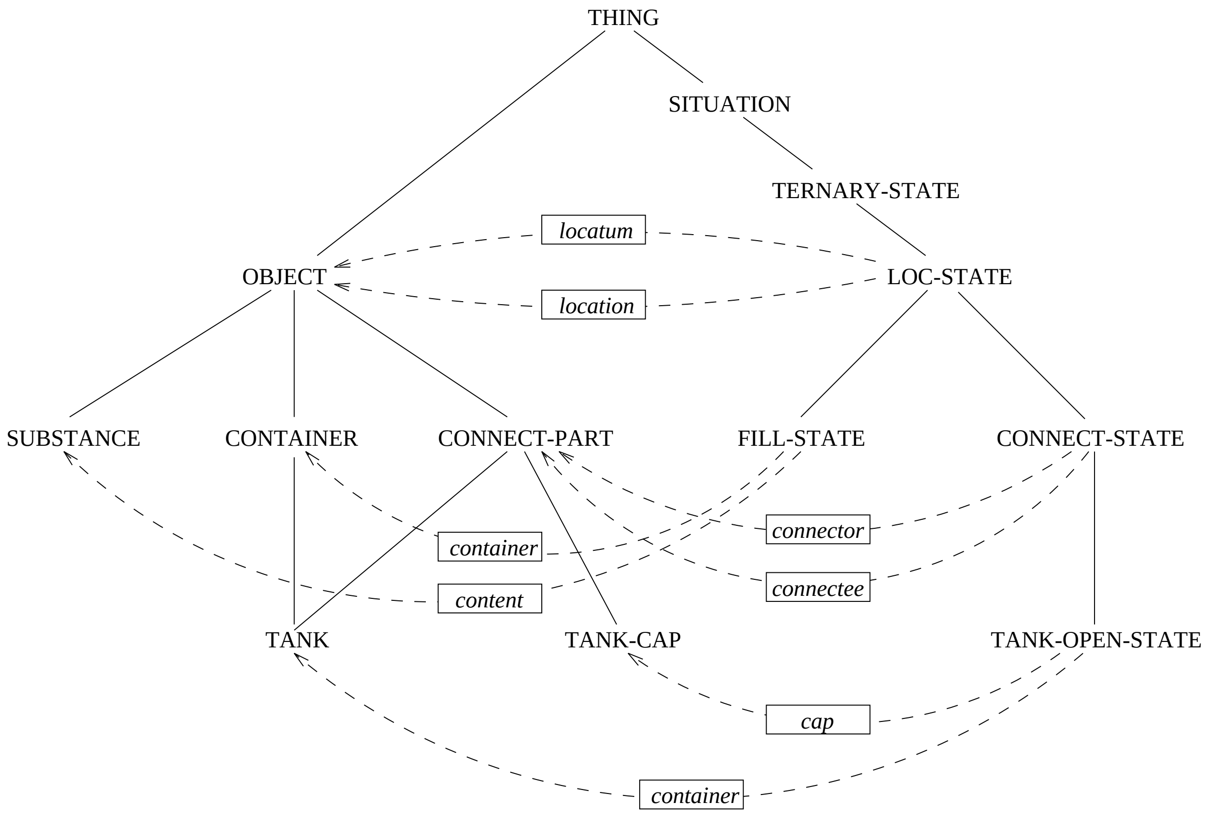

Figure 5.9: Subsumption of concepts and relations for TERNARY-STATES

```
(defconcept path :is-primitive
   (:and abstraction
         (:at-most 1 has-path-source)
         (:at-most 1 has-path-destination)
         (:at-most 1 has-path-direction)))

  (defrelation has-path-source      :domain path :range object)
  (defrelation has-path-destination :domain path :range object
  (defrelation has-path-direction   :domain path :range direction)
    (defset direction :is (:one-of 'upward 'downward 'into 'outof 'left 'right))
```

Figure 5.10: LOOM definition of PATH

of paraphrasing where an inference on the knowledge level is responsible for the two variants—the paraphrase relation is highly situation-dependent and does not follow from general lexical knowledge. (And, of course, the two utterances do not convey exactly the same meaning.)

5.4.4 Activities

As most of the "semantic load" in our system resides within STATES and their combinations in EVENTS, only a few primitives for ACTIVITIES are needed in the domain. Here we will be concerned just with a MOVE concept and some specializations of it. Our modelling of movement is inspired by Jackendoff [1990], who treats it as an OBJECT traversing a PATH, which can be characterized by two places, one of them the SOURCE and the other the DESTINATION. But whereas Jackendoff grants a special ontological status to places, we simply take OBJECT as the filler type of SOURCE and DESTINATION: thereby, anything can move from any object to any other object. If PLACE were a separate entity, we would have to decide whether it is to subsume OBJECT or vice versa, and either way leads into difficulties; since Jackendoff does not organize his 'semantic primitives' in a taxonomy, he is less constrained in this respect.

In addition to SOURCE and DESTINATION, a PATH can have a DIRECTION role, whose filler is a symbol like 'upward or 'left, quite similar to our treatment of the LOCALIZER in the LOCATION-STATE. Figure 5.10 shows the definition; note that all the roles are optional, so that all sorts of combinations are possible to model different kinds of paths.

MOVE is obviously a very general concept, and we need to account for the fact that movement can occur in many different manners. To this end, we introduce a number of sub-concepts denoting specific forms of movement, like DRIP or POUR; we treat these as primitive concepts instead of trying to further decompose them.

Finally, as pointed out in section 5.3, ACTIVITIES are unspecified with respect to *agency*; they can have a CAUSER role associated with them, but they need not. Thus, *The rock fell from the cliff* and *Tom threw the rock from the cliff* would have the same representation except that for the latter there is a CAUSER role associated with the MOVE, filled by TOM (save the fact that we might choose a more specific primitive for *to throw*).

5.4.5 Events

As explained in section 5.3, we treat events as composite entities, consisting of two states and, possibly, an activity. Consequently, the domain model has no single concepts that would

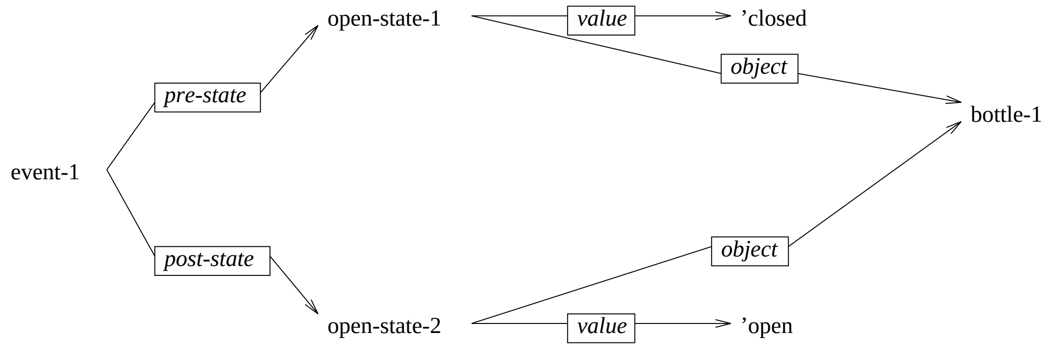

Figure 5.11: Opening the wine bottle as TRANSITION

represent an entire event. A sample event representation was shown in figure 5.4; this is in fact a CULMINATION, because the ACTIVITY causing the state change is present. A less informative version of the same event is given in figure 5.11. Here, only the state change is represented, thus it is a TRANSITION, which could be verbalized as ?*The bottle opened.*

Unlike this contrast between TRANSITION and CULMINATION, the difference between a PROTRACTED-CULMINATION and a MOMENTANEOUS-CULMINATION is not visible in the structure of the event; instead, it merely depends on the type of the embedded ACTIVITY, as can be seen in figure 5.5. It is also possible for a CULMINATION that the embedded ACTIVITY is left underspecified. For example, the event of Jill emptying the wine bottle, with no further information given, would be represented as a TRANSITION from one FILL-STATE of the bottle to another, and there would be only an instance of the general ACTIVITY concept with a CAUSER role filled by Jill.

Many more examples of event representations will follow in later chapters when we discuss verbalization of the various kinds of events in detail.

Chapter 6

Levels of representation: SitSpec and SemSpec

The representations of situations in the domain model, as introduced in the last chapter, are quite distant from specific natural language sentences, and trying to map these structures directly to linguistic output would not be a promising endeavour. Very many decisions need to be made, and, particularly in a multilingual environment, a lot of work would be duplicated if language-specific modules were in charge of the complete mapping, because many decisions will be identical for all the target languages. This chapter thus argues for a division between a language-neutral level of situation specification and an intermediate, language-specific level of semantic sentence representation. By drawing upon language-specific lexical resources, a single language-neutral algorithm can produce the semantic representations for any target language; specifications on this level can then be processed by surface generators and converted to individual sentences.

6.1 Finding appropriate levels of representation in NLG

The relationships between knowledge and language, in particular the dependencies between conceptual and linguistic categories, have been and will be the subject of much psychological and philosophical debate. For our purposes, we take a rather pragmatic stand on the issue, driven by the motivation of building a practical system that can verbalize a given representation in multiple ways and multiple languages. This desire puts us into the same camp as *interlingual machine translation*, which assumes the existence of a level of representation common between two or more natural languages. This level, the *interlingua*, is occasionally labelled as 'language-independent', thereby claiming its categories to be relevant without referring to linguistic terms at all. This claim we regard as too strong; rather, we follow a useful distinction made, for instance, by Hovy and Nirenburg [1992]: Categories in the interlingua are seen not as *independent* of language, but as *neutral* with respect to the particular natural languages the interlingua is being mapped to. That is, instead of granting them a "deeper" existence that every natural language has to respect and build upon, a language-neutral approach merely says that every interlingual representation can be systematically mapped to any of the participating natural languages—but when further languages are added to the system, it is very possible, even likely, that the interlingua will need to be refined in order to account for the new requirements.

6.1.1 Decision-making in sentence generation

Sentence generation in our framework means mapping a specification of a situation to a single English or German utterance, in accordance with a number of parameters, which will be discussed in this and the following chapters. Different parameters can result in quite different utterances, and therefore our task is quite distinct from what is commonly taken as 'front-end generation', where the input to the generation module largely pre-determines the output, and very little parameterization is possible. In effect, we need to systematically convert one representation, which is "understood" by the domain model, to another representation, which is understood by human readers. This task has two aspects to it: producing a syntactically *well-formed* utterance, and ensuring that this utterance best conveys the intended *meaning*.

The formative view Let us first look at sentence generation from a 'formative' perspective, which concerns aspects of the sentence relating purely to its form, not its meaning. From this angle, we can identify the following realization decisions that need to be made when producing sentences:

- Decide on the basic verb/argument structure, and choose the verb lexeme.
- Decide on lexemes for non-arguments, and attach them to the verb.
- Decide on expressions to refer to objects: full NPs, pronouns, or ellipsis.
- Decide on syntactic structure, and possibly choose connectives.
- Decide on constituent ordering.

Obviously, the items on this list are highly interdependent. For example, selecting the verb determines what elements can become arguments, and which will be adjuncts or other modifiers, i.e., non-arguments. For realizing certain modifiers, one often has a choice between a relative clause and an adjective, which is a matter of syntactic structure. And syntactic structure constrains the possibilities for constituent ordering.

The task of choosing referring expressions (see, e.g., [Dale 1992]) is very much a matter of discourse processing: it heavily depends on the preceding context whether something is being referred to with a fully explicit NP, or a shorthand NP, or a pronoun, or an ellipsis. Thus, the construction of suitable referring expressions is largely beyond our concern here; an exception is stylistic choice between nominal groups, which will be covered by our approach (see chapter 7).

In general, formative decisions are largely concerned with the well-formedness of the sentence: we cannot just *choose* to switch the order of a noun and a determiner. And, beyond such strict rules, there are also certain conventions that need to be respected. For example, when several adjectives are to modify a noun, there is very often a 'natural' order to them: everybody notices that *the small green car* sounds all right, while *the green small car* is 'marked', i.e., works only in a specific context where the object referred to is picked from a (mutually known) particular range of cars. Otherwise, it is very important that the unmarked ordering be chosen.

These kind of realization decisions under syntactic or conventionalized constraints are to be left to the generation grammar—they need not be controlled by semantic information. At any rate, all the formative decisions need to be made at some point in the generation procedure. To determine the most suitable point for each individual decision, we have to look at the problem from the perspective of 'meaning'.

The semantic view Obviously, producing language involves much more than having the 'form' right; on the contrary, many formative questions are secondary to considering the *reasons* for uttering a sentence in a particular way. The crucial step in designing a sentence generator is to identify those realization decisions that affect *meaning*, in the widest sense, and then to isolate the parameters that govern the choices between the different formative options.[1]

When looking at sentence generation from this semantic perspective, we can speak of 'goals' that the generator pursues in order to communicate a particular meaning with a sentence. Here, we do not have 'communicative goals' of the sort *be persuasive, be co-operative* and so forth in mind; these work on yet a higher level of abstraction, prior to sentence planning. Instead, we are concerned here with a level of 'semantic goals', which can be related to some of the cases of paraphrase shown in chapter 4.[2] Here is a list of such goals—they will all be discussed in more detail in the following chapters.

- **Cover the whole propositional content with words** Obviously, it is important to verbalize all the units of meaning present in the input specification, and not to leave anything out that ought to be said. In our system, it will be possible to have elements marked as 'optional', though, which can be omitted in the verbalization (see section 6.3).

- **Emphasize certain aspects of the situation** A sentence can make important aspects of the input specification prominent and leave others in the background. *We crossed the river by swimming* emphasizes the achievement of reaching the other side of the river, whereas *we swam across the river* treats the manner in which the crossing occurred as more central.

- **Establish discourse focus** When the sentence, as part of an ongoing discourse, is to place a particular element into the focus, appropriate verb choices can accomplish this: *I spent twenty dollars on that book* renders the amount as most important, whereas *I bought that book for twenty dollars* rather talks about the book.

- **Add certain connotations to the utterance** When rephrasing *to fool someone* as *to pull someone's leg*, the utterance gets a more colloquial tone.

Goals like these may at times very well be in conflict with one another, as we have already noted in chapter 2. For example, choosing a particular verb phrase in order to signal some connotations might render it impossible to assign a prominent enough position to the element in the discourse focus. In such cases, the relative importance of the various goals has to be compared to arrive at a decision. In general, a language generator should pay attention to all these semantic goals and account for the effects they have on the formative decisions—and, potentially at least, every semantic decision can have some effect on any formative decision.

But, at any rate, the individual generation decisions need to be serialized in some way: a generation process just *has* to make its commitments in some order, despite the fact that many, if not most, of the various commitments constrain one another. The important fact is that ordering the realization decisions necessarily implies an ordering of the importance of the

[1] One extreme position is that *every* single formative decision is also one of "making meaning", or in other words: each difference, even the slightest, in surface form corresponds to a difference in meaning.

[2] When embedding the system in some application involving user models and other pragmatic information, the high-level communicative goals (or 'rhetorical goals' in the terminology used by Hovy [1988]) are to be put into correspondence with the semantic goals; but that is beyond our project here.

generation parameters, or of the goals that the generator pursues. Hence, the architectural challenge for NLG is to keep this process as flexible as possible.

Scenarios for generation As a thought experiment, imagine the least flexible sentence generator: it would completely hard-wire the ordering of the formative decisions. When given an input specification, a fixed series of formative decisions is made by inspecting individual parts of the input. Piece by piece the sentence is built up in the order in which the syntactic decisions are made. Such a procedure makes it conveniently easy to end up with a grammatically well-formed utterance—which is one of the goals, of course. However, all the other tasks, which have to do with communicating different kinds of meaning, on the basis of slight variations of the sentence, would be neglected. It is not possible to tailor the sentence to a particular context of utterance on the basis of more information beyond the propositional input specification.

At the other extreme is the ideal, most flexible generator. It starts from the situation specification plus a list of all the parameters that involve the various shades of meaning, together with weights defining their relative importance. This wealth of information gets translated by the generator into a set of formative decisions that collectively give the best possible approximation of all the target parameters: the sentence that is well-formed, expresses the situation correctly, and most closely corresponds to all the goals and their relative weights.

However, as explained in chapter 2, the range of parameters that influence language generation, and lexicalization in particular, is far from being well understood at present. Furthermore, the inter-dependencies between these parameters, whose elicitation would enable a 'holistic' lexical choice, are not clear. What is important at this stage is to devise generation architectures that in principle allow for high flexibility in ordering the decisions that affect meaning, even if the criteria for choice are not all specified in detail.

6.1.2 A two-level approach

When building the domain model in chapter 5, we effectively defined the "deepest" level of representation for our generator: the possible range of situation specifications (*SitSpecs*) that the system can expect as input. Mapping that input to an intermediate semantic representation is the crucial step for all the generation decisions involving meaning. Our examples illustrating the semantic goals given above demonstrate that word choice is a most critical instrument in accomplishing these goals. First, the set of all lexemes chosen for the sentence has to collectively cover all the units of meaning in the input specification. Then, different ways of covering can offer different options for distributing prominence across the units. Furthermore, choosing the main verb can emphasize certain aspects of the situation, and it can place the elements of the situation into the foreground or background as desired. Finally, connotations can result from well-chosen words or phrases.

The process of making all these decisions is *lexicalization*, and we specifically divide it into two, related, subtasks:

- Distribute the units of meaning across the lexemes (chunking; recall the notion of 'incorporation' discussed in chapter 4).

- Select a verb–argument structure that assigns desired degrees of prominence to the different elements of the proposition.

Importantly, these tasks are not semantic goals in themselves; rather, they are the central *means* for accomplishing these goals.

The central resource for making the important generation decisions is therefore the availability of different lexemes with their individual incorporation and structuring behavior. Hence, we take as the very first step in sentence generation collecting all the candidate words that could in principle be used to convey some part of the proposition, and then determining how a complete covering is possible. Such *verbalization options* are lexemes together with possible constraints they place on their environment; they do not necessarily correspond to single words: *to be under pressure* is an example of a phrasal option.

A verbalization option needs to have the information as to what parts of the situation specification it can cover, and in what way it can combine with other verbalization options in forming higher-level linguistic constituents. Furthermore, features need to be present that distinguish lexemes in terms of their focusing behavior and their connotations. Given all this information, the pool of verbalization options effectively defines the search space for the subsequent processes, i.e., for *sentence planning.* And the central task is to actually choose those open-class lexemes that will participate in the sentence, in other words, to do lexicalization. By finding a suitable chunking and by choosing the verb with its configuration of arguments and non-arguments, it is possible to establish the perspective and the focus, and to emphasize an aspect of the situation, insofar as lexical means are available to do so.

Again, when selecting the lexemes it is not feasible to directly produce finished linguistic utterances, because too many steps are still to be taken. Instead, it is important to separate those decisions that are central for building the meaning of the intended utterance from the job of ensuring the grammatical well-formedness of the utterance. To this end, we employ a level of intermediate semantic sentence specification that on the one hand reflects the decisions made in accordance with the semantic goals, and at the same time guarantees that the specification can be verbalized correctly. This level of SemSpecs, as we call them, therefore serves as the interface between the two views introduced above: the formative and the semantic. When building SemSpecs from verbalization options, the various possibilities for achieving as much of the semantic goals as possible can be explored, and at the same time the combinatory rules for building SemSpecs make sure that the resulting expression can indeed be converted to linguistic output by the front-end generation module. In that last step, decisions like the following need to be made:

- Establish the constituent ordering, insofar as it has not yet been fixed by the verb-argument configuration selected.
- Take care of morphology; for example, position the prefix of verbs in the clause (for German), and ensure morphosyntactic agreement.
- Choose function words; for instance, select prepositions on the basis of properties of the objects involved.
- Insert pronouns for intra-sentential anaphora.

In support of this overall division of tasks, we use SemSpecs as a principled level of intermediate, semantic structure from which surface generation can proceed. For the task of sentence production, SemSpec has to fill the "generation gap" that Meteer [1992] described for the task of text planning: the problem of ensuring that the output of some planning module can indeed be converted into a text that is well-formed and in accordance with the communicative goals that the planner had employed. For individual sentences, we are in a somewhat similar position here, exactly *because* we have decided not to organize the domain model strictly along linguistic categories. The SemSpec level has to serve as a bridge to linguistic realization; the next section

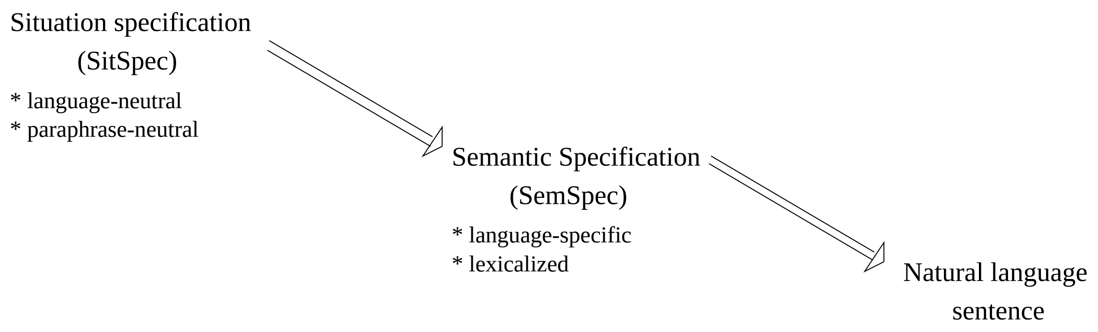

Figure 6.1: Representation levels in the generation system

will explore this goal. As an overview, figure 6.1 depicts the two representation levels; the system architecture to be introduced in chapter 8 will expand on this.

6.2 Linguistic ontology: adapting the 'Upper Model'

In defining our intermediate level of semantic specifications (SemSpecs), we make use of the notion of an 'Upper Model', as it was developed with the PENMAN generator, introduced in section 2.5.2. Recall that a UM is a linguistic ontology whose concepts reflect the distinctions a language makes; the generation grammar draws upon the UM-type of the entities to be verbalized when syntactic decisions are made. Employing an Upper Model to mediate between an application program that performs some *reasoning* operations and a language generator that needs to make *grammatical* decisions is very useful—in particular in a multilingual environment, where language-specific knowledge resources have to be kept separate from the language-neutral domain model. We are, however, in disagreement with two of the key suggestions of the original PENMAN/UM framework: that the domain model be subsumed under the lines of the UM, and that all lexical information be worked into the grammar; hence lexical choice would be treated as one aspect of grammatical realization that cannot be explicitly controlled. This section discusses our alternative proposals on these matters, which will enable us to perform more fine-grained lexical choices and to produce a range of paraphrases that is not possible with strict UM–DM subsumption. But first, we briefly comment on the issue of language-specificity of UMs.

Upper Models for English and German The UM was originally developed for English and meant to be specific for that language, as different languages can have different means for expressing the same content.[3] In the TECHDOC project, several extensions were made to the original English UM, and large parts of a German grammar and UM were developed [Grote 1993]. For languages that are closely related, significant portions of UMs can be identical,[4] and it is sensible to use shared representations where possible.

For our purposes here, the question of whether to have either language-specific or merged, language-neutral Upper Models is a rather uninteresting point to debate, because the answer to such questions depends crucially on the division of labor between UM and grammar, and

[3]The current development of a "Generalized Upper Model" (which incorporates certain new theoretical developments [Halliday and Matthiessen, forthcoming]) appears to take a new stand on the issue of language-specificity.

[4]Henschel [1993] describes efforts on merging an English and a German UM.

on interfacing the two. This matter is beyond our concern, though; we only emphasize the need for language-specific SemSpecs in order to handle a range of divergences between the two languages, as illustrated in the following section, and earlier in chapter 4.

Upper Model and domain model PENMAN was designed as a domain-independent sentence generation module, which should be straightforward to interface with arbitrary application programs. The only prerequisite for using PENMAN with an existing domain model (DM) is linking the DM to the UM, so that every DM concept is subsumed by a UM concept. Then it is possible to directly use DM concepts in SPL expressions, and an application program that operates with DM expressions can construct its SPL expressions and hand them over to PENMAN. In development of this idea, Bateman et al. [1994] proposed to take the UM as a general guideline for ontology building—independent of the specifically linguistic needs of PENMAN. In effect, the proposal was to build any domain model in accordance with the UM. Support for these ideas came from work on applying the UM approach to various other languages (e.g., [Bateman et al. 1991]), which prompted a shift in the underlying philosophy and led to suggesting the UM as an appropriate tool for enforcing ontological consistency in general domain modelling. However, we do not subscribe to this view.

As made clear at the beginning of chapter 4, we see a domain model as highly purpose-specific, or application-dependent: the kind of reasoning to be performed within the model determines the abstractions necessary and the category distinctions that are most convenient for performing that reasoning. Purposes for domain models can differ widely, and it seems neither practical nor attainable at all to enforce for any such purpose a categorization that adheres to the linguistic ontology of some or several natural languages, and our DM was built accordingly.

Sometimes, in a domain model one needs to make abstractions that are not mirrored by the vocabulary of a natural language. Specifically, the *roles* attached to concepts need not always correspond nicely to the semantic roles associated with verbs. Consider, for instance, the taxonomy of CONNECTIONS that our DM uses. The verb *to connect* would correspond to the general CONNECTION concept, and its arguments can be expressed as *to connect A and B*. With the more specific concept PLUG-IN-CONNECTION, however, this pattern is not available; instead, *to plug A into B* requires mapping the conceptual role CONNECTOR to the UM role ACTEE (expressed as direct object), and the CONNECTEE to a DESTINATION role.

Here is another example to demonstrate the need for separating UM and DM. In one branch of the UM, a taxonomy of LOCATIONS represents the distinctions necessary for, amongst other tasks, choosing prepositions. For PENMAN to produce *in the box*, the concept representing the BOX needs to be of the type THREE-D-LOCATION; for *on the table*, the TABLE has be a ONE-OR-TWO-D-LOCATION, and for *at the station* the STATION must be a ZERO-D-LOCATION. Now suppose, for instance, we had a concept FURNITURE-FOR-SLEEPING, subsuming things like BED, FUTON, and others. Then, the verb *to sleep* combines with these objects using different prepositions (*to sleep in a bed / to sleep on a futon*), in spite of their being subsumed by the same domain model concept. Or, consider the example that two moose can meet *at the Danube*, watch the swans *on the Danube*, and afterwards go for a swim *in the Danube*.

All these cases demonstrate the necessity of performing a mapping step from the domain model to the level that respects UM types. After all, the dimension of an object is not an *inherent* feature, but depends entirely on the perspective taken towards that object—we certainly do not want three different DANUBE concepts in our domain model, only to have them subsumed by three different UM concepts. As argued earlier, verbalizing a domain-specific representation can involve restructuring that representation, and type shifts of the kind just illustrated are

<table>
<tr><th>Upper Model</th><th>English</th><th>German</th></tr>
<tr><td>ZERO-D-LOCATION</td><td>at</td><td rowspan="4">an</td></tr>
<tr><td>ONE-D-LOCATION</td><td rowspan="4">on</td></tr>
<tr><td>TWO-D-LOCATION</td></tr>
<tr><td>-VERTICAL</td></tr>
<tr><td>-HORIZONTAL</td><td>auf</td></tr>
<tr><td>THREE-D-LOCATION</td><td>in</td><td>in</td></tr>
</table>

Table 6.1: Correspondence of some prepositions and UM types

one example of restructuring. We will discuss this point in chapter 7.

Furthermore, from the viewpoint of multilinguality, the LOCATIONS are an example of English and German making different distinctions and requiring language-specific UM types [Grote 1993]. The distinction between *at* and *on* in English requires the separation of ZERO-D-LOCATION and ONE-OR-TWO-D-LOCATION. For German, this demarcation is not relevant; instead, the TWO-D-LOCATION needs to be split along a different dimension, to which the English prepositions do not attend: whether the surface is horizontal or vertical. Thus, a picture can be both *on the wall* (vertical) and *on the table* (horizontal), but in German it would be *an der Wand* and *auf dem Tisch*, respectively. Hence, to obtain these language-specific realizations, we need different UM types, as summarized in table 6.1.

Finally, if an event representation were strictly associated with a single UM type, it would require a very powerful generation grammar to derive certain non-trivial paraphrases from that representation. To take an example from chapter 4, it seems questionable whether a grammar could (or should) produce both *Tom drove the books to Sally* and *Tom brought the books to Sally by car* from one and the same specification.

In conclusion, we see a strong need to separate paraphrase-neutral situation specifications from UM-oriented, semantic sentence representations. Mapping from one to the other can require a lot of re-structuring and type-shifting, so that it is not feasible to strictly subsume DM concepts under the UM. (Further arguments are given by Stede and Grote [1995].)

Lexicon, grammar, and Upper Model In systemic-functional grammar, making lexical decisions has a status no different from that of any other grammatical decision. As opposed to other linguistic theories, there is no separation between a set of grammatical rules on the one hand, and a lexicon providing the terminal items for those rules on the other. Instead, the synthesized lexicogrammar weaves word choice into the overall sentence production process. PENMAN, as one implementation of a systemic-functional grammar, generates a sentence by proceeding from higher-level to lower-level *ranks*: it first decides how to realize the process-participant structure (on the clause rank) and chooses the main verb; then the group and phrase ranks are successively realized and lexemes for them chosen.

From the linguistic perspective, we subscribe to the view that a strict separation between 'lexicon' and 'grammar' is not a useful starting point for describing the process of language production. The whole range of phenomena listed in chapter 2 as 'phrasal items' points to the fact that a 'holistic' approach is needed in order to explain the combinatory potential of the various units: free-combining words, collocations of lexemes or lexeme classes, phrasal verbs, idiomatic expressions.

However, we have also noted that these problems are largely unsolved. Not surprisingly then, PENMAN is not particularly strong in producing lexical variation from an input SPL. In the original system, lexical choice is restricted to looking up the lexemes associated with a concept showing up in an SPL, and selecting that candidate whose syntactic-functional features match the set of features that the grammar has determined to be needed at this point of lexical insertion. While it is possible to add choice modules to the grammar that realize certain stylistic preferences within a class of lexemes with identical syntactic behavior [Stede 1993], it is extremely difficult to push the system towards more sophisticated interactions between grammatical and lexical information, e.g., to choose a phrasal lexical item and propagate its syntactic requirements to the grammar.

Conveniently, though, PENMAN offers the option of annotating the SPL expressions with `:lex` terms that directly point to dictionary entries—in other words: to treat the SPL as a fully-lexicalized input to the grammar. While this procedure is not exactly in accordance with the theory underlying the system, it offers the possibility of performing a systematic lexicalization prior to entering the grammar; and this is the path we are pursuing in this thesis. We have emphasized the central importance of lexical choice for accomplishing the semantic goals in sentence generation, and consequently we need to explicitly *control* lexical choice in the generation process. Executing lexicalization first is also advantageous from the perspective of multilinguality. Since the lexical options for verbalizing a situation can vary significantly between languages, it is useful to produce a lexicalized, language-specific semantic sentence representation that can be given to the surface generator.

We therefore re-interpret PENMAN's SPL expressions as fully lexicalized and language-specific SemSpecs, which involves defining a subset of the instruments available for writing SPLs. This definition will be our topic below in section 6.4. Accordingly, we re-interpret the role of the Upper Model as a taxonomy of lexical classes that constrain the combinations of verbalization options; thereby, the UM ensures that the SemSpec can be correctly converted to linguistic output.

6.3 SitSpecs

In order to generate a variety of monolingual and multilingual paraphrases from the same underlying representation, two requirements are important: First, the representation has to be *fine-grained* so as to account for various incorporation possibilities—basically, every unit of meaning that can be incorporated by one lexeme or another needs to be represented as a separate entity. Second, it needs to abstract from the peculiarities of any of the target languages so that it can be mapped straightforwardly to any of them.

To these ends, we introduce a situation specification SitSpec: an instantiation of a SITUATION as defined in chapter 5. As such it is language-neutral as well as paraphrase-neutral, i.e., not implicitly geared towards one particular verbalization. A SitSpec can be built by some application program, or constructed by a text planner that is in charge of dividing an overall text representation into sentence-size pieces, or created with an authoring tool; for our purposes of sentence generation, the exact source does not matter.

Technically, a SitSpec is a directed acyclic graph that is rooted in some instance of type SITUATION. The nodes are names of LOOM instances, but the leaf nodes can also be atomic values of relations ('open, 'closed, etc., as specified in the domain model). Every arc is labelled with a LOOM relation name. When represented as a list, then for every path of the graph (i.e., every sublist) relation names and instance names alternate. Figure 6.2 shows a grammar defining SitSpecs in this way. Not encoded is the requirement that the root node be an instance of type SITUATION. Furthermore, the compatibility of instances and relations is constrained by

```
SITSPEC            ::=  ( INSTANCE RELATION+ )
RELATION           ::=  ( ROLE FILLER )
FILLER             ::=  INSTANCE  |  ATOMIC-VALUE  |  SITSPEC
INSTANCE           ::=  DM-INSTANCE{SALIENCE-LABEL}{OPTIONALITY_LABEL}
ROLE               ::=  DM-RELATION
ATOMIC-VALUE       ::=  'open | 'closed | 'full | ...
SALIENCE-LABEL     ::=  _B  |  _F
OPTIONALITY-LABEL  ::=  _O
DM-INSTANCE        ::= "set of all instances defined in domain model"
DM-RELATION        ::= "set of all relations defined in domain model"
```

Figure 6.2: Syntax of SitSpecs

the DM: it specifies exactly what relations can be attached to what instances, and to what kind of fillers they can point. These restrictions, too, are not visible in the grammar.

Here is an example of a situation specification, denoting an event where a person named Jill puts some water into a tank. The activity is of type POUR, defined in the domain model as a specialization of MOVE, whose OBJECT is to be a LIQUID, and which occurs in a particular manner (that of pouring), which is not analyzed further. Following our convention, instance names are formed by taking the concept name and attaching a number to it. The _F and _O suffixes will be explained below. To enhance readability, we write the relation names in upper-case letters.

```
(event-1 (PRE-STATE    (fill-state-1 (VALUE 'not-full)
                                     (CONTAINER tank-1)))
         (ACTIVITY     (pour-1 (CAUSER jill-1)
                               (OBJECT water-1_O)
                               (PATH (path-1 (DESTINATION tank-1)))))
         (POST-STATE_F (fill-state-2 (VALUE 'full)
                                     (CONTAINER tank-1)
                                     (CONTENT water-1_O))))
```

Figure 6.3 shows the same SitSpec drawn as a graph; this is a more readable form we will use in chapter 9 when discussing more examples in detail. In the graph notation, relation names appear in italics and are surrounded by boxes to indicate that they function as arc labels.

It is important to note that the full functionality of LOOM is "sitting behind" a SitSpec: Every node and every relation are names of actual LOOM objects, and therefore it is possible to execute queries determining, for example, all the more general concepts that `pour-1` is an instance of. Typically, a SitSpec is an *excerpt* from all the relationships actually holding in the knowledge base; that is, the KB has more information about the instances participating in the situation. For example, `tank-1` might be in a `has-part` relation with its filler cap and other parts. The SitSpec represents only those parts of the situation that a text planner has decided to be communicated in the verbalization.

Beyond the conditions for basic well-formedness, we allow SitSpecs to be annotated with both *optionality* and *salience* information.

Any element filling a role inside a STATE or an ACTIVITY can be marked as 'optional', which means that any correct verbalization of the situation can, but need not, cover this element. This optionality for verbalization is not related at all to the optionality of roles

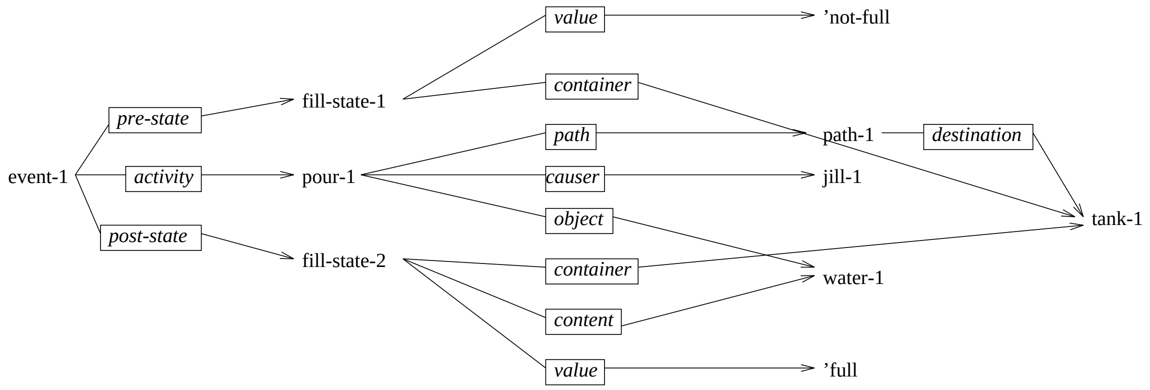

Figure 6.3: Example of situation specification as graph

in the domain model; recall that in chapter 5 we defined, for example, an EVENT as having `:at-most 1 activity`, which renders the ACTIVITY an optional part of an EVENT. But here, we are concerned with a different matter: that in the mapping from SitSpec to language, some element does not necessarily have to be verbalized. The rationale behind this is that a discourse planner that produces SitSpecs might have information about the specific situation of utterance that warrants the exclusion of elements. For example, when the system is to produce an instruction to remove a cap from a tank, and the tank has been talked about in the sentence before, then the instruction *Remove the cap* is fully sufficient. We would not want to *delete* the TANK-instance from the SitSpec, though, because it is an integral part of the event. In fact, it cannot be deleted, because the definition of LOCATION-STATE in the domain model requires its presence in order to be well-formed. Besides, there can always be other reasons to prefer an event verbalization that happens to include the optional element. The notation we use for optionality is an `_O` suffix attached to the instance name (see the example above).

Additionally, the instances of a SitSpec can be given a 'foreground' or 'background' label. As a consequence of this mark-up, the verbalization process will try to find an utterance that assigns a relatively prominent or a less prominent position, respectively, to the word representing the element. What exactly this means will be defined when we discuss the role of salience in generation in section 7.4. But as a well-known example, consider the *passive* alternation, which takes prominence away from the actor and shifts it to the affected object: *Tom filled the tank / The tank was filled by Tom.* We allow salience labels to be attached to any instance, though there is no *guarantee* that it can be accounted for in a verbalization. Foregrounding and backgrounding is, just like stylistic goals, a matter of relative preference. As notation, we use `_B` and `_F` suffixes at the instance nodes, both in the graphic representations and when writing them as lists.

An important restriction is that in a CULMINATION, either the ACTIVITY or the POST-STATE can have a 'foreground' label, but not both, and there can be no 'background' label. The reason is that a verbalization of a CULMINATION can often emphasize either of these two aspects (recall chapter 4, or compare examples (1) and (3) in figure 6.5); but we do not strive to achieve a "graded emphasis" in the sense of producing different, and appropriate, verbalizations for two versions of the same CULMINATION, one of which marks the ACTIVITY with 'foreground' and the other *in addition* the POST-STATE with 'background'. Thus the limitation: Just one 'foreground' feature can be shared between the two elements.

The SitSpec in figure 6.3 has `_F` and `_O` annotations to the effect that the transition should

```
SEMSPEC     ::=  ( VARIABLE / UM-TYPE MOD+ )
MOD         ::=  KEYWORD SEMSPEC  |  KEYWORD VARIABLE  |
                 :lex LEX-POINTER  |  :name STRING
KEYWORD     ::=  :domain  |  :range  |  :actor  |  :actee  |  :limit  |  ...
VARIABLE    ::=  x1  |  x2  |  ...
UM-TYPE     ::=  "set of all concepts defined in upper model"
LEX-POINTER ::=  "set of all morphosyntactic entries defined in lexicon"
STRING      ::=  "arbitrary string of characters"
```

Figure 6.4: Syntax of SemSpecs

be foregrounded and the water be optional; this configuration would, for instance, make *Jill filled the tank* a preferred verbalization.

6.4 SemSpecs

SemSpecs, as pointed out above, are special kinds of SPL expressions [Kasper 1989]. In fact, the range of possible SemSpecs is a subset of the range of possible SPL expressions, because we do not use their entire potential. SPL was developed for PENMAN, and as a consequence, an SPL expression can contain very specific directives to PENMAN's grammar, which are needed when the sentences to be produced become complex. For SemSpecs, though, we use only the most central of SPL's instruments; after all, we are developing our approach only for quite simple sentences here. While restricting the expressive power of SPL, we at the same time make the demand that a SemSpec be fully lexicalized, whereas a general SPL expression need not be. More precisely, the SemSpec contains pointers to morphosyntactic entries, from which PENMAN can produce a correctly inflected word form. We name these lexical entries with the suffix `_el` if they belong to the English lexicon, and `_gl` if they belong to the German one.

The basic syntax of SemSpecs is shown in figure 6.4. SPL and PENMAN have more `KEYWORDS` than shown here; we list only some of those used in our generation examples to follow. Again, there are some additional restrictions not encoded in the grammar, though. The outermost UM-type (the "root") must be subsumed by the UM-type PROCESS. The names of variables in a SemSpec must be distinct unless referring to the same entity. Hence, if the same variable occurs more than once, it must not be associated with conflicting sub-SemSpecs.

Just as the DM specifies what relations combine with concepts, the UM specifies what keywords can be or must be present with a certain UM-type. For example, `:name` can only be attached to specific kinds of OBJECTS, and `:domain` and `:range` only to RELATIONAL-PROCESSES. Finally, there is the requirement that a `:lex` keyword and filler be associated with any UM-type in the SemSpec, apart from RELATIONAL-PROCESSES. These always tie entities together, and their linguistic realization is decided by the grammar—it could be a copula, a preposition, a connective, or no lexical item at all, e.g., an assignment of genitive case for POSSESSION in German. Since the UM guards the well-formedness of a SemSpec, we are guaranteed that it can be mapped to a well-formed English or German sentence—with one exception: the UM does not know what lexemes are allowed to be attached to what UM-types. Thus, it could happen that in a SemSpec some OBJECT is annotated with a verb lexeme (and there is no nominalization intended or possible). To avoid this, we have to make sure that the *partial* SemSpecs, from which a complete SemSpec is produced, are lexically sound. Partial

(1) *Jill poured water into the tank until it was filled.*

```
(x1 / anterior-extremal :domain (x2 / directed-action :lex pour_el
                                  :actor (x3 / person :name jill)
                                  :actee (x4 / substance :lex water_el)
                                  :destination (x5 / three-d-location :lex tank_el))
                        :range (x6 / nondirected-action :lex fill_el
                                  :actee x5))
```

(2) *Jill füllte Wasser in den Tank, bis er voll war.*

```
(x1 / anterior-extremal :domain (x2 / directed-action :lex fuellen_gl
                                 :actor (x3 / person :name jill)
                                 :actee (x4 / substance :lex wasser_gl)
                                 :destination (x5 / three-d-location :lex tank_gl))
                       :range (x6 / property-ascription
                                 :domain x5
                                 :range (x7 / quality :lex voll_gl)))
```

(3) *Jill filled the tank with water.*
(4) *Jill füllte den Tank mit Wasser.*

```
(x1 / directed-action :lex fill_el
                     :actor (x2 / person :name jill)
                     :actee (x3 / object :lex tank_el)
                     :inclusive (x4 / substance :lex water_el))
```

Figure 6.5: Semantic specifications and corresponding sentences

SemSpecs will be introduced in section 7.2.

Figure 6.5 shows a number of SemSpecs that can all be derived from the SitSpec in figure 6.3 above—the exact procedure will be explained later. The corresponding output produced by PENMAN and its German variant is also shown. The last example in the figure is annotated with the English lexemes; the exact same specification with German lexemes can be given to the German module. The system producing these SemSpecs, which will be developed in the next chapters, has to account for the facts that *to fill* and *füllen* behave the same to a certain extent (examples 3 and 4 in figure 6.5) but not entirely: The German verb can undergo the locative alternation and appear in configuration (2), whereas English needs a different verb to get the parallel structure (1). The general difference between (1,2) and (3,4) is that the latter express only the result of the action, whereas the former add information on how that resulting state came about. Returning to the issue of salience, (1,2) would be produced when the ACTIVITY of the SitSpec is marked with 'foreground', and (3,4) are the preferred verbalizations when the POST-STATE has a 'foreground' label.

To summarize, a SemSpec is one particular, lexicalized version of the propositional content of the utterance to be produced, the basic configuration of process, participants, and circumstances.[5] PENMAN derives the surface-linguistic realization on the basis of the combination of UM-types in the SemSpec. The SemSpec is still underspecified with regard to a few formative decisions. PENMAN will produce a default constituent order, which can be over-

[5]Which keywords correspond to participants and which to circumstances depends on the process type. We will say more about the distinction in the next chapter.

written with a `:theme` directive in the SemSpec (see section 7.4). Also, PENMAN makes some pronominalization decisions by itself: If the same entity is referred to more than once in the SemSpec (i.e., with the same variable), then its second realization in the sentence will be with an appropriate pronoun, as can be seen in the examples (1,2) in figure 6.5.

As they stand, SemSpecs are not specific to the PENMAN surface generator. Similar generators expect similar inputs, and could in principle be used instead. The important requirement is that the UM-types have to be known to the generator so that it can derive the right verbalizations.

In our system, SemSpecs are constructed from a SitSpec by selecting a process and mapping SitSpec elements to participant roles of that process, so that all elements of the SitSpec are covered, i.e., it is ensured that they take part in the verbalization. The mechanisms will be introduced in the chapters to come; as a central preparatory step, we now need to consider the role and representation of word meaning in our model.

Chapter 7

Representing the meaning of words: a new synthesis

As we have seen in the previous chapters, any language generator needs knowledge about the meanings of the words it can use. Specifically, we have stressed the need for separating the different kinds of lexical information, so that a generator can make an informed choice among candidate paraphrases of an utterance. Thus, in contrast to previous language generators where lexical information consists simply of the concept denoted by the word and morphosyntactic features, we will in this chapter step by step develop lexical entries consisting of several components:
`NAM`: The *name* of the lexical entry;
`LAN`: The natural *language* the entry applies to;
`DEN`: The *denotation* of the word: its applicability condition with respect to SitSpecs;
`COV`: The subset of SitSpec nodes actually *covered* by the word;
`PSS`: A *partial SemSpec*: the contribution the word can make to sentence meaning;
`CON`: The *connotations*: a list of stylistic features and values;
`SAL`: For verbs only: the *salience assignment* on the participants and circumstances;
`AER`: For verbs only: pointers to *alternation and extension rules* that apply to the verb.
A central point to be made is the separation of denotation and partial SemSpec. One of the benefits of this treatment will be the possibility of deriving more complex verb entries from simpler ones by means of alternation and extension rules, to be explained in section 7.3.

7.1 Denotation and covering

The central theoretical distinction between denotation and connotation was discussed in section 3.4. In this section, we explain the treatment of denotation in the lexical representations of our system. To illustrate the task, consider the group of lexical items *to die, to perish, to pass away, to kick the bucket*, which all refer to the same event, the death of some animate being, but which differ in their stylistic 'color'. From the perspective of knowledge representation, we want to avoid having four different DIE concepts in the KB merely to gain the ability of generating the four different items (not to mention the addition of similar items in other languages); the distinctions made in the KB should be geared foremost towards the underlying *reasoning* tasks and not towards subtleties in possible verbalizations. In other words, we do not want the grain-size of the conceptual representation to be determined by the grain-size of the lexicons of the languages we want to generate.[1] Hence, we assume a single concept DIE, and in turn have to represent the differences between the lexical items in another way. One difference is that the items do not equally apply to the same class of entities: anything that lives, including plants, can *die*, but *pass away*, according to Cruse [1986], applies only to human beings, whereas both humans and animals can be said to *kick the bucket* or *perish*. Correspondingly, the German *sterben* is translation-equivalent to *die*, *entschlafen* is a formal word used for human beings, and *abkratzen* is a rather vulgar word for the death of animals and humans. Selectional restrictions of this kind can be treated in the link between words and knowledge base, because the KB provides exactly the suitable means for representing them. On the other hand, the words clearly differ in terms of their tone or their formality, which is to be represented with features associated to lexical items, outside of the knowledge base.

Thus, the basic idea is that coarse denotational differentiation between words occurs in the link between knowledge base and lexicon, hence in the denotations. On the other hand, fine-grained connotational differentiation occurs in a different component of the lexical entries, which will be the topic of section 7.5.

7.1.1 SitSpec templates

In section 3.4 we limited the denotation of a word, for our purposes, to those aspects that can be treated in a taxonomic knowledge base, and decided to leave other—more fine-grained—semantic features outside the scope of our framework. When performing language generation from a KB, we are then in a position to give a clear, operational definition of a word's denotation—because this part of word meaning is responsible for linking the word to the domain model, which is well-defined. The denotation has to match some elements of the input representation, which thereby opens up the possibility of using that word in the verbalization. In effect, the denotation amounts to the necessary *applicability condition* of a word: it has to be present in the input to the generator for the word to be considered at all. Going back to the discussion of concept–word linking, if a simple one-to-one mapping between KB entities and lexical items is assumed, then the denotation is trivially the lexeme's one and only concept. In the system presented here, however, the mapping can be more elaborate; thus matching lexeme denotations against a SitSpec is a more complicated task. Besides involving entire subgraphs, it cannot simply check for identity but has to respect subsumption: A word denoting a general state of affairs must be available to express a more specific situation, too.

To enable the matching, a denotation is defined as a *partial SitSpec*, or a SitSpec *template*, which may contain variables. In the case of lexemes denoting OBJECTS, this template can possibly reduce to the simple case of a single concept,[2] but with events and the corresponding verbs the situation is more interesting.

The effect of linking lexical items to concepts *and* roles is that we can represent more finely grained semantic distinctions than those made by the concepts only: similar lexical items all map onto the same, fairly general, semantic predicate, and the associated roles and fillers represent the smaller denotational differences.

As a first simple example, consider the different words denoting DIE. They differ in their connotations, which are the topic of section 7.5, and in the class of entities they can apply to,

[1] This contrasts with approaches like that of Emele *et al.* [1992], who deliberately introduce a new concept wherever there is a word in one of the target languages to be generated.

[2] This depends, of course, on the granularity of the OBJECT branch of the knowledge base; it is perfectly possible to decompose objects and thereby arrive at more complex denotations for nouns, but we ignore this here.

which is a matter of denotation. Let us assume that in the domain model, the entity undergoing death is linked with the relation EXPERIENCER to the concept DIE. Here are the denotations representing the different selectional restrictions of the various English and German words (the term `V <TYPE>` is a variable `V` with a restriction on the type of its value):

```
to die, sterben
(die (EXPERIENCER (V living-thing)))

to pass away, entschlafen
(die (EXPERIENCER (V person)))

to perish, to kick the bucket, abkratzen
(die (EXPERIENCER (V animate-being)))
```

When these are matched against a SitSpec representing the death of someone or something, only those words whose selectional restriction subsumes the type of the EXPERIENCER in the SitSpec will be valid options for verbalizing the situation. Assuming that in the domain model LIVING-THING subsumes PLANT and ANIMATE-BEING, and ANIMATE-BEING in turn subsumes PERSON, then the death of some plant can be denoted only by *to die* and *sterben*, whereas all the verbs are available to describe the death of a person.

Note the fact that we are using the traditional notion of *selectional restriction* on two different levels here: in the domain model, the concept DIE can restrict its EXPERIENCER role to the general class ANIMATE-BEING. The various lexemes can then impose more fine-grained restrictions by using the specific subtypes of ANIMATE-BEING in the denotation, as shown.

For a more interesting example of a denotation, we return to the situation of Jill filling some container with water, which was introduced in chapter 6. The SitSpec is repeated below, together with the denotation of *to fill* in its causative reading: It says that the word can be used in any situation where a fill-state whose value is not identical to `'full` changes into another fill-state of the same container, where the value is now `'full`. Some unspecified activity that has a causer is responsible for the transition.

SitSpec for, e.g., *Jill filled the tank with water*:

```
(event-1 (PRE-STATE  (fill-state-1 (VALUE 'not-full)
                                   (CONTAINER tank-1)))
         (ACTIVITY   (pour-1 (CAUSER jill-1)
                             (OBJECT water-1)
                             (PATH (path-1 (DESTINATION tank-1)))))
         (POST-STATE (fill-state-2 (VALUE 'full)
                                   (CONTAINER tank-1)
                                   (CONTENT water-1))))
```

to fill (causative):

```
(event (PRE-STATE (fill-state (VALUE (not 'full))
                              (CONTAINER A)))
       (ACTIVITY (CAUSER B))
       (POST-STATE (fill-state (VALUE < D 'full >)
                               (CONTAINER A)
                               (CONTENT C))))
```

The denotation contains variables that are bound to instances or atomic values of the SitSpec when the two are matched against each other. Here, `A` will be bound to `tank-1`, `B` to `jill-1`, and `C` to `water-1`.

```
DENOTATION    :=  ( TYPE RELATION* )
RELATION      :=  ( ROLE FILLER )
FILLER        :=  VARIABLE  |  ( VARIABLE TYPE )  |  DEFAULT  |
                  RELATION+  |  ( VARIABLE RELATION+ )  |  ( VARIABLE TYPE RELATION+ )
                  |  ATOMIC-VALUE  |  ( not ATOMIC-VALUE )
DEFAULT       :=  ( < VARIABLE ATOMIC-VALUE > )
TYPE          :=  DM-CONCEPT
ROLE          :=  DM-RELATION
ATOMIC-VALUE  :=  'open | 'closed | 'full | ...
VARIABLE      :=  a | b | c | ... | v1 | v2 | ...
DM-CONCEPT    ::= "set of all concepts defined in domain model"
DM-RELATION   ::= "set of all relations defined in domain model"
```

Figure 7.1: Syntax of a lexeme denotation

Defaults As an important aspect of lexical meaning, we provide in our system the possibility that words encompass *default* values as part of their meaning. In the entry given above, the term `(VALUE < D 'full > )` in the `POST-STATE` is an example of a default value, which we denote with angle brackets.[3] The semantics is the following: Matching this branch of the denotation against the corresponding branch of the SitSpec always succeeds. If the value in the SitSpec is different from the default value, then the variable (here, `D`) is bound to the value in the SitSpec. If the two values are identical (which is the case here), then the variable remains unbound, and for the corresponding position in the partial SemSpec we thus have the information that the value need not be verbalized separately. Intuitively speaking, *to fill* implies that the tank ends up full; *Jill filled the tank* conveys exactly this. But it is perfectly all right to say *Jill filled the tank to the second mark*—now the value has to be made explicit in the verbalization, because it differs from the default.

The bound variables are used in other parts of lexical meaning: the covering information (see below), and the partial SemSpecs, which will be introduced in section 7.2. In the *fill* example, variables occur only at the leaf nodes, but in principle they can also be at internal nodes, if an instance name needs to be bound (examples will follow in the next chapter). Also, any variable can be associated with a type restriction, as shown in the DIE examples above.

To sum up this description, figure 7.1 gives the syntax of denotations. It is, of course, very similar to the SitSpec grammar in figure 6.2; denotations may have in addition: defaults, negated atomic values, and variables that can be placed anywhere in the denotation, possibly with type restrictions.

As with the DIE example above, a central point is that when determining the applicability of a lexeme, we use the inheritance relationships as defined in the LOOM KB. The word *to tank up*, for instance, is largely synonymous with *to fill*, but it applies only to the gasoline tank of a vehicle, instead of the general containers for which *to fill* is defined. Thus, *to tank up* would have the same denotation, except that the filler of the role `CONTAINER` is to be restricted to the type `gas-tank-in-vehicle`. Then, for a SitSpec representing somebody filling a gasoline tank, both *to tank up* and *to fill* would be found as lexical candidates.

[3] For parsing a denotation, the angle brackets are, strictly speaking, redundant; but for the human eye they make it easier to notice the presence of a default value.

7.1.2 Covering

Words serve to verbalize parts of SitSpecs. When forming sentences that are supposed to verbalize SitSpecs *completely*, we need a measure for which parts, exactly, are expressed or *covered* by individual lexemes. After all, we have to make sure that every non-optional SitSpec element is somehow covered. And at the same time, we want to avoid elements being covered more than once; for example, we do not want to produce a verbalization that incorporates an INSTRUMENT in the verb *and* expresses it separately: *I flew to Hawaii by plane* makes sense only in a specific context where the hearer might have expected, for instance, that the speaker had gone by helicopter. In general, such an utterance is to be avoided.

If a complete verbalization covers every node of a SitSpec exactly once, and lexemes cover subsets of the SitSpec nodes, then the joint of the coverings of the lexemes participating in the sentence is the set of all SitSpec nodes. That is, roughly, the picture; we will refine it a little in chapter 8.

Besides the denotation, we therefore associate with a lexeme a list of nodes covered by it. An obvious constraint is that the covering-list cannot contain an element that is not part of the lexeme's denotation. In other words, a lexeme cannot express some chunk of meaning that is not part of the applicability condition of the lexeme. And how about the opposite question: Can the denotation contain elements that are not covered by the lexeme?

Typically, all the nodes appearing in a lexeme's denotation are also on the covered-list, except for the external variables—they stand for entities that will be covered by the lexeme filling their position. Thus, the covered-list for a noun like *water*, whose denotation is simply `(water)`, is `(water)`. Upon matching the denotation against a SitSpec, the general type `water` in the denotation is replaced with the name of the instance in the SitSpec matching it, say `water-1`. Accordingly, the covering-list of the lexeme becomes `(water-1)`, so that at the end of the matching phase, all the 'instantiated' lexemes (which have been successfully matched against some portion of the SitSpec) have in their covering-lists the actual instance names or atomic values from the SitSpec, and no longer the generic types. These instantiated lexical entries we call 'verbalization options', or *vo*s for short.

For a more interesting example, let us consider the causative *to fill* again. When this verb is used, it expresses the change from one fill-state to another; hence it covers both the `pre-state` and the `post-state` of the SitSpec. Furthermore, since we are dealing with the causative reading, the verb expresses the fact that some activity brought the transition about, which is also covered. And finally, as we have noted when discussing the default, *to fill* covers the `value` of the `post-state`, which is `'full`. In order to place both states and the activity on the covered list, they need to be referred to with variables, so that the denotation is slightly more complex than shown above. To distinguish variables that are co-indexed with the partial SemSpec (see next section) from those that only appear on the covering-list, the latter are always named with the letter `V` followed by a number. Here are the denotation and the covering-list:

to fill (causative):

```
DEN: (event (PRE-STATE  (V1 fill-state (VALUE (not 'full))
                                        (CONTAINER A)))
            (ACTIVITY   (V2 (CAUSER B)))
            (POST-STATE (V3 fill-state (VALUE < D 'full >)
                                        (CONTAINER A)
                                        (CONTENT C))))

COV: (V1 V2 V3 < 'full >)
```

But the covering-list need not always contain all the nodes of the denotation. Sometimes, a lexeme is applicable only in a specific context; this characterization of context is necessarily encoded in the denotation, but it is not expressed by the lexeme, and hence not on its covered-list. For example, *open* as a predicative adjective can verbalize a TANK-OPEN-STATE. Its denotation is the complete `(tank-open-state (VALUE 'open))`, but the word covers only the value `'open`. The state itself needs to be covered by a different lexeme, e.g., by a copula, which will then provide the link to the thing that is open.

7.1.3 Aktionsart

The role of the denotation needs now to be related to the notion of *Aktionsart*, which was introduced in section 3.6: the verb-inherent features characterizing (primarily) the temporal distribution of the event denoted. For a language generator that is to produce different descriptions of events represented in an underlying domain model, the Aktionsart categories are highly relevant if its capabilities are to move beyond dealing with simplistic input like `read(sally,book)`. Both the domain model from which the generator receives its input and the lexical specifications need to be rich enough to provide the information required.

As we pointed out in section 3.6, the variety of phenomena in both aspect and Aktionsart are far from clear-cut, and there is no generally accepted and well-defined set of Aktionsart features. In the following, we use the terms given by Bussmann [1983] and discuss only those Aktionsart features that are directly relevant for us because they relate types of SITUATIONS to denotations of verbs. In fact, within the context of our system, we provide a clear definition of Aktionsart features in terms of verb denotations.

Simple cases are *stative* verbs like *to own* or *to know*. According to Bussmann, they denote properties or relations that do not imply change or movement and that cannot be directly controlled by the participating entities; therefore, such verbs cannot be used in the imperative mood: **Own a car!* **Know the chancellor!* The denotation of these verbs in our system is of the form `(state X)`. Many states, though, can be verbalized with an adjective and the copula *to be* or *sein*, respectively: *The car is clean. Das Auto ist sauber.*

For the rest, the basic dichotomy is that between *durative* and *non-durative* verbs. The former characterize continuous events that do not have internal structure, like *to sleep, to sit*. In our framework, these verbs denote situations of the type PROTRACTED-ACTIVITY.

In the class of non-durative verbs we find, amongst others, the opposition between *iterative* and *semelfactive* ones. The former are durative activities that result from repeating the same occurrence. In German, these are sometimes morphologically derived: *sticheln* is a derivative of *stechen* ('to poke') and denotes continuous poking; the *-eln* morpheme occurs with the same meaning in a number of other verbs as well. In contrast, a *semelfactive* verb denotes a single occurrence, thus in our system a MOMENTANEOUS-ACTIVITY, as for example *to knock* or the just-mentioned *to poke*.

Transformative verbs involve a change of some state, without a clearly recognizable event that would be responsible for it: *The room lit up.* The denotation of such verbs involves a pre-state and a post-state, which is the negation of the former: `(event (PRE-STATE A) (POST-STATE not-A))`. *Resultative* verbs, on the other hand, characterize situations in which something is going on and then comes to an end, thereby resulting in some new state. These verbs have a denotation with the pattern `(event (ACTIVITY A) (POST-STATE B))`. In the literature, such verbs are often also called *inchoative*.[4]

[4]The term 'inchoative' is used to cover a rather broad range of phenomena, including the beginning of an event (e.g., *to inflame*) or its coming to an end; recall that Jackendoff [1990] discusses the 'inchoative' reading of

Aktionsart	Denotation pattern
stative	`(state X)`
durative	`(protracted-activity X)`
semelfactive	`(momentaneous-activity X)`
transformative	`(event (PRE-STATE X)` `(POST-STATE not-X))`
resultative	`(event (ACTIVITY X)` `(POST-STATE Y)`
causative	`(activity (CAUSER X))`

Table 7.1: Correspondences between verb denotation and Aktionsart

Another verb-inherent feature is *causative*: A causative verb denotes a situation where an agent performs an activity. Some verbs can be used for situations both with or without an agent, as for example *to fill: The tank filled / Tom filled the tank.*

To summarize, in our system a number of Aktionsart features can be defined in terms of verb denotations: If the denotation follows a certain pattern, then the respective Aktionsart feature is associated with the verb. Table 7.1 lists the correspondences.

Of course, there are more features pertaining to Aktionsart (which is a notoriously fuzzy area anyway), which cannot be reflected within our model of situations. To account for the difference between "one-way" (*die, kill*), "full-cycle" (*flash, hit*), and "multiplex" (*breathe, beat*) situations [Talmy 1988], a yet more fine-grained model of activities distributed over time would be required.

7.2 Partial SemSpecs

Besides linking word meaning to the underlying knowledge representation and naming features for isolated properties of words, it is necessary to account for *compositional* meaning: the behavior of words in a sentence when combined with other words. To this end, we associate with each lexeme a partial SemSpec that characterizes its combinatory potential on a semantic level of description. In effect, this partial SemSpec defines the *case frame* of the verb.

7.2.1 Lexico-semantic combinations

The combinatory potential of words can be described in syntactic or in semantic terms. From the perspective of syntax, a transitive verb requires a subject and a direct object in order to become 'saturated'. Here, we provide the description of lexical combinations on a semantic level, namely that of SemSpec as introduced in chapter 5. On this level, the notion corresponding to the transitive verb is a process of a particular type, e.g., DIRECTED-ACTION, which requires two participants to be specified completely. To describe such requirements, we associate with each lexical entry—in addition to its denotation—a partial SemSpec that characterizes the contribution that the word can potentially make to a sentence SemSpec. The generation algorithm, which will be explained in chapter 8, can then systematically create a sentence SemSpec by unifying the partial SemSpecs of the lexemes to be used.

to fill. We therefore think the term is overloaded and prefer to use 'resultative' for the latter group.

```
PSEMSPEC      ::=  ( INT-VARIABLE / UM-TYPE MOD+ )
MOD           ::=  KEYWORD EXT-VARIABLE  |  < KEYWORD EXT-VARIABLE >  |
                   KEYWORD (ts EXT-VARIABLE UM-TYPE)  |
                   :lex LEX-POINTER  |  :name STRING
KEYWORD       ::=  :domain  |  :range  |  :actor  |  :actee  |  :limit  |  ...
INT-VARIABLE  ::=   x1  |  x2  |  ...
EXT-VARIABLE  ::=  a  |  b  |  ...
UM-TYPE       ::=  "set of all concepts defined in upper model"
LEX-POINTER   ::=  "set of all morphosyntactic entries defined in lexicon"
STRING        ::=  "arbitrary string of characters"
```

Figure 7.2: Syntax of partial SemSpecs

A partial SemSpec, or PSemSpec for short, is thus defined much like a general SemSpec, with one major exception: The PSemSpec can contain *external variables* following keywords, and these are to be bound by other PSemSpecs. By 'external variables' we mean variables different from those that are defined within a SemSpec, i.e., the variables in the line
`SEMSPEC ::= ( VARIABLE / UM-TYPE MOD+ )`
of the SemSpec grammar given in figure 6.4. All external variables occurring in a PSemSpec must also occur in the denotation of the lexeme; but the denotation can have additional variables for inclusion in the covering-list, as pointed out earlier. Figure 7.2 gives the syntax of PSemSpecs. Internal and external variables are abbreviated as `INT` and `EXT`. Another difference between a PSemSpec and a SemSpec is that the outermost UM type of a PSemSpec does not have to be subsumed by PROCESS, because a PSemSpec can correspond to elements of different kinds.

We call a PSemSpec with external variables *unsaturated* and one with no such variables *saturated*. Among the lexemes with saturated PSemSpecs are the nouns, denoting objects, e.g., *tank*: `(x / object :lex tank_el)`, and proper names, which are arbitrary strings that do not point to lexical entries: `(x / person :name jill)`.

The standard group of *unsaturated* PSemSpecs is those associated with verb lexemes. We repeat here the denotation for the causative reading of *to fill* and add its PSemSpec:

```
(event (PRE-STATE  (fill-state (VALUE (not 'full))
                               (CONTAINER A)))
       (ACTIVITY (CAUSER B))
       (POST-STATE (fill-state (VALUE < D 'full >)
                               (CONTAINER A)
                               (CONTENT C))))

(x / directed-action :lex fill_el :actor A  :actee B  < :inclusive C > )
```

The PSemSpec will be saturated as soon as other (saturated) PSemSpecs replace the variables `A, B`, and `C`. The mechanism for this step will be explained in chapter 8, and we will continue the discussion of this example there.

When lexemes in English and German are synonymous, we do not want to store the identical information twice. For example, the English *tank* and German *Tank* behave exactly the same. They share the same denotation and covering-list and the same PSemSpec—but they have to point to their individual morphosyntactic feature set. In the PSemSpec, we therefore have the

```
SitSpec: (location-state-1 (LOCATUM swan-1)
                           (LOCATION danube-1)
                           (LOCALIZER 'on-top-of))

---------------------------------------------------------
Lexeme:         Danube
Denotation:     (river (NAME 'danube))
Covering-list:  (river 'danube)
PSemSpec:       (x / object :name danube)
---------------------------------------------------------
Lexeme:         swan
Denotation:     (swan)
Covering-list:  (swan)
PSemSpec:       (x / object :lex swan_el)
---------------------------------------------------------
Lexeme:         on-location
Denotation:     (location-state (LOCATUM A)
                                (LOCATION B)
                                (LOCALIZER 'on-top-of))
Covering-list:  (location-state 'on-top-of)
PSemSpec:       (x / nonorienting
                   :domain A
                   :range (ts B one-or-two-d-location))
---------------------------------------------------------

Result of unification:
(x1 / nonorienting
    :domain (x2 / object :lex swan_el)
    :range (x3 / one-or-two-d-location :name danube))

---------------------------------------------------------

Output: "The swan was on the Danube."
```

Figure 7.3: Example for type shifting

possibility of filling the `:lex` keyword with a list of two pointers, here `:lex (tank_el tank_gl)`. The generation algorithm will then use the correct pointer, depending on the target language that the user has selected for the sentence. This possibility of sharing parts of lexical entries is an important feature of our approach, enabled by the strict separation of the various parts of lexical information.

7.2.2 Type shifting

The division between denotation and PSemSpec also offers us a way of accounting for *type shifting* phenomena that we discussed under the heading 'two-level semantics' in section 3.5. Recall the examples given by Bierwisch [1983], for example the different readings of nouns like *school* or *sonata*.

Within our framework, the notion of type shifting plays a role in the move from SitSpec to SemSpec. Objects can be seen from a particular viewpoint and possibly need a UM-type different from the 'standard' type in their PSemSpec. For example, the standard 'lexical' UM-type of *school* would be the general OBJECT, but in a sentence like *The boys walked into the school* it semantically acts as a THREE-D-LOCATION.

To illustrate this, recall the example of *at/on/in the Danube* given in chapter 6. In its PSemSpec, the lexeme *Danube* has the UM-type OBJECT, but when it participates in a LOCATION-STATE, the LOCALIZER determines the viewpoint and thus the dimensionality of the LOCATUM. The lexicon entries that express LOCATION-STATES are thereby responsible for appropriately shifting the UM-type of the lexeme expressing the LOCATION. To this end, the term `(ts <VAR> <UM-TYPE>)` can occur in a PSemSpec in the place of the single `<VAR>`, and when the variable is replaced by a SemSpec in the unification process, the outermost type of that SemSpec is replaced with the new `<um-type>`. To make this clearer, figure 7.3 shows the SitSpec, the participating lexical entries, and the resulting SemSpec for the sentence *The swan was on the Danube*.[5] In the entry for `on-location`, which verbalizes the root node of the `location-state`, the variable `B` in the denotation matches the SitSpec node `danube-1`. The PSemSpec associated with the lexeme covering that node (which is `Danube`) undergoes the type shift once it replaces the `B` in the PSemSpec of `on-location`: `object` becomes `one-or-two-d-location`.

To demonstrate that this approach also works for cases other than spatial relationships, consider a slight variation of one of Bierwisch's examples: *Faulkner is hard to read.* We would standardly define the verb *to read* has having a selectional restriction for its object to be of a type like WRITTEN-MATTER, and *Faulkner* as an instance of PERSON. Then, *$*$reading Faulkner* is an ungrammatical expression and cannot be generated. With the additional knowledge that *Faulkner* is also an AUTHOR, though, we can state in the lexical entry of *to read* a type shift that extends the denotation of *Faulkner* from the person to the writings he has produced, and accordingly shifts the UM-type in the PSemSpec from PERSON to WRITTEN-MATTER. This would be a general rule applying to all instances of AUTHOR, and it can be handled conveniently because of our separating denotation from PSemSpec in characterizing word meaning.

7.2.3 Valency and the Upper Model

Since the Upper Model is our basic instrument for ensuring the well-formedness of PSemSpecs and SemSpecs, we now have to examine the role of the Upper Model in characterizing verbal case frames, continuing the discussion of valency in section 3.7. We will in this section uncover some deficiencies of the UM approach with respect to lexical valency, and propose an improvement within our generation framework.

As we have stated earlier, the Upper Model is rooted in the process classification of systemic-functional linguistics, as developed by Halliday [1985]. He also stresses the distinction between *participants* and *circumstances*, which was explained in section 3.7. In their description of the UM, Bateman *et al.* [1990, p. 8] characterize participants as "in some sense essential to the performance, or 'actualization' of the process" and circumstances as providing "additional contextualizing information such as temporal and spatial location, manner of performance of the process, purposes, etc." Significantly, the precise distribution of participants and circumstances depends on the type of process. There are four basic types (corresponding to subtrees in the PROCESS taxonomy of the UM), which differ in the way participants can be realized syntactically:

- MATERIAL-PROCESSES have the participants ACTOR and ACTEE.

[5] In all our generation examples, we abstract from tense and definiteness; see the remark at the beginning of chapter 9.

- VERBAL-PROCESSES have a SAYER, an ADDRESSEE, and a SAYING.
- MENTAL-PROCESSES have a SENSER and a PHENOMENON.
- RELATIONAL-PROCESSES can have a variety of participants, depending on the specific type of process.

Concerning the linguistic realizations, Bateman *et al.* note that participants are typically realized as nominal groups (with some obvious exceptions, as in *say that x*), and circumstances often appear as prepositional phrases. These are only tendencies, though, as we have already shown in section 3.7.

On the one hand, it has often been pointed out that the distinction between participants and circumstances is difficult to make; Halliday acknowledges this, and the creators of the Upper Model are also aware of the problem. On the other hand, the UM cannot have fuzzy boundaries but has to make some clear distinctions. Thus, the required participants for process types, as listed above, are coded in LOOM as obligatory roles. Furthermore, for specific process types, the roles can be value-restricted. Circumstances, on the other hand, are coded as LOOM relations, and there are no restrictions as to what circumstances can occur with what processes.

Limitations Given the sample domain for our generation system, we are concerned here predominantly with MATERIAL-PROCESSES, whose taxonomy in the UM was explained in section 2.5.2 and depicted in figure 2.2. More specifically, we are dealing to a large extent with verbs of physical movement. While the UM has a subtype of NONDIRECTED-ACTION for MOTION-PROCESSES, there are no additional constraints on valency encoded with this process.

Of the spatio-temporal aspects of situations, many can indeed be clearly classified as circumstances, and they are consistently expressed with adverbs or prepositional phrases: something happened *yesterday / on Monday*, and it occurred *in the city*. But, as was pointed out, neither the syntactic division corresponding to participants and circumstances (direct or indirect object versus adverbs or prepositional phrases) nor the semantic postulate that spatio-temporal aspects are circumstances hold *in general*. Focusing on spatial relationships, we find verbs that specifically require *path*-expressions, which cannot be treated on a par with circumstances; recall *to put*, which requires a direct object and a destination. Causative *to pour* requires a direct object as well as a path with either a source, or a destination, or both: *pour the water from the can into the bucket*.[6] Some verbs, as is well-known, can occur with either a path (*Tom walked into the garden*) or with a place (*Tom walked in the garden*), and only the latter can be treated as a standard circumstance. And consider *to disconnect*, which requires a direct object (the entity that is disconnected) and a source-expression (the entity that something is disconnected from). The source can be omitted if it is obvious from the context; in chapter 4 we have cited the instruction *Disconnect the wire*. But the source expression, e.g., *from the plug*, does not have the status of a spatial circumstance like *in the garage*.

The Upper Model in its present form cannot make distinctions of this kind. It is not possible to specify a PATH expression as an obligatory participant, and it is not possible to represent the difference in valency for *to walk* in *walk in the garden / walk into the garden*. About *to disconnect*, which is a MATERIAL-PROCESS, the UM can only state that the roles ACTOR and ACTEE must be filled (given the causative reading), but not the fact that there is another entity involved—in the domain model we called it the CONNECTEE—which can be verbalized as a

[6]We disregard the reading found in *Tom poured the wine*; such utterances can become conventionalized because the path is obvious in the situation.

SOURCE. Moreover, the UM does not know that the CONNECTEE is optional in the verbalization.

Improvements As a step forward to a more fine-grained distinction between participants and circumstances, we propose to differentiate between requirements of process types (as coded in the UM) and requirements of individual verbs, which are to be coded in the lexical entries. In a nutshell, *lexical valency* needs to supplement the participant/circumstance requirements that can be stated for types of processes.

Essentially, we wish to distinguish these cases:

- *Tom disconnected the wire {from the plug}.*
  To disconnect requires a SOURCE, but it can be omitted in a suitable context.

- *Tom put the book on the table.*
  To put requires a DESTINATION, and it cannot be omitted.

- *The water drained {from the tank} {into the sink}.*
  To drain requires some PATH expression, either a SOURCE, or a DESTINATION, or both. But (in this reading) it cannot occur with no PATH at all.

- *In the garage, the water drained from the tank.*
  Locative circumstances like *in the garage* are not restricted to particular verbs and can occur in addition to PATHS required by the verb.

To capture these differences, we differentiate the participants into *obligatory* and *optional* ones, similar to the distinction made by Helbig and Buscha [1991] between 'obligatory complements' and 'optional complements'.

To encode the valency information, we use the instrument of the partial SemSpec, which for verbs serves as the case frame by listing the obligatory and the optional participants. Here, obligatory participants are to be stated as absolutely required. That is, the verb is only applicable if the elements denoted by these participants are present in the SitSpec, as we have explained in the previous section. Optional participants, while also part of the case frame, are marked as optional for verbalization; when a sentence SemSpec is built, it need not necessarily include them. In the PSemSpec, we use angle brackets to indicate this. For *to disconnect*, the PSemSpec thus is the following:

```
to disconnect: (x / directed-action :actor A  :actee B  < :source C >)
```

Genuine circumstances, as distinguished in the UM, do not appear in the lexical entry of a verb; instead, as is common practice, general adjunct rules are responsible for them. They will be introduced in the next section. But how, exactly, can we motivate the distinction between optional participants and circumstances in our framework? By relating the PSemSpec to the SitSpec, via the denotation. In the *disconnect* case, for instance, the two items CONNECTOR and CONNECTEE are both integral elements of the situation. The situation would not be well-formed with either of them absent, and the domain model encodes this restriction. Therefore, both elements also occur in the denotation of *to disconnect*, as shown below, and a co-indexed variable provides the link to the PSemSpec. Only when building the sentence SemSpec is it relevant to know that the CONNECTEE can be omitted (in particular, if it is in the SitSpec marked as 'optional' for verbalization). The CONNECTEE in the denotation therefore must have its counterpart in the PSemSpec—that is the SOURCE, but there it is marked as optional.

```
(event (PRE-STATE (connection-state (CONNECTOR B)
                                    (CONNECTEE C)
                                    (VALUE (not 'disconnected))))
       (ACTIVITY (CAUSER A))
       (POST-STATE (connection-state (VALUE 'disconnected))))
```

With adjuncts, the situation is different: A SitSpec is complete and well-formed without the information on, for instance, the location of an event. Hence, a verb's denotation cannot contain that information, and it follows that it is not present in the PSemSpec, either.

7.3 Alternations and extensions

Having explained denotations and PSemSpecs, specifically for verbs, we now face the task of accounting for the different *alternations* a verb can undergo, as discussed in section 3.8. One simple option is to use a separate lexical entry for every configuration, but that would clearly miss the linguistic generalizations. Instead, we wish to represent the common "kernel" of the different configurations only once, and use a set of lexical rules to derive the alternation possibilities.

7.3.1 Alternations as meaning extensions

In section 3.8, we mentioned Jackendoff's [1990] proposal to use primitives like INCH and CAUSE for deriving related verb configurations. From our NLG perspective, the idea of deriving complex verb configurations from more basic ones is very attractive, but for our purposes we have to relate verb meaning to our treatment of event structure, instead of masking that with a primitive like INCH.

When verbalizing a SitSpec, we first have to determine candidate lexemes, i.e., match the SitSpec against lexicon entries; having only one lexicon entry for a verb reduces the search space dramatically. Moreover, since the verb entry will be the most basic form, its denotation is relatively simple and therefore the matching is inexpensive. Finding more complex verb configurations will then require some further matching, but *only* locally and to those verbs that have already been determined as verbalization options.

Therefore, the idea is to see verb alternations not just as relations between different verb forms, but to add directionality to the concept of alternation and treat them as functions that map one into another. As we noted in section 3.8, there are two groups of alternations:

(1) Alternations that do not change meaning, i.e., the denotation of the verb;

(2) Alternations that do change the denotation of the verb.

The critical group is (2), because if we derive verb configurations from others and rewrite the denotation in this process, it has to be ensured that the process is monotonic, so that the process of applying the rules will terminate. Therefore we define the directionality for group (2) to the effect that an alternation always *adds* meaning: the newly derived form communicates more than the old form—the denotation gets extended. We thus assume the existence of a minimal base form of a verb, from which extension rules will proceed. This notion of extension is different from the standard, non-directional way in which alternations are seen in linguistics; to label the difference, we henceforth call alternations of group (2) *extensions*. In this section, we will introduce a number of extension rules for which we can give a clear definition in terms of Aktionsart features, as they were given in section 7.1.3. These rules extend both the denotation of a verb and rewrite its PSemSpec to reflect the change; the result is a new verbalization option, which can differ from the previous one in terms of coverage or attribution of salience (see the next section). The Aktionsart of the verb is thus projected from a more basic one to a more complex one. The rules will be conveniently simple, thanks to the Upper Model, which provides the right level of abstraction from syntax.

We illustrate our goal with an example. If a SitSpec encodes the situation of Tom removing all the water from a tank, then the verb *to drain* is a candidate lexeme. While it can appear in a number of different configurations, we wish to match only one of its forms against the SitSpec. This is the most basic one, denoting an ACTIVITY: *The water drained from the tank.* Here, the case frame of the verb has to encode that *from the tank* is an optional constituent. Now, an extension rule has to systematically derive the CAUSATIVE form: *Tom drained the water from the tank.* And also from the first configuration, another rule derives the RESULTATIVE reading, which adds the information that the tank ended up empty: *The tank drained of the water.* Here, *of the water* is an optional constituent. To this last form, a 'causative' extension can apply and yield *Tom drained the tank of the water.*

To compute such configurations automatically, we define an alternation or extension rule as a 5-tuple with the following components:
`NAM`: a unique name;
`DXT`: extension of denotation;
`COV`: additions to the covering-list;
`ROC`: role changes in PSemSpec;
`NRO`: new roles: list of additional PSemSpec roles and fillers.
The `DXT` contains the denotation subgraph that the new verbalization has in addition to the old one. The syntax is, of course, the same as that of the denotation of a lexical entry. Specifically, it can contain variables; these can co-occur in the `COV` list: the items that the new verbalization covers in addition to those of the old one. `ROC` is an ordered list of pairs that exchange participant role names or the UM-type in the PSemSpec; this replacement can also change optionality. For example, `(< :actee > :actor)` means "replace the term `:actee` in the PSemSpec of the old verbalization, where it was optional, with obligatory `:actor`." Finally, `NRO` contains new roles and fillers that are to be added to the new PSemSpec; these will also contain variables from the denotation extension.

Applying such a rule to a verbalization option vo works as follows: Add the contents of `DXT` to the denotation of vo, and match the new part against the SitSpec. If it matches, make a copy vo' of vo and assign it a new name as well as the denotation just formed. Add the `COV` list, which has been instantiated by the matching, to the covering-list of vo'. Exchange the role names in the PSemSpec of vo' as prescribed by `ROC`, and, importantly, in the order they appear there. Finally, add `NRO` to the PSemSpec.

Before introducing several rules now, two final points should be emphasized: First, note that we do not provide *applicability conditions* for the alternation and extension rules. Instead, they are triggered directly from the lexical entry of a verb. Whether general applicability conditions can be specified, so that the rules need not be attached to each individual verb that undergoes the alternation, is exactly the open research question that we have mentioned in section 3.8 when discussing Levin's work. The second point is that the rules are not specific to a target language. In our system, any English or German verb can trigger any alternation and extension rule. Again, this is due to our using these rules on the level of SemSpec, and is thus due to the Upper Model, which abstracts over language-specific syntax.

7.3.2 Lexical rules for alternations and extensions

Passive Example: *Tom emptied the bucket / The bucket was emptied by Tom.* Of the alternations that do not affect the denotation, we first consider the passive. This alternation rule is very simple, as the functionality we need is already encoded in the UM: If the participant role `:agentive` is used instead of `:actor`, PENMAN produces a passive sentence. Hence the rule is:

```
NAM: passive
DXT: ()
COV: ()
ROC: (:actor :agentive)
NRO: ()
```

This leaves the denotation unchanged and merely replaces one keyword in the PSemSpec. The dative alternation can be handled similarly.

Substance–source Example: *The tank leaked water / Water leaked from the tank.* This is an alternation discussed by Levin [1993] for verbs of 'substance emission, for example *drip, radiate, sweat,* and *leak*[7]. To make use of this alternation here, we have to add directionality and declare one of the two configurations as more basic. For making that decision, we use the fact that in *The tank leaked water* the *water* is an optional constituent, hence the minimal configuration of the verb is *The tank leaked.* With the *from* configuration, no deletion is possible.

To show a representative of the verb class, here are the denotation and PSemSpec of *to leak*:

```
DEN: (leak (OBJECT A)
           (PATH (SOURCE B)))

PSS: (x / nondirected-action :lex leak_el :actor B  < :actee A >)
```

The following extension rule applies to all these 'substance emission' verbs and derives the *from* configuration:

```
NAM: substance-source
DXT: ()
COV: ()
ROC: ((:actor :source) (< :actee > :actor) (nondirected-action directed-action))
NRO: ()
```

Let us now consider several alternations that change denotation, and hence are extensions.

Stative–resultative Example: *Water filled the tank / The tank filled with water.* In discussing verbs that denote a STATE, Jackendoff [1990] points out that *fill, cover, surround,* and *saturate* can describe either a STATE or an inchoative event, and encodes the difference with the primitive INCH, as mentioned in section 7.3.1. Our goal is to do without the primitive, and to define the change in terms of the Aktionsart of the verb; to this end, we use RESULTATIVE in the place of 'inchoative' (see section 7.1.3).

[7] Unnoticed by Levin, *to leak* can also be a verb of substance "intrusion", as in *The camera leaked light.* This reading reverses the directionality of the PATH involved; we do not handle that reading here.

On a similar matter, Levin [1993] describes the 'locatum subject' alternation, which for instance holds between *I filled the pail with water* and *Water filled the pail.* It thus relates a causative and a non-causative form. Levin states that the alternation applies to a class of 'fill verbs', which can be described as follows (p. 120): "When the argument that is the object of *with*—the locatum—is expressed as the subject, the sentence can be understood as describing a state [Jackendoff 1990]. These verbs typically describe the resulting state of a location as a consequence of putting something on it or in it." Levin lists many more verbs of filling than the four given by Jackendoff, and her alternation is not exactly the one we need here, since it also involves a causative form—deriving this, however, is yet another step.

What we need here is a mixture of Jackendoff's and Levin's insights: Several of Levin's 'fill verbs' can be both transitive and intransitive; and some of the intransitive readings denote 'to become Xed'. Among these verbs are *fill, flood, soak, encrust,* and *saturate*: *The kitchen flooded with water* means the same as *The kitchen became flooded with water.* For this subgroup of the 'fill verbs' we define an extension rule that derives from a STATE reading a RESULTATIVE one. Note that this is different from Levin's 'locatum subject' alternation, since it does not involve a causer.

```
NAM: stative-resultative
DXT: (event (Y (ACTIVITY X)))
COV: (X Y)
ROC: ((:actor :inclusive) (:actee :actor) (directed-action nondirected-action))
NRO: ()
```

To illustrate the rule with an example, consider the denotation and PSemSpec of the STATE reading of *fill*:

```
DEN: (fill-state (CONTAINER A)
                 (CONTENT B)
                 (VALUE C))

PSS: (x / directed-action :lex fill_el  :actor B  :actee A  < :destination C >)
```

When matching it against a SitSpec with a tank and water, like the one shown earlier in section 7.1.1, then (ignoring the `VALUE` for now) this yields the verbalization *The water filled the tank*, covering only the POST-STATE of the SitSpec. Now, the alternation rule extends the denotation to also covering the EVENT and the ACTIVITY that brings the filling about. Applying the changes to the PSemSpec results in

```
(x / nondirected-action :lex fill_el :inclusive B :actor A < :destination C >)
```

from which PENMAN produces *The tank was filled with the water.*

In German, the RESULTATIVE verbs that are not CAUSATIVE are typically reflexive: *Der Tank füllte sich mit Wasser* (lit. 'The tank filled itself with water'). The surface generator is aware of this, so at the level of SemSpec there need be no difference between English and German.

A few stative verbs cannot be RESULTATIVE without being also CAUSATIVE. Consider *to cover* in these examples from Jackendoff:

Snow covered the ground.

∗*The ground covered with snow.*

Bill covered the ground with snow.
For these, a 'stative–culmination' extension derives the RESULTATIVE+CAUSATIVE form directly from the STATIVE one. The rule is similar to the one given above, so we do not show it here.

Causative extensions Example: *The napkin soaked / Tom soaked the napkin.* Levin discusses a 'causative–inchoative' alternation that applies to a large number of verbs. The class formed by them is somewhat heterogeneous with respect to the Aktionsart, though; it contains for example *to move* as well as *to open*. The former is in its basic form DURATIVE (*The cat moved*), and the latter TRANSFORMATIVE (*The door opened*). Accordingly, we split the alternation in two, which differ only in the `DXT` component, reflecting the difference in Aktionsart. The alternation adds a CAUSER to the denotation, makes the former `:actor` the new `:actee`, and accordingly changes the overall UM-type from NONDIRECTED-ACTION to DIRECTED-ACTION, because there is now an ACTEE present.

```
NAM: durative-causative
DXT: (activity (CAUSER X))
COV: ()
ROC: ((:actor :actee) (nondirected-action directed-action))
NRO: (:actor X)

NAM: resultative-causative
DXT: (event (ACTIVITY (X (CAUSER Y))))
COV: ()
ROC: ((:actor :actee) (nondirected-action directed-action))
NRO: (:actor Y)
```

The first rule derives, for example, *Tom moved the door* from *The door moved*, and the second *Tom closed the door* from *The door closed.*

Locative extensions Example: (a) *Sally sprayed the wall with paint.*/ (b) *Sally sprayed paint onto the wall.* We have mentioned the locative alternation in our introduction to the topic in section 3.8; in our new terminology it belongs to the group of extensions. Its characteristic is that one configuration of the verb (a) conveys that something is performed in a 'complete' or 'holistic' manner, whereas the other configuration (b) lacks this facet of meaning.[8] Levin points out that this alternation has received much attention in linguistics research and notes that, in spite of the efforts, a satisfactory definition of the 'holistic'-facet has not been found. Jackendoff, in his treatment of the alternation, suggests encoding the 'holistic' feature in a primitive: the function ON_d is a derivative of ON and means that something 'distributively' covers a surface, e.g., the paint covers all of the wall. Introducing a primitive, though, amounts to conceding that no explanation in terms that are already known can be given. We cannot solve the question of 'holisticness' either, but we want to point to the fact that the two verb configurations correlate with a change in Aktionsart: *Sally sprayed paint onto the wall* is durative (she can do it *for two hours*), whereas *Sally sprayed the wall with paint* is transformative (she can do it *in two hours*). That observation leads us to propose that the example is best analyzed as involving a mere ACTIVITY in the *with* configuration, and an additional TRANSITION in the *onto* configuration.

Support for this analysis comes from Pinker [1989], who postulates a change in meaning when moving from one configuration to the other: In (b) above, Sally causes the paint to move

[8] In fact, sentence (a) can be read both as conveying the holistic aspect and not doing so; we disregard this ambiguity and focus on the holistic reading, as the alternation researchers in linguistics did as well.

Sally sprayed paint onto the wall.

```
(spray-1 (CAUSER sally-1)
         (OBJECT paint-1)
         (PATH (path-1 (DESTINATION wall-1))))
```

Sally sprayed the wall with paint.

```
(event-1 (PRE-STATE (covered-state-1 (OBJECT wall-1)
                                     (VALUE (not 'covered))))
         (ACTIVITY (spray-1 (CAUSER sally-1)
                            (OBJECT paint-1)
                            (PATH (path-1 (DESTINATION wall-1)))))
         (POST-STATE (covered-state-1 (OBJECT wall-1)
                                      (VALUE 'covered))))
```

Figure 7.4: SitSpecs for sentences corresponding to configurations of *to spray*

onto the wall, whereas in (a), Sally causes the wall to change its state by means of moving the paint onto it. Pinker sees (a) as derived from (b) and suggests as constraint on the applicability of the alternation that the motion (here: *spray*) *causes an effect* on the surface.

While we decided not to discuss applicability conditions here, we support the idea that the difference between (a) and (b) can be expressed with an additional state change. In our framework, different input SitSpecs result in the two sentences, one ACTIVITY and one EVENT, as shown in figure 7.4.

The crucial point now is that the first SitSpec is fully embedded in the second; this is in correspondence with the truth conditions: If Sally has sprayed the wall with paint, then she also has sprayed paint onto the wall. To generalize the correspondence to an extension rule, we need to assume in the domain model a concept like COMPLETION-STATE, which is to subsume all those STATES in the domain model that have "extreme" values: an empty bucket, a fully loaded truck, and so forth. The exact interpretation of COMPLETION-STATE is the open question that Levin [1993] referred to, and that Jackendoff treated with his '$_d$' subscript. We do think, though, that an abstract STATE in the domain model, which subsumes a range of the concrete STATES, is preferrable to introducing a primitive on the linguistic level (unless the primitive is relevant for other linguistic phenomena as well).

The following alternation rule applies to durative verb readings that denote ACTIVITIES of something being moved to somewhere, and extends them to also cover the POST-STATE, which must be subsumed by COMPLETION-STATE. In this way, it derives reading (a) from (b) in the *spray* example, and analogously for the other verbs undergoing the alternation, e.g.: *Tom loaded hay onto the wagon / Tom loaded the wagon with hay*; *Jill stuffed the feathers into the cushion / Jill stuffed the cushion with the feathers*. The PSemSpec is modified as follows: The former `:destination` (*wall*) becomes the new `:actee`, whereas the former `:actee` (*paint*) now fills the role `< :inclusive >`, and is optional there, because *Jill sprayed the wall* is also well-formed.

```
NAM: locative-transitive
DXT: (event (MOVE (OBJECT X)
                  (PATH (DESTINATION Y)))
```

```
           (POST-STATE (Z completion-state (OBJECT Y))))
COV: (Z)
ROC: ((:actee < :inclusive >) (:destination :actee))
NRO: ()
```

Most of this rule covers two kinds of locative alternation, which Levin distinguishes: the 'spray/load' alternation and the 'clear (transitive)' alternation. The latter applies only to the verbs *clear, clean, drain, empty* and can be seen as the 'semantic inverse' of the spray/load alternation, because one group of verbs denotes activities of placing something somewhere, and the other describes activities of removing something from somewhere; but both have the same 'holistic' effect in one of the verb configurations. For example, the rule derives *Tom drained the container of the water* from *Tom drained the water from the container.* Thus, the rule for the clear-alternation is the same as the one shown above, with three exceptions: the keyword replacing `:actee` is not `< :inclusive >` but `< :of-matter >`, the `DESTINATION` in the denotation is a `SOURCE`, and correspondingly, the keyword `:destination` is `:source`.

The German verb *füllen*, which is only sometimes translation-equivalent to *fill*, undergoes the locative alternation, as we have mentioned at the very beginning of the thesis. Thus, our rule appears in the lexical entry of *füllen* and thus derives *Tom füllte den Tank mit Wasser* from the base form *Tom füllte Wasser in den Tank.* With *fill*, this operation is not possible.

The *clear* verbs, except for *to clean*, can in addition be intransitive, and Levin states a separate alternation for them. For *to drain*, the first configuration is *The water drained from the tank*, and the second is either *The tank drained* or ?*The tank drained of the water.* According to Levin, "the intransitive form may be best in the absence of the *of*-phrase" [Levin 1993, p. 55]. The SitSpec denoted by the first configuration is:

The water drained from the tank.

```
(move-1 (OBJECT water-1)
        (PATH (path-1 (SOURCE tank-1))))
```

Note that our durative–causative extension rule given above applies in this case and extends the coverage of the SitSpec to the one corresponding to *Tom drained the water from the tank.* A rule that is parallel to that for the transitive case is given below; it derives ?*The tank drained of the water*; since the `< :of-matter >` is optional, we can also produce *The tank drained*, which is, according to Levin, preferred.

```
NAM: locative/clear-intransitive
DXT: (event (MOVE (OBJECT X)
                  (PATH (SOURCE Y)))
           (POST-STATE (Z completion-state (OBJECT Y))))
COV: (Z)
ROC: ((:actor < :of-matter >) (:source :actor))
NRO: ()
```

Summary The extensions introduced now can be applied in a sequential order to a verb. Figure 7.5 provides a synopsis: The boxes contain the denotation patterns and the corresponding Aktionsart feature, and the arcs are labelled with the names of the rules that transform a configuration with one Aktionsart into another. In this graph, every verb base form has an entry point corresponding to the Aktionsart of its most basic configuration. Examples: *to fill* is STATIVE, *to drain* is DURATIVE, *to open* is TRANSFORMATIVE, *to remove* is RESULTATIVE+CAUSATIVE. The "double box" in the middle is the entry point for both TRANSFORMATIVE and RESULTATIVE

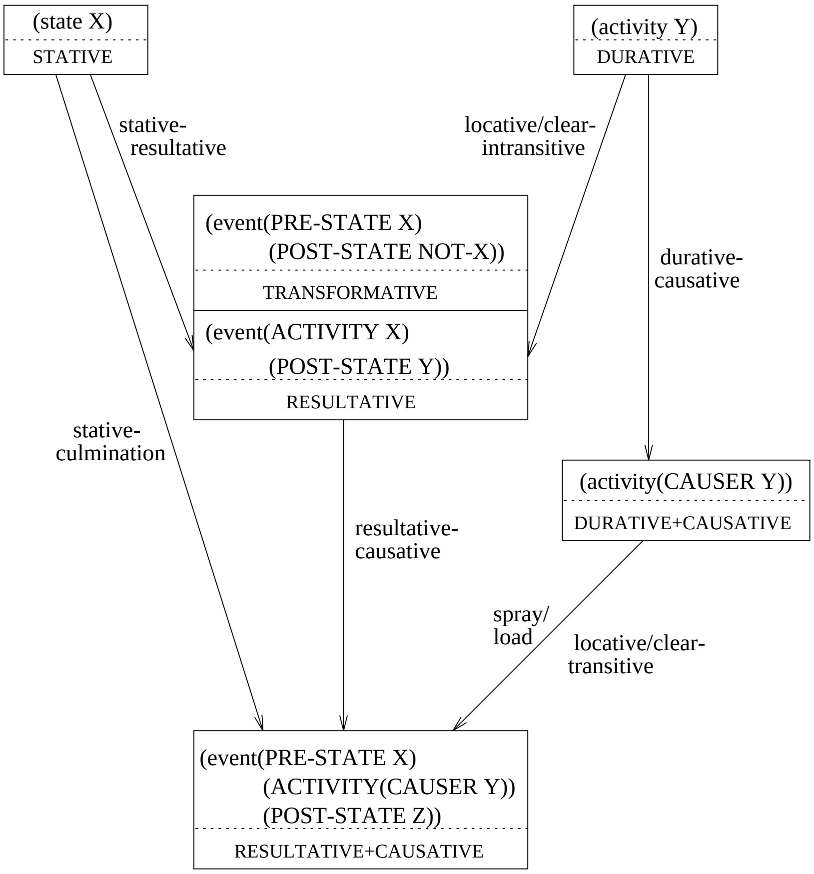

Figure 7.5: Dependency of extension rules

```
Denotation: (activity (OBJECT A)
                      (PATH (SOURCE B)))
PSemSpec: (x1 / nondirected-action  :lex drain_el :actor A :source B)
```

(0) *The water drained from the tank.*

Locative/clear-intransitive of (0):

```
Denotation: (event (ACTIVITY (OBJECT A)
                             (PATH (SOURCE B)))
                   (POST-STATE (C (OBJECT B))))
PSemSpec: (x1 / nondirected-action  :lex drain_el :of-matter A :actor B)
```

(1) *The tank drained of the water.*

Durative–causative of (0):

```
Denotation: (activity (OBJECT A)
                      (PATH (SOURCE B))
                      (CAUSER C))
PSemSpec: (x1 / directed-action  :lex drain_el :actee A :source B :actor C)
```

(2) *Tom drained the water from the tank.*

Resultative–causative of (1):

```
Denotation: (event (ACTIVITY (OBJECT A)
                             (PATH (SOURCE B))
                             (CAUSER C))
                   (POST-STATE (C (OBJECT B))))
PSemSpec: (x1 / directed-action  :lex drain_el :of-matter A :actee B :actor C)
```

(3) *Tom drained the tank of the water.*

Figure 7.6: Derivation of *drain*-configurations by extension rules

verbs, but the incoming arrows produce RESULTATIVE forms. From the entry point of a verb, arcs can be followed if the respective alternation is specified in the lexical entry. At the end of this section, we will give summarizing examples for these. For now, returning to the example of *to drain*, figure 7.6 shows how the rules successively derive the various configurations.

7.3.3 Extension rules for circumstances

We have described the instrument of extension rules for dealing with traditional verb alternations that add new participant roles to a verb. Turning now to the task of associating *circumstances* with the SemSpec (elements that are not part of the case frame of the verb, in other frameworks handled by *adjunct rules*), we can employ the very same rule mechanism, without having to introduce new machinery. We will deal with the circumstance extensions only very briefly and, to continue our examples of spatial movement, give some rules adding PATH elements to MOVE processes.

To go is a verb that requires a DESTINATION as a participant in its case frame; ??*Jill went* and ??*Jill went from school* are marked utterances that work only in very specific situations, whereas *Jill went to Woolworth's* is unproblematic. But optionally, the SOURCE can also be present, as in *Jill went from school to Woolworth's*. For this and many other verbs of movement, a 'path-source' extension checks whether a SOURCE is present in the input SitSpec and if so adds it to the SemSpec. Correspondingly, movement verbs that do not already have the `:destination` in their case frame, can accept it as a circumstance. The two rules that perform the extension are given below.

```
NAM: path-source
DXT: (move (PATH (SOURCE X)))
COV:
ROC:
NRO: (:source X)

NAM: path-destination
DXT: (move (PATH (DESTINATION X)))
COV:
ROC:
NRO: (:destination X)
```

There is, however, an important distinction to be made; the rules given above apply only to those verbs where adding the SOURCE or DESTINATION does not change the Aktionsart. This is true for, amongst others, *to drain*: In *The water drained {into the sink}* and *The tank drained {into the sink}* the additional phrase leaves the Aktionsart unaffected. That is different, for example, with *to move*, where adding a DESTINATION implies a change of LOCATION-STATE and thus a change in Aktionsart: *The cat moved for an hour / The cat moved to the kitchen in an hour*. This, in turn, depends on whether the PATH is BOUNDED—otherwise the cat would only move *toward* the kitchen. For the bounded cases, we need a 'bounded-path-destination' extension that appropriately extends the denotation and the covering-list.

```
NAM: bounded-path-destination
DXT: (event (move (PATH (DESTINATION X)))
            (post-state (Y loc-state (LOCATION X))))
COV: (Y)
ROC:
NRO: (:destination X)
```

Circumstance rules apply to large classes of verbs, and for storing and applying them intelligently, an appropriate structure has to be found for the lexicon—something that we did only for the background knowledge base. But this leads to the more far-reaching point of inter-connecting lexemes in general: Applying a causative extension to *rise* should lead to the verb *raise*, and similarly for many other examples. We have only treated relationships between different forms of the same lexeme here, and we leave the whole problem of organizing the lexicon, including the definition of verb classes, as an issue for future work (see section 10.4).

7.3.4 Examples: lexical entries for verbs

To illustrate our treatment of valency, argument linking, and alternation/extension rules, figure 7.7 shows excerpts from lexical entries of nine different verbs. The information is arranged as follows: On the right-hand side of each entry is the case frame of the verb, written as the

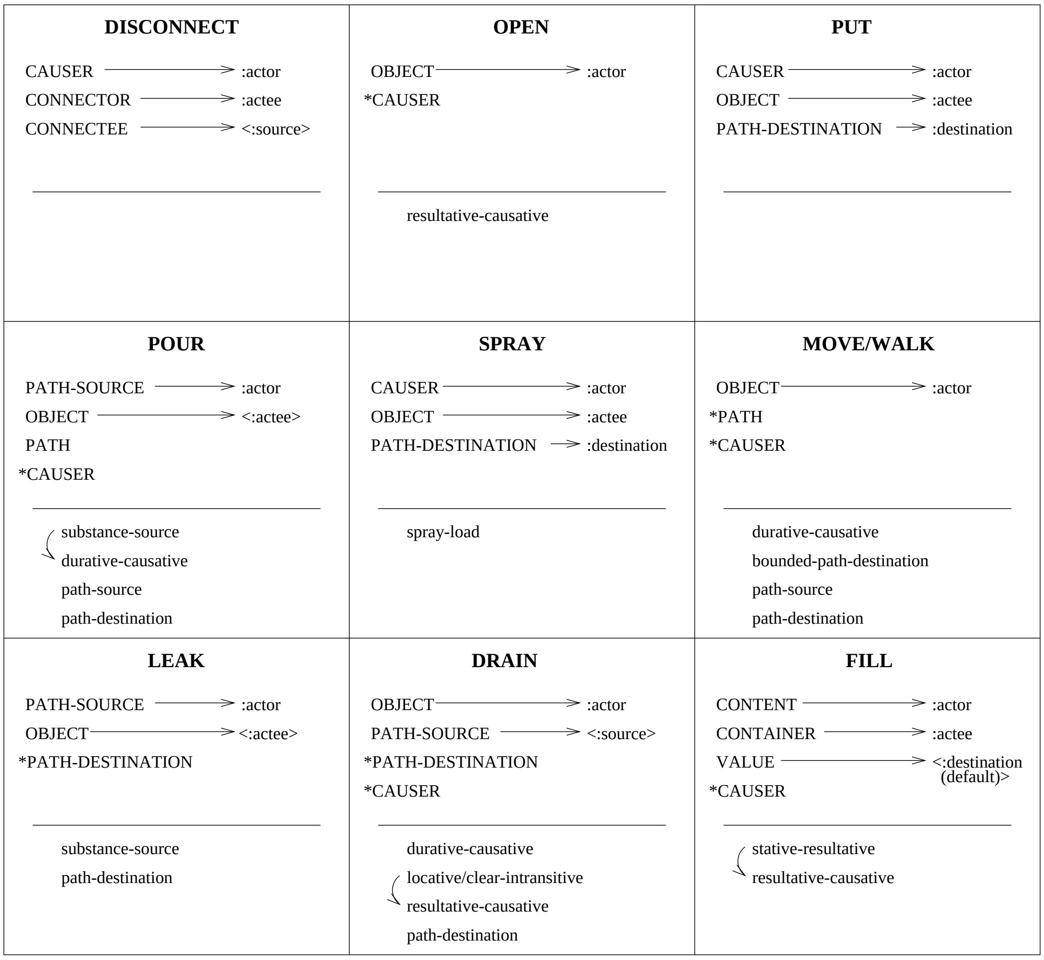

Figure 7.7: Sample lexical entries (abridged) for verbs

SemSpec participant keywords (each starting with a colon). Optional participants are enclosed in angle brackets. On the left-hand side are excerpts from the denotation: the names of the roles whose fillers are co-indexed with the respective position in the case frame. Thus, the arrows give the argument linking for the base form of the verb, which can be quite simple, as in *open* or *move*. From the perspective of the domain model, the roles on the left-hand side of the arrows are required to be filled—as is encoded in the LOOM definitions of the underlying concept. Items appearing with an asterisk in front of them are *optional in the SitSpec*: for example, a SitSpec underlying an OPEN event is well-formed without a CAUSER being present. The optional elements are listed here because they can be verbalized with the extension rules that we have introduced. The names of all the applicable rules (those that we have discussed here) for a verb appear below the line. Arrows indicate the order in which the rules are to be applied—if that order is important for the verb. The extension rules for CIRCUMSTANCES, those that add elements of a PATH, can apply at any time. With *pou¡r*, for example, the destination extension can apply to the base form, *or* to the one derived by the substance–source alternation, *or* to the one derived by the durative–causative extension. The figure illustrates a range of quite different verbs, among them those we have mentioned earlier in our discussions of valency.

7.3.5 Summary

In this section we have discussed verb alternations and proposed viewing them in a 'generative' manner as productive devices that take one form of a verb and derive a more complex form. This is accomplished by rewriting the partial SemSpec, and in the case of extension rules, adding a new subgraph to the denotation, and possibly adding nodes to the covering-list. Therefore, an extension can

- express more pieces of meaning, i.e. have larger coverage,
- add new participant linkings and/or overwrite old ones,
- thereby project the inherent Aktionsart of a verb to that of the sentence, and
- shift emphasis between elements of the sentence (which will be discussed in the next section).

With the help of these rules, it is possible to systematically derive more complex verb configurations from basic ones, and we have shown that the entire alternation space for a verb like *to drain* can be generated (judging from the alternations that Levin lists for the verbs). As a central feature, we apply these rules on the semantic level, so that they are not specific to any of the target languages.

Our main point was to define the rules in terms of Aktionsart features, and then define the order of rule application such that the whole range of the Aktionsart variants that we discuss in this thesis can be produced. Of the range of alternations and extensions discussed by Levin, we have obviously worked only with a few, but several others can be treated in similar ways; for example, the 'conative' alternation would also be treated as an extension deriving a RESULTATIVE reading (*read the paper, cut the meat*) from a DURATIVE one (*read in the paper, cut at the meat*).

In development of the lexical entries for our system, we have in the last three sections defined the following components of word meaning:

- Denotation: partial SitSpec;

- Covering-list: list of nodes from the denotation;
- Partial SemSpec;
- For verbs, pointers to the alternations and extensions they undergo.

With these, it is possible to derive a variety of verbalizations from the same underlying SitSpec, and we will illustrate this with examples in the next chapters. In the remaining two sections of this chapter, though, we turn to factors for *preferring* one verbalization over another, thereby completing our lexical entries.

7.4 Salience

In this and the following section, we discuss two dimensions of difference between similar verbalizations, which provide reasons for preferring one verbalization over another. The purpose of the sections is foremost to demonstrate in general that accounts of dimensions of preference can be integrated into our generation approach; they are not worked out in detail.

In section 3.9 we have discussed the notion of salience and its role in NLG, and in particular examined two contributions from linguistics: Kunze's [1991] theory of 'semantic emphasis' and Talmy's [1988] work on distributing 'attention' in sentences. Now, we will sketch how their theoretical insights can be integrated in our generation framework, which will result in the addition of a `SAL` component to the lexical entries.

The motivations for assigning salience to elements of an ongoing discourse are to be treated on the level of text planning by the strategic generation module that decides "what to say". The sentence generator can then assume its input expressions to be annotated with salience information. In our definition of SitSpecs in chapter 6, we have therefore accounted for the possibility that elements of a situation can have 'foreground', 'background', or 'optional' features associated with them. Such annotations can result from considerations about discourse focus development, from the distinction between given and new information, and so on. For the sentence generator, the source of the annotation is not relevant; this module is only in charge of finding the best possible way to accommodate the additional parameters in its realization decisions.

We have stated the constraint that the ACTIVITY and the POST-STATE node can share at most one 'foreground' label. If a label is present at either of these nodes, it creates a strong preference for choosing a verb that emphasizes this aspect of the event. For the episode of Jill pouring water into a tank, whose SitSpec and several SemSpecs were given in chapter 6 (figures 6.3 and 6.5), this results in either using *to fill* (expressing the POST-STATE) or using *to pour* (expressing the ACTIVITY) with an additional clause expressing the POST-STATE. This difference in emphasis is related to the phenomenon of emphasizing different 'partial propositions', as Kunze [1991] called it; the activity occurring and the state changing collectively characterize the event, and either of them can be emphasized. Note, however, that Kunze, in discussing the transfer-of-possession examples, was concerned with partial propositions that are all part of the state changes. But the notion of emphasis can apply to the ACTIVITY versus POST-STATE distinction as well.

Kunze's example of *give / receive* can be handled in our framework very much as he suggested. The two verbs would share the same denotation, which encodes the selectional restrictions (common to both verbs), but they have different PSemSpecs, and hence differ in linking the elements of the situation to participant roles. Kunze's second aspect of lexical meaning, specialization of the base form by means of subsumption, is clearly in the focus of our approach, as should have become clear. And regarding his third point, the fine-grained features, we have a preliminary proposal on connotation (in the next section), but most of the fine-grained distinctions are—not surprisingly—beyond the abilities of our system, too.

Salience labels at SitSpec nodes other than ACTIVITY and POST-STATE are to be treated differently. Applying the linguistic means given by Talmy [1988] to our level of SemSpec, we have these lexical possibilities to place elements in the foreground:

(1) Do not incorporate the element.

(2) Map the element to ACTOR.

(3) Map the element to ACTEE.

And there are three ways of placing them in the background:

(1) Incorporate the element.

(2) Map it neither to ACTOR nor to ACTEE.

(3) If the element is also marked as 'optional', then do not verbalize it at all.

Consider now the following verbalizations of the fill-situation, which all place the state transition in the foreground by using the verb *to fill*, but differ in their assigning salience to Jill, the tank, and the water.
(a) *Jill filled the tank with water.*
(b) *The tank was filled with water by Jill.*
(c) *The tank was filled with water.*
(d) *The tank filled with water.*
(e) *The tank filled.*
Of these sentences (a) is the standard, unmarked verbalization: It covers every element and does not result from applying any particular alternation. Sentence (b) places the tank in the foreground, or Jill in the background: If either of these annotations was made to the SitSpec, then the passive alternation achieves the desired effect. Version (c) goes a step further and expresses the fact of causation but not the causer. It is the preferred verbalization if Jill were given a 'background' label *and* was marked as 'optional' in the SitSpec. The causation vanishes entirely from the verbalization in (d). In (e), the water is not mentioned, either, which would result from having it also marked with 'background' and 'optional'.

Note that, interestingly, when using the verb *to fill*, we cannot place the water in the foreground and at the same time mention Jill. If this setting of salience is desired, the generator has to switch to emphasizing the ACTIVITY and say *Jill poured water into the tank*, which at least maps the water to the ACTEE role. For a stronger effect, the generator can now use the passive alternation: *Water was poured into the tank by Jill.* In this way, the emphasis on an aspect of the event results from the foregrounding and backgrounding of elements, not from attaching the 'foreground' label at either the ACTIVITY or the POST-STATE. Thus, the verb choice can be influenced either directly or indirectly.

In German, on the other hand, *füllen* can undergo the locative alternation, and therefore, given the just-mentioned salience settings, we do not have to switch to a different verb: *Jill füllte Wasser in den Tank.*

We will not work out the salience assignment procedures here, because, as we have stressed in the introduction, a thorough treatment of the topic is a rather complex endeavour. However, here is a sketch of how one can proceed in our framework. Given the means of realizing salience on the level of PSemSpec listed above, any verb lexical entry (i.e., its basic form) can be evaluated for the salience assignments it realizes. That is, for any verbalization option *vo* we can determine how well it performs in expressing the 'foreground' and 'background' labels attached to the SitSpec nodes that the *vo* covers. Thus, we have a measure for preferring one *vo* over another on the grounds of salience. Then, the alternations and extensions introduced in the last section can alter the salience assignment systematically. This information needs to be added as a sixth component to the alternation/extension rules. For instance, the passive places the former ACTOR rather in the background, and the former ACTEE in the foreground. The salience-modification of the other rules can be stated similarly; if a rule shifts a former ACTEE to a less prominent role (as the locative extensions do), then that former ACTEE is effectively backgrounded. Hence, the rules would in addition to rewriting denotations and PSemSpecs also alter the salience assignment of the *vo*. As a result, all the different *vo*s can at the end be compared with respect to their salience behavior, and the preferred one be chosen, so that, for instance, one of the variants of the fill-verbalizations shown above can be selected.

In the lexical entries, an additional `SAL` field would contain two lists of nodes, each a subset of the covering-list of the lexeme, that state which items are foregrounded and backgrounded, respectively, and the alternation/extension rules would alter these lists in the verbalization options. In chapter 9, we will illustrate the influence of the salience parameters on lexical choices with further examples.

It is to be noted that there are obviously other, non-lexical means of assigning salience: thematizing a constituent makes the element very prominent, and using a cleft-sentence makes this effect even stronger:
To Mary he gave the book.
It was Mary he gave the book to.
With examples of this kind, salience really becomes a matter of degree and is difficult to evaluate. But, in principle, these ways of making elements prominent are available in our system, too. In the event that lexical means did not succeed in achieving the degree of prominence that is annotated at the SitSpec node corresponding to MARY, the system can realize the desired effect by adding a `:theme` keyword to the SemSpec, followed by the variable corresponding to the participant or circumstance that represents MARY. PENMAN will then accordingly thematize the constituent.

7.5 Connotation

When discussing the opposition of denotation and connotation in section 3.4, we likened connotational lexical features to *stylistic* ones. The notion of style is most commonly associated with literary theory, but that perspective is not suitable for our purposes here. Style has also been investigated from a linguistic perspective (e.g., Sanders [1973]), and a computational treatment has been proposed by DiMarco and Hirst [1993]. What, then, is *style*? Like Sanders, we view it broadly as the choice between the various ways of expressing the same message. Linguists interested in style, for instance Crystal and Davy [1969], have analyzed the relationships between situational parameters (in particular, different genres) and stylistic choice, and work in artificial intelligence has added the important aspect of (indirectly) linking linguistic choices to the intentions of a speaker, as done by Hovy [1988a].

Hovy also proposed a number of stylistic dimensions along which words can differ, and we have, in some preliminary work reported by Stede [1993], refined this set by evaluating a number of dictionaries and guidebooks for "good writing". Stylistic features can also be isolated by carefully comparing words within a set of near-synonyms, from which a generator is supposed to make a lexical choice: When words within one lexical field can be differentiated along some dimension, and the same dimension comes back in other lexical fields, then there is a candidate to be admitted into the set of features. For each feature, the words in the lexical field can then be rated on a corresponding numerical scale; our initial experiments so far have shown that a range from 0 to 3 is sufficient to represent the differences. Several features, however, have an 'opposite end' and a neutral position in the middle; for them, the scale is -3 to 3.

Rating words is best done by constructing a "minimal" context for a paradigm of synonyms so that the semantic influence exerted by the surrounding words is as small as possible (e.g.: *They destroyed/annihilated/ruined/razed/... the building*). Words can hardly be compared with no context at all—when informants are asked to rate words on a particular scale, they typically demand a sentence surrounding the word immediately. If, on the other hand, the context is too specific, i.e., semantically loaded, it becomes more difficult to get access to the inherent features of the particular word in question.

Following are the stylistic features that have been determined by investigating various dictionaries and guides on good writing and by analyzing a dozen synonym-sets that were compiled from thesauri:[9]

- FORMALITY: $-3 \ldots 3$
  This is the only stylistic dimension that linguists have thoroughly investigated and that is well-known to dictionary users. Words can be rated on a scale from 'vulgar' via 'colloquial' to 'very formal' or something similar (e.g., *flick / movie / motion picture*).

- EUPHEMISM: $0 \ldots 3$
  The euphemism is used in order to avoid the "real" word in certain social situations. Euphemisms are frequently found when the topic is strongly emotional (death, for example) or related to a social taboo (in a *washroom*[10], the indicated activity is merely a secondary function of the installation).

- SLANT: $-3 \ldots 3$
  A speaker can convey a high or low opinion on the subject by using a slanted word: a favorable or a pejorative one. Often this involves metaphor: a word is used that in fact denotes a different concept, for example when an unfavorable person is called a *rat*. But the distinction can also be found within sets of synonyms, e.g., *skinny / slim*.

- ARCHAIC ... TRENDY: $-3 \ldots 3$
  The archaic word is sometimes called 'obsolete', but it is not: old words can be exhumed on purpose to achieve specific effects, for example by calling the pharmacist *apothecary*. This stylistic dimension holds not only for content words: *albeit* is the archaic variant of *even though*. At the opposite end is the trendy word that has only recently been coined to denote some modern concept or to replace an existent word that is worn out.

- FLORIDITY: $-3 \ldots 3$
  This is one of the dimensions suggested by Hovy [1988a]. A more flowery expression for *consider* is *entertain the thought*. At the opposite end of the scale is the *trite* word. Floridity is occasionally identified with high formality, but the two should be distinguished: The

[9] DiMarco, Hirst, and Stede [1993] discuss a subset of these features and add 'emphasis' as a connotative dimension, similar to what we have discussed in the last section, but on a sub-lexical level.

[10] This is the Canadian word for what in American English is, equally euphemistic, a *restroom*.

flowery word is used when the speaker wants to sound impressively 'bookish', whereas the formal word is "very correct". Thus, the trite *house* can be called *habitation* to add sophistication, but that would not be merely 'formal'. Another reason for keeping the two distinct is the opposite end of the scale: a non-flowery word is not the same as a slang term.

- ABSTRACTNESS: $-3\ldots3$
  Writing guidebooks often recommend replacing the abstract with the concrete word that evokes a more vivid mental image in the hearer. An example is to characterize an *unemployed* person (abstract) as *out of work* (concrete). The recommendations found in the literature typically suggest to use *semantically more specific* words (e.g., replace *to fly* with *to float* or *to glide*), which add traits of meaning and are therefore not always interchangeable; thus in such cases the choice is in fact not merely stylistic.

- FORCE: $0\ldots3$
  Some words are more forceful, or 'stronger' than others, for instance *destroy* / *annihilate*, or *big* / *monstrous*.

There is an interesting relationship (that should be investigated more thoroughly) between these features and the notion of *core vocabulary*, as it is known in applied linguistics. Carter [1987] characterizes core words as having the following properties: they often have clear antonyms (*big—small*); they have a wide collocational range (*fat cheque, fat salary* but *∗corpulent cheque, ∗chubby salary*); they often serve to define other words in the same lexical set (*to smile happily = to beam, to smile knowingly = to smirk*); they do not indicate the genre of discourse to which they belong; they do not carry marked connotations or associations. This last criterion, the connotational neutrality of core words could be measured using our stylistic features, with the hypothesis being that core words tend to assume the value 0 on the scales. However, the coreness of a word is not *only* a matter of style, but also of semantic specificity: Carter notes that they are often superordinates, and this is also the reason for their role in defining similar words, which are, of course, semantically more specific. It seems that the notion of core words corresponds with that of *basic-level categories*, which Reiter [1990] employed in NLG, but which originated not in linguistics but in cognitive psychology [Rosch 1978].

Given a lexicon in which words are ranked with respect to the aforementioned features, and a target specification of the desired 'color' for verbalizing a SitSpec element, the preferred word can be determined with a distance metric. For example, one can add for every word the squares of the differences between the target feature value (*tf*) and the value found in the lexical entry (*wf*) for each of the n features: $\sum_{i=1}^{n} (tf_i - wf_i)^2$

The fine-tuning of the distance-metric is subject to experimentation; in the version shown, the motivation for taking the square of the difference is to, for example, favor a word that differs in two dimensions by one point over another one that differs in one dimension by two points (they would otherwise be equivalent). The word with the lowest total difference is chosen; in case of conflict, a random choice is made.

Although the features suggested above resulted from only preliminary work, the main point is that stylistic choice dimensions do exist between similar words, and that they can be accounted for in language generation. In addition to the salience distinctions explored in the last section, stylistic features represent a set of criteria for preferring a word from a set of near-synonyms; in our framework, they are a factor for making choices from the pool of verbalization options. Therefore, our dictionary entries are extended by a final component: a list of connotational features and values.

In the sample domain we are using for the generator developed in this thesis, stylistic distinctions play a rather small role. However, some variation is possible. Consider:
Tom drew the water off the container. (somewhat colloquial)
Tom drained the water from the container.
Tom discharged the water from the tank. (highfalutin)
We represent these words in our system as follows: *to drain* has been discussed earlier and does not have any particular connotational features—it is a neutral word. *To discharge* behaves similar to *to drain*, but receives positive values on the connotation scales for 'floridity' and 'formality'. *To draw off*, on the other hand, receives a negative value on the 'formality' scale. The `CON` components of the lexical entries are as shown below. When our system now verbalizes a SitSpec representing a draining situation, it can make a choice among these verbs according to the connotational dimensions, if the target specification includes a setting for 'formality' or 'floridity'.

```
to drain
CON: ()

to discharge
CON: ((floridity +2) (formality +2))

to draw off
CON: ((formality -1))
```

7.6 Summary: lexicalization with constraints and preferences

Lexical entries Let us now summarize what we take to be the components of word meaning in our system. We have introduced the *denotation* and *covering-list* in section 7.1, and pointed out that there are other aspects of denotational meaning, certain fine-grained semantic traits, that our approach cannot account for. The *partial SemSpec* was the topic of 7.2. It includes a pointer to a set of *morphosyntactic features* needed by the surface generator. We have not elaborated on them here; they characterize the inflectional behavior of the word and its syntactic role in the sentence. In section 7.3 we introduced *alternation and extension rules* that can apply to verbs, and their lexical entries thus contain pointers to such rules. The final two sections dealt with preferring verbalizations on the grounds of *salience* distribution and *connotational features*. Thus, we define lexical entries as an 8-tuple with the following components:

`NAM`: name,
`LAN`: language: `E, G,` or `E/G`,
`DEN`: denotation: a SitSpec template,
`COV`: covering-list: a subset of nodes in `DEN`,
`PSS`: Partial SemSpec,
`SAL`: a list of two lists with subsets of nodes in `DEN`,
`CON`: a list of stylistic features and values,
`AER`: for verbs only: names of alternation and extension rules.
In the `PSS`, there is a `:lex` field whose filler is the name of a morphosyntactic entry. Its suffix `_E` or `_G` indicates the language.

Advantages These comprehensive specifications, which cleanly separate the different realms of lexical information, are in contrast to the lexical entries that previous generation systems used, where little more than a denotation and morphosyntactic features were provided.

By means of sharing variables between denotation template and partial SemSpec, the lexicon entries serve as a "bridge" between the input to the generator, a SitSpec, and the intermediate representation SemSpec, which is given to a front-end sentence generator. Since the 'ontological categorizations' in SitSpec and SemSpec differ, the combination of lexemes chosen can thus produce a structure different from that of the SitSpec. This restructuring can occur when it is appropriate for verbalization, and in language-specific ways. One important consequence is that SitSpec actant roles are not the same as SemSpec participant roles, i.e., one need not commit oneself to the well-established linguistic 'deep cases' at the SitSpec level. For example, a FILL-STATE in the domain model has the roles OBJECT, SUBSTANCE, VALUE, which can be mapped to various SemSpec process–participant configurations and accordingly verbalized as, for instance, *The tank was full to the second mark with oil* or *Oil filled the tank to the second mark*. For more discussion of the "lexical bridge" conception, see [Stede and Grote 1995].

As an important result of separating denotation from PSemSpec and using the Upper Model as a tool for abstracting over syntax, we were able to state productive rules for deriving more complex verb configurations from simpler ones. We implemented some verb alternations, as they are investigated in linguistics, by treating them as functions that *add* meaning to the simpler configuration—solely by rewriting the denotation and the PSemSpec in tandem. Since the rules operate on the level of SemSpec, they are not specific to a target language and apply to English and German verbs alike.

Also from the perspective of multilinguality, an important feature of the system is the possibility of sharing parts of lexical entries among words belonging to different languages. The 'best' case from this viewpoint is when an English word is synonymous in meaning to a German one, and they differ only in their morphosyntax. The morphosyntactic features, though, are stored in a distinct object, used only by the surface generator (PENMAN). Therefore, the `LAN` field of an entry can be filled by `E/G` if the entry applies to both languages, and the `:lex` component in the `PSS` would include both the English and the German pointer, and the generator will use the right one depending on the target language selected. Example:

```
NAM: upward
LAN: E/G
DEN: (path (DIRECTION 'upward))
COV: ('upward)
PSS: (x / nonscalable-quality :lex (upward_el aufwaerts_gl))
```

Correspondingly, words of the same language can share parts of lexical entries: Synonyms would have the identical denotation and differ only in terms of the `CON` features. Verbs can often share the denotation and differ only in terms of the `PSS`, the `AER` pointers, and the `SAL` assignment. This implements and extends Kunze's [1991] findings that verbs can share the same 'base form' and differ in terms of their emphasizing different aspects.

Constraints and preferences When applying the components of word meaning to the realm of generation, lexicalization turns out to be a matter of *constraints* and *preferences*. On the one hand, the denotation or applicability condition of a lexeme has to be exactly right for it to be usable; on the other hand, there are dimensions along which variation is tolerable, and deciding on a particular word is a matter of relative preference.

Phrased in terms of *communicative goals*, it is clear that the goal of referring to the right object must be achieved—and referring is not a matter of degree, but a yes/no question.[11]

[11] Whether the referring expression is *understood* correctly by the hearer is a different matter [Heeman and

Whenever there is more than one lexeme with the right denotation at hand, other goals can come in to direct the choice. These goals, however, are of a different kind: they can be fulfilled partially, and they concern the connotations of words, or the assignment of salience. If one wants to sound flowery, or highfalutin, in order to impress the audience, then there may be more or less flowery words for the things one talks about; for some things, however, there are none—a *laser printer* is just a laser printer, so to speak. The same holds for goals like "talk formally" or "talk in a vulgar way": when there is a choice of words, one can try to pick a maximally formal or vulgar one, but many times there will just be none available.[12] In these cases, one has to resort to the neutral word: the one that does not exhibit the desired 'color', but at least not any other, unwanted, color, either. Knowing which these words are is important for language production as it minimizes the chance of inadvertently implying things not intended.

Similarly, achieving effects of salience distribution is a goal that can be fulfilled to various degrees; we have listed in section 7.4 a number of means for making elements more or less prominent, and these again interact with other decisions. Moreover, the goals that influence text production can be conflicting at times. When a text is supposed to be brief and concise on the one hand (say, in order to fit onto a small page), and to sound overly bookish on the other (say, in order to amuse the reader), then the choice between *Mr. Miller entertained the possibility of vacating the premises* and *Mr. Miller considered leaving the house* depends on which goal is to be favored here.

Thus, communicative goals other than saying the "right" thing (that is picking the correct denotation) amount to preferences: They are tendencies for steering generation decisions into particular directions; they can succeed to different degrees in different situations, and they can conflict with one another, in which case an order of importance is to be established.

This observation on the distinction between constraints and preferences motivates the basic architecture of the generator, which will be explained in the next chapter.

Paraphrases To conclude this chapter, we return to the notion of 'paraphrase', which was discussed at length in chapter 4. There, we gave many examples of the phenomenon but also stressed the point that it made little sense to provide a precise definition of the term. Meaning equivalence, so went the argument, can only be judged in the context of a specific framework that explains *meaning*. Now, we have a framework in place, and paraphrases can be characterized as sentences that are derived from the same SitSpec but possibly result from different SemSpecs; these SemSpecs, however, must cover the same set of SitSpec elements.

Hirst 1995]; the point is that a speaker's intention is to refer to one element "in total", and not just to some degree.

[12]But, on the other hand, it is always possible to insert a modifier that creates the vulgarity.

Chapter 8

A new system architecture for multilingual generation

We have developed many declarative representations in the preceding chapters and will now introduce the generation *procedure* that uses all the information and derives verbalizations from SitSpecs. After discussing the overall computational problem, we give a brief overview of the generation architecture and then discuss the individual steps in more detail. Then, we describe 'MOOSE', an implementation of a prototype realizing the architecture, and explain how it could be integrated into a larger generation system.

8.1 The computational problem

Abstracting now from meaning, natural language, and represented knowledge, the computational problem of the generation task is to be specified. As already explained in chapter 6, we delegate surface generation, i.e., the mapping from a SemSpec to a sentence, to PENMAN, a separate module that operates as a 'black box.' That part of the problem will thus not concern us here.

For the remaining work, a line is to be drawn between determining the pool of possible verbalization options on the one hand, and constructing from this pool a SemSpec that verbalizes the SitSpec on the other. This division is useful because we have emphasized the important role of 'packing' the elements of the SitSpec into lexemes in different ways. The pool of verbalization options implicitly offers a potentially large number of possibilities for distributing meaning units to words—and in order to make a choice on the basis of various parameters, it is important to compare all the options as they stand. Intertwining the search for lexeme candidates with SemSpec construction would make it difficult to evaluate different alternatives of distribution; besides, the problem would be enormously complex. Thus, the overall task falls into three sequential steps:

(1) Matching: Find the verbalization options whose denotation subsumes some part of the SitSpec.

(2) SemSpec construction: Find a well-formed and preferred SemSpec covering the SitSpec.

(3) Surface generation: Map SemSpec to a sentence in natural language.

Although the first step is, obviously, crucial for the performance of the system, we do not elaborate it here, since it is a standard matching task that has to be tackled with intelligent indexing techniques. What exactly the matching has to do will be explained in section 8.2.1.

The central problem to be analyzed here is that of SemSpec construction. The input to this step consists of:

- a SitSpec, possibly annotated with salience information,
- a set of verbalization options *VO*,
- a specification of target stylistic features.

The goal is to find a subset of the verbalization options whose elements can be combined into a well-formed SemSpec. The *vo*s must collectively cover the set of SitSpec nodes, and they are to be optimal with respect to the stylistic and salience parameters.

To formalize these conditions, we first note that the Upper Model defines the well-formedness conditions for SemSpecs, i.e., it specifies in what ways PSemSpecs can be combined into a well-formed and saturated SemSpec. In effect, it defines the "combinable" PSemSpecs as a subset of the power set of the verbalization options: $VO_c \subseteq 2^{VO}$. Further, the covering information associated with verbalization options can be seen as a relation between VO and the power set of the SitSpec nodes. For the preferential factors, salience and style, we here assume a single specification of the target values that the generator tries to come close to. A distance function can then compute the difference between an actual set of verbalization options participating in a SemSpec and the target specification. Thus, given

- a set of SitSpec nodes $SN = \{sn_1, \ldots, sn_n\}$
- a set of verbalization options $VO = \{vo_1, \ldots, vo_m\}$
- a cover-relation mapping verbalization options to sets of SitSpec nodes:
  C: $VO \rightarrow 2^{SN}$,
- a specification of target stylistic and salience features *tf*,
- a distance function for evaluating stylistic and salience features D: $VO \times tf \rightarrow N$,

we have to select from the verbalization options VO a subset $VO' = \{vo'_1, \ldots, vo'_o\}$ with the following properties:

(1) VO' yields a well-formed and saturated SemSpec:
$VO' \in VO_c$

(2) VO' completely covers SN:
$\forall sn_i \in SN \;\; \exists vo' \in VO' \; [sn_i \in C(vo')]$

(3) There is no overlap in coverage:
$\forall sn_i \in SN \; \neg\exists vo'_j, vo'_k \in VO' \; [sn_i \in C(vo'_j) \wedge sn_i \in C(vo'_k) \wedge vo'_j \neq vo'_k]$

(4) VO' is minimal under D:
$\neg\exists \; VO'' = \{vo''_1, \ldots, vo''_k\} \subseteq VO \; [\; \sum_{i=1}^{k} D(vo''_i, tf) < \sum_{i=1}^{o} D(vo'_i, tf) \;]$
where VO'' also fulfills the other conditions.

In order to convert this hard problem to a manageable search task, we make use of the fact that the preferential factors we are using here can all be computed locally for each individual element of VO. By doing this and loosening the requirement to find the *overall* preferred solution, we can devise a search algorithm that considers *vo*s at every node in their preferred order. The algorithm tries to cover a SitSpec node first with the *vo* that has the best value under D, and only when backtracking becomes necessary, does it consider the next best *vo*. The search procedure will be explained in section 8.2.4.

8.2 Overview of the architecture

The various modules of the generator and the data flow are shown in figure 8.1. Solid boxes depict knowledge resources or data, and boxes with round edges stand for processes; they are numbered for cross-referencing with the following description. The SitSpec to be verbalized, shown in the upper left corner, is an instantiation of domain model concepts. In a matching phase with the denotations of lexical entries (1), the applicable lexemes are determined; they constitute the pool of verbalization options *VO*. In the next step, alternation and extension rules, triggered by the verbs in the pool, are employed to derive further verbalization options and thus enlarge *VO* (2). Next, for each individual node in the SitSpec, the lexemes that can cover it are brought into an order of preference (3), on the basis of an evaluation function that inspects the salience and stylistic features associated with the verbalization option and compares it to a target specification of these values. Then, the central task is building a language-specific SemSpec from the verbalization options (4); its well-formedness is ensured by the Upper Model, and a preferred option is found by considering the various parts in their order of preference. Finally, the SemSpec is given to the English or German surface generator, which maps it to a natural language sentence (5).

We will now explain each step in detail, and illustrate the procedure by continuing with the example of Jill filling a tank with water, which was used in chapters 6 and 7. The SitSpec was given in figure 6.3, and its textual version is repeated below; some SemSpecs were shown in figure 6.5. We will in the following section explain the production of the (relatively simple) sentence *Jill filled the tank with water*, and discuss the generation of the other sentences given in figure 6.5 in the next chapter.

8.2.1 Find lexical options

If language generation is based on a rich lexicon, offering an array of synonymous or nearly synonymous lexical items for expressing a certain concept, lexical decisions will interact not only with one another, but also with many other decisions to be made by the system. In order to be able to account for as many of these inter-dependencies as possible, we take the first step in the generation process to be the determination of *verbalization options*: the set of all words or phrases that can express some part of the proposition to be expressed.

Technically, determining the set of lexical options amounts to finding those lexemes whose denotation subsumes some part of the SitSpec. That is, for every node I in the SitSpec, we want to find all lexical items whose denotation is in a SUBSUME relationship to the subgraph rooted in I.[1]

The first condition for SUBSUME is that the type of the root node of the denotation is more general than, or the same as, the type of I's root. Next, the denotation root must not have a role associated to it that I does not have; otherwise, the lexical item would imply more than is warranted by the proposition. But conversely, I may very well have roles that are not defined for a lexical item, yet the item may be appropriate; in this case, the item is more general, i.e., it conveys *less* than warranted by the proposition—which, for some reason or another, might be desired. Finally, since we have to match paths of (in principle) arbitrary length, SUBSUME must also hold recursively between role fillers. For all of I's roles, if they are also defined for the candidate denotation, then SUBSUME most hold between the role filler of the denotation and the role filler of I.

[1]We are dealing here with a very simple form of subsumption that only involves checking the positions of nodes in the concept taxonomy.

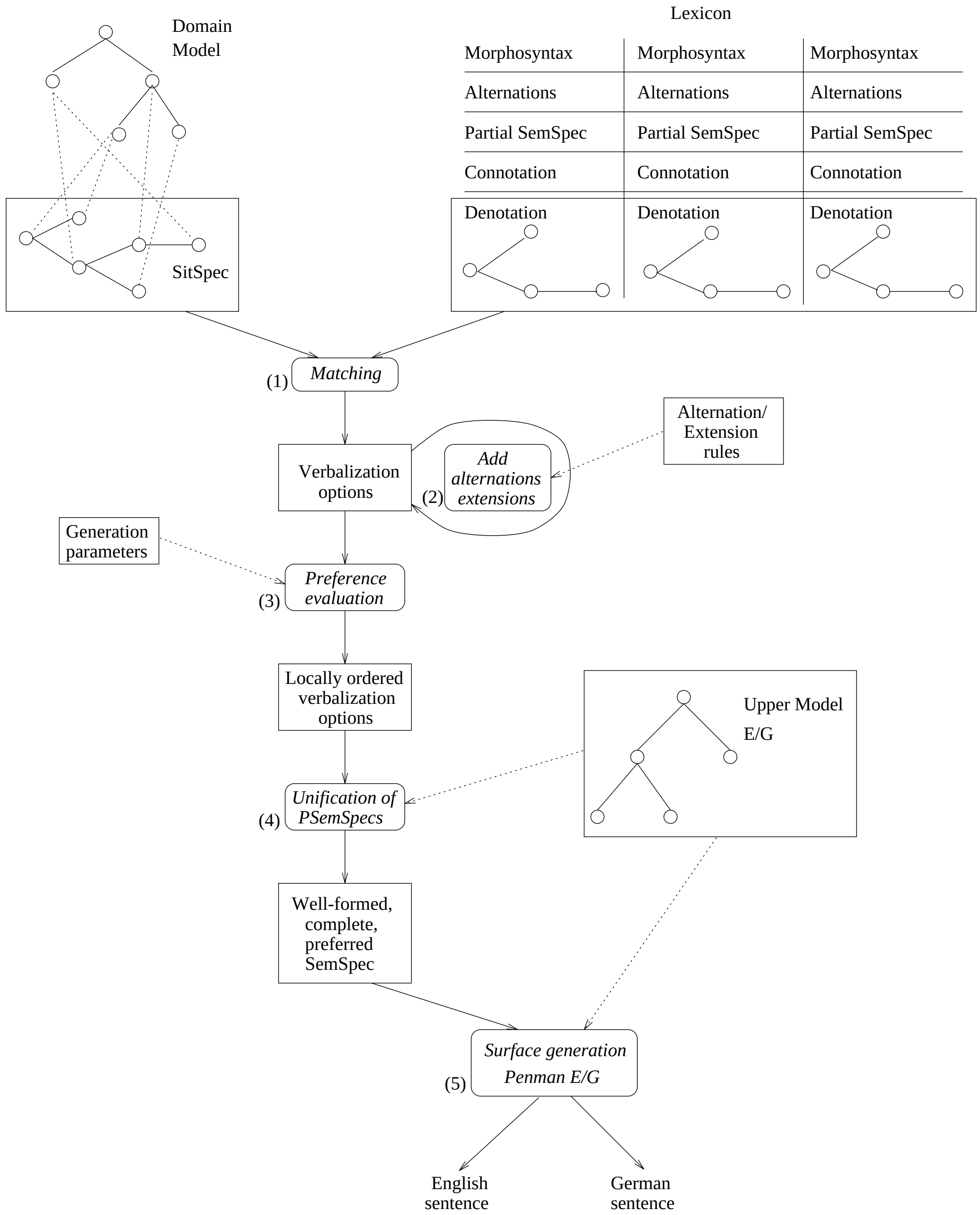

Figure 8.1: Overall system architecture

More formally, the function SUBSUME can be described as follows. Let I denote the SitSpec node under consideration, and $t(I)$ a function that returns the type of I. $C_1 \succeq C_2$ denotes the subtype relationship in the domain model, i.e., C_1 is more general than C_2; $R(i_1, i_2)$ means that relation R holds between two nodes (in other terminology, i_2 fills role R of i_1). Then we are looking for the set of lexemes with denotations i such that:

$$subsume(i, I) \Leftrightarrow [\quad t(i) \succeq t(I) \tag{8.1}$$

$$\wedge \forall R \; \forall x \; [R(i,x) \rightarrow \exists y \; [R(I,y) \wedge subsume(x,y)]] \quad] \tag{8.2}$$

The actual matching procedure is slightly more elaborate, because the syntax of denotations and SitSpecs is not exactly the same. The procedure has to account for variables in the denotation and bind them to the SitSpec nodes they match; if there is a type restriction associated with a variable, it has to be ensured that the SitSpec node is in fact subsumed by that type. And finally, if the denotation contains a *default*, then the role need not be present in the SitSpec for the matching to succeed—but if it is present, subsumption must hold, as for any other role filler.

The matching procedure is executed for every node of the SitSpec, in order to determine the lexemes that can potentially cover that node. But instead of blindly searching the entire lexicon at every SitSpec node, only those entries are tested whose denotation root node has either the same type as the SitSpec node or a more general one. In the implementation, this indexing is performed by exploiting the functionality of LOOM; see the description in section 8.3. Also, all subtype checks mentioned above are performed by LOOM.

If a lexical entry *LE* matches a SitSpec node, it becomes *instantiated* and is added to the pool of verbalization options *VO*. Instantiation means:

- The names of nodes in the denotation and covering-list of *LE* are replaced by the names of the SitSpec nodes they match.
- Variables in the denotation of *LE* are bound to the SitSpec nodes they match. This binding is propagated to the same variable in the PSemSpec of *LE*.
- Any type constraints in the denotation of *LE* are removed—future matchings do not have to re-check them.
- A default marker `< >` in the denotation and the covering-list of *LE* is removed if the default value matches that of the SitSpec, and the corresponding term in the PSemSpec is also removed (because it need not be expressed). If the default value does not match the SitSpec, the optionality markers in the PSemSpec are removed, because in this case the information is not incorporated in the lexeme but needs to be expressed separately (as explained in section 7.1.1).
- Simultaneously, a backward pointer is established from the SitSpec node to the *vo*: Each SitSpec node has a 'covered-by' list associated with it, and the name of the *vo* just formed is added to this list.

Thus, after the matching phase, the exact covering relationships between *VO* and SitSpec are recorded on both sides. The 'covered-by' lists associated with the SitSpec nodes will later be used to drive the SemSpec construction.

Example In our Jill-filled-the-tank example, the matching phase finds, *inter alia*, the lexical entries shown in the upper portion of figure 8.2 (for brevity, we list only the denotation,

to fill (stative) DEN: (fill-state (CONTENT A) (CONTAINER B) (VALUE < C ’full >)) PSS: (x / directed-action :lex fill_el :actor A :actee B < :destination C >) AER: ((stative-resultative resultative-causative)) COV: (fill-state < ’full >)	jill DEN: (jill) PSS: (x / person :name jill) COV: (jill)	water DEN: (water) PSS: (x / substance :lex water_el) COV: (water)	tank DEN: (tank) PSS: (x / object :lex tank_el) COV: (tank)
fill (stative) DEN: (fill-state-2 (CONTENT A) (CONTAINER B) (VALUE ’full)) PSS: (x / directed-action :lex fill_el :actor A :actee B) COV: (fill-state-2 ’full)	jill DEN: (jill-1) PSS: (x / person :name jill) COV: (jill-1)	water DEN: (water-1) PSS: (x / substance :lex water_el) COV: (water-1)	tank DEN: (tank-1) PSS: (x / object :lex tank_el) COV: (tank-1)

Figure 8.2: Lexicon entries matching the SitSpec in *fill*–example, and their instantiations

the PSemSpecs, the pointers to alternations/extensions, and the covering-lists). As can be seen when comparing it to the SitSpec (repeated below), the denotations match the respective SitSpec nodes. The resulting instantiations are shown in the lower portion of figure 8.2. Note that the default marker is removed from the denotation of `fill`, and correspondingly the `:destination` term from the PSemSpec, because the value `’full` in the lexical entry is identical to that in the PSemSpec.

```
(event-1 (PRE-STATE  (fill-state-1 (VALUE ’not-full)
                                    (CONTAINER tank-1)))
         (ACTIVITY   (pour-1 (CAUSER jill-1)
                             (OBJECT water-1)
                             (PATH (path-1 (DESTINATION tank-1)))))
         (POST-STATE (fill-state-2 (CONTENT water-1)
                                    (CONTAINER tank-1)
                                    (VALUE ’full))))
```

8.2.2 Construct alternations and extensions

The lexical entries of verb base forms include pointers to alternation and extension rules that can potentially apply to the verb. The second step of the generation procedure examines the verb entries in the pool of verbalization options and tries all its associated rules for their applicability. We have explained this already when introducing the rules in section 7.3; here is a summary of the procedure.

A verb entry can trigger several rules, and the order of application may be critical; if the list of rules associated with the verb contains sublists, then the rules therein have to be executed sequentially. General extension rules that correspond to CIRCUMSTANCES (we have dealt only with PATH-EXTENSIONS) apply to each intermediate result of such a sequence.

An alternation or extension rule consists of the components `NAM` (name), `DXT` (extension of denotation), `COV` (additions to covering-list), `ROC` (role changes in the PSemSpec), and `NRO` (new roles for the PSemSpec). Applying it to a *vo* works as follows:

(1) Make a copy of the *vo* and assign it a new name.

(2) Match the denotation path in the `DXT` field to the SitSpec. If the match fails, stop.

```
NAM: stative-resultative
DXT: (event (Y (ACTIVITY X)))
ROC: ((:actor :inclusive)
      (:actee :actor)
      (directed-action nondirected-action))
NRO: ()
COV: (X Y)
```

```
NAM: resultative-causative
DXT: (event (ACTIVITY (X (CAUSER Y))))
ROC: ((:actor :actee)
      (nondirected-action directed-action))
NRO: (:actor Y)
COV: ()
```

```
fill2 (stative-resultative of fill)
DEN: (event-1 (ACTIVITY pour-1)
              (POST-STATE
              (fill-state-2 (CONTENT A)
                            (CONTAINER B)
                            (VALUE 'full))))
PSS: (x / nondirected-action :lex fill_el
        :inclusive A
        :actor B)
COV: (fill-state-2 'full pour-1 event-1)
```

```
fill3 (resultative-causative of fill2)
DEN: (event-1 (ACTIVITY (CAUSER Y)
              (POST-STATE
               (fill-state-2 (CONTENT A)
                             (CONTAINER B)
                             (VALUE 'full))))
PSS: (x / directed-action :lex fill_el
        :inclusive A
        :actee B
        :actor Y)
COV: (fill-state-2 'full pour-1 event-1)
```

Figure 8.3: Extension rules for *fill*–example, and resulting *vo*s

Otherwise proceed.

(3) Add the path in the DXT field to the denotation of the new *vo*.

(4) Proceeding from left to right, execute the role name changes in the ROC list, i.e., replace old names with new names.

(5) Add the items on the NRO list to the PSemSpec.

At the end of this step, all possible lexical options for verbalizing some parts of the SitSpec have been determined.

Example To continue our example, figure 8.3 shows in the upper portion the two extension rules that apply to *to fill* (repeated from section 7.3). In the lower portion, the two *vo*s resulting from applying these rules are shown.

8.2.3 Establish preference ranking of options

So far, the lexical options on the covered-by list of a SitSpec node are in no particular order. The next step is to induce a local preference ranking over the options on the basis of target stylistic parameters and the salience annotations in the SitSpec.

As pointed out in the beginning of the thesis, the interaction of various choice factors is a wide-open question and not discussed here: We make no proposals about the relative weight of the factors and how these would be combined most appropriately. And accordingly, the precise computation of preferences is not our concern here.

As for stylistic features, the preferences can be computed with a distance function that compares the features associated with the *vo* with the target set of features that have been set for the utterance. In section 7.5 we gave the example of a function that takes the sum of the squares of the differences between the target feature value (tf) and the value in the verbalization option (wf) for each of the n stylistic features: $\sum_{i=1}^{n} (tf_i - wf_i)^2$

Regarding the salience assignment, we need to distinguish between the foreground labels on either the ACTIVITY or the RESULT of an event, and the foreground or background labels on any other node. The former typically result in quite different verbs to express the event; these labels should give rise to a very strong preference for verbs that denote the aspect marked as foregrounded. As for salience labels on other elements, we have listed in section 7.4 different means of foregrounding and backgrounding elements of the SitSpec. For every *vo* that covers SitSpec elements that have a salience feature attached to it, we can simply check whether the foregrounding or backgrounding is actually achieved with that *vo*. The more of the right salience assignments a *vo* performs, the higher its preference value. But, the assignment of 'foreground' to ACTIVITY or POST-STATE must result in a much stronger preference for verbs realizing this assignment.

Note again that evaluating these parameters for every verbalization option establishes only a *local* order for every SitSpec node. This means that preferences are assumed to be "context-free": no interactions between lexicalizations of different nodes are possible when evaluating the preferential factors. This is clearly a simplification that would have to be surrendered if, for example, collocational constraints were to be added to the mechanism.

If no preferred verbalization results from the parameters, then we always use the *most specific* of the candidate lexemes; this is a rule of thumb that often works but is not always correct; we leave a more thorough treatment of choosing on the grounds of specificity as an issue for future work (see section 10.4).

Since the preference evaluation is only a side issue, we do not elaborate it with respect to our example here.

8.2.4 Determine the complete and preferred SemSpec

As already explained in section 8.1, we need to find a subset VO' of the verbalization options such that:

(1) The partial SemSpecs associated with the elements of VO' can be unified into a single, well-formed SemSpec.

(2) The elements of VO' collectively cover the entire SitSpec, i.e., every element in the SitSpec will be expressed in the sentence.

(3) No element of the SitSpec is expressed more than once.

(4) The resulting SemSpec is *preferred.*

Well-formedness (1) of the SemSpec is guaranteed by constraints specified in the Upper Model concepts (the names of which appear in the partial SemSpecs of the lexical options). Specifically, the PROCESS concepts in the UM define which participants are obligatory, and what UM type the participants should belong to. Requirement (2) makes sure that no element of the SitSpec is excluded from the verbalization (unless the element is marked as optional). There are two exceptions to this requirement, i.e., nodes that do not have to be on the covering-list of a participating verbalization option:

- A PATH node is automatically covered as soon as all its dependent nodes are covered. That is, if some *vo* covers the DESTINATION, and there are no other nodes present in the PATH, then the PATH node itself is covered.

- In a CULMINATION, the PRE-STATE node and the associated VALUE do not have to be covered. If the ACTIVITY and the resulting POST-STATE are covered, then that is enough to verbalize the situation—because the PRE-STATE is in opposition to the POST-STATE

```
Procedure BuildSemSpec(sitspecnode, [vo_1, ..., vo_n])
i := 1
L1: vo := vo_i
    IF Saturated(vo.psemspec)
       THEN RETURN [vo.psemspec, vo.covering]
       ELSE FOR EVERY ext_var IN vo.psemspec
                    newnode := CorrespondingNode(ext_var, vo.denotation, sitspec)
                    result := BuildSemSpec(newnode, newnode.vo)
                    IF result = 'fail' OR UM-Incompatible(result.semspec)
                       THEN IF i = n
                               THEN RETURN 'fail'
                               ELSE i := i + 1
                                    GOTO L1
                       ELSE ext_var := result.semspec
                            vo.covering := vo.covering ∪ result.covering
            RETURN [vo.psemspec, vo.covering]
```

Figure 8.4: The procedure for building SemSpecs (simplified)

and the prior value of the state is often not relevant. Or, it can be expressed with an adjective that does not verbalize the complete PRE-STATE: *The empty tank was filled by Tom.* This is not discussed further, though.

In order not to complicate the procedure of checking the completeness of a verbalization, we simply attach an 'optional' label to nodes of these kinds in every SitSpec.

Requirement (3) ensures that the resulting sentence is *minimal* in the sense that the generator does not produce a sentence like *We drove by car*, which covers the INSTRUMENT with both the verb and a prepositional phrase.

We use a weakened notion of preference (4), as explained in section 8.1. A local order of preference is already determined for the lexical options at every SitSpec node (section 8.2.3), and SemSpec construction attempts to use the partial SemSpecs in that order at every node. In the optimal case, the preferred options for every node yield a well-formed SemSpec. But all the same, it might happen that at one particular node the algorithm succeeds only with the worst option, whereas the problem might have been solved by accepting a somewhat less preferred option at a different node, resulting in an overall better solution. Thus, there is no guarantee of finding the 'global' best solution, except by an exhaustive search of the entire option combination space.

The procedure 'BuildSemSpec' is given in figure 8.4. It takes as arguments a SitSpec node—the one that a SemSpec is to be built for—and the ordered list of verbalization options *VO* that cover this node (recall that in the matching phase the *vo*s were placed on the 'covered-by' list of the respective SitSpec nodes).

When verbalizing a SitSpec, we make use of the obvious fact that the SemSpec we are looking for must cover the root node of the SitSpec. We thus apply the BuildSemSpec procedure to the root node and try, in their order of preference, to saturate one of the PSemSpecs associated with it. As soon as one can be saturated, which also covers the complete SitSpec (except possibly for optional nodes), we have the result.

BuildSemSpec returns a vector of two components, result.semspec and result.covering. The procedure works as follows. The current *vo* is set to the most preferred one (line 2). If the PSemSpec component of the *vo* is already saturated, we are finished and return the result vector for the current *vo* (line 4). Otherwise, every external variable in the PSemSpec needs to be replaced by a saturated SemSpec. In line 6, the procedure CorrespondingNode is called; it determines the SitSpec node that corresponds to the external variable shared by *vo*.psemspec and *vo*.denotation. This is the SitSpec node that needs to be handled in order to replace the current external variable; line 7 has the recursive call of BuildSemSpec. If the result is a well-formed SemSpec, the external variable is replaced with that SemSpec (line 13), and the covering list of the *vo* is extended with the covering list determined by the recursive call (line 14). If, on the other hand, the recursive call did not succeed in finding a saturated SemSpec, or if the result does not respect the constraints imposed by the UM (line 8), we need to backtrack (lines 9–12). If we have further *vo*s available, then the next one is tried; otherwise the procedure has to return 'fail'.

Once BuildSemSpec has been applied to the root node of the SitSpec and has produced a SemSpec and a covering list, this covering-list is compared to the list of SitSpec nodes. Nodes marked as 'optional' are not required to be included in the covering-list; but if any other SitSpec node is missing, i.e., left uncovered, then the procedure has to be invoked again, with only the remaining *vo*s that have not been tried yet.

Two additional tasks are performed by the procedure but are not included in the description in figure 8.4. For one thing, the internal variables of PSemSpecs are renamed when unifying them into a SemSpec, because they need to be unique in the overall result; the lexicon entries always use `x` as the internal variable. And finally, when replacing a variable by a new PSemSpec, a type shift may have to be performed, as was explained in section 7.2.1.

Example Returning to the Jill-filled-the-tank example, the 'covered-by' list of the SitSpec node `event-1` contains, amongst others, the *vo* `fill3` given in figure 8.3. Assuming here that this option is first in the list, the BuildSemSpec procedure inspects the PSemSpec associated with `fill3` and has to handle the external variables `Y, A`, and `B` in turn. These variables are looked up in the denotation, and the corresponding nodes in the SitSpec are determined, which are `jill-1, water-1`, and `tank-1`, respectively. For each node, the procedure calls itself and immediately returns the PSemSpecs associated with the nodes (see figure 8.2, because they are already saturated. Thus, they replace the variables, no further recursion is necessary, and the final SemSpec is as shown below, together with the covering-list that results from merging the PSemSpecs.

```
(x1 / directed-action :lex fill_el
    :inclusive (x2 / substance :lex water_el)
    :actee (x3 / object :lex tank_el)
    :actor (x4 / person :name jill))

COV: (fill-state-2 'full pour-1 event-1 jill-1 water-1 tank-1)
```

8.2.5 Generate sentence

The resulting SemSpec is finally given to the front-end sentence generation module that is in charge of the language the system is set to. The English module is an adaptation of the PENMAN system, with several extensions to UM and grammar made in the TECHDOC project at FAW Ulm. The German variant has also been developed at FAW; it was developed for the application of generating instructional text and has a smaller overall coverage than the English PENMAN, but a range of phenomena that are important for German have been worked out in detail.

PENMAN, as described earlier, is based on a large systemic-functional grammar (called NIGEL). Any grammar of this kind is a large network of choice points (called 'systems') at which a functional decision is made, which has an associated realization statement as well as output features that influence the decisions at other points in the network. A realization statement, however, can be very fine-grained and its consequences need not be immediately 'visible' in the resulting output sentence. Every system in the network has an associated 'chooser', which is in charge of making the decision on how to proceed, i.e., of setting the output-features of this system and propagating the associated realization statement. When the network (which consists of about 800 choice points) is traversed, all the realization statements are collected, and at the end they amount to a full specification of a sentence. Choosers make their decision by inspecting the input expression (in PENMAN terminology, an SPL expression; in our terminology, a SemSpec) and querying the Upper Model: The UM type of an entity in the SPL/SemSpec influences the realization decisions.

Example The finished SemSpec is given to PENMAN, which produces the sentence *Jill filled the tank with water.*

8.3 Implementation of a prototype: MOOSE

The architecture described in the previous sections has been implemented in a prototypical system called MOOSE. It is built in Common Lisp and uses LOOM; the domain model as described in chapter 5 is fully implemented as a basis for MOOSE to operate on. The input to MOOSE is a SitSpec as defined in chapter 5, and the system produces a range of verbalizations of that SitSpec, differing along the dimensions we have have discussed. A screendump of MOOSE is shown in figure 8.5, and further examples of its input and output will be given in chapter 9.

The main menu of MOOSE offers to perform one of the following steps for the user, corresponding to the steps described above:

(1) 'Build SitSpec' activates a tool that assists the user in composing a well-formed SitSpec.

(2) 'Lexify SitSpec' matches the SitSpec against the lexicon of the current target language and links the SitSpec nodes to all the verbalization options that can cover the node.

(3) 'Compute Alts/Exts' applies the alternation and extensions rules belonging to the verbalization options that are associated with the SitSpec nodes and thereby extends the number of *vo*s available at the nodes.

(4) 'Build SemSpec' applies the SemSpec construction procedure to the root node and the *vo*s stored there, and hence to the entire SitSpec.

(5) The option 'Set parameters' brings up a menu for configuring the generation process. The target language can be set to English or German, and target values for two stylistic dimensions can be chosen (merely to demonstrate the functionality). Also, there is a setting for having the system generate all possible verbalizations instead of only the preferred one, which is useful for debugging purposes.

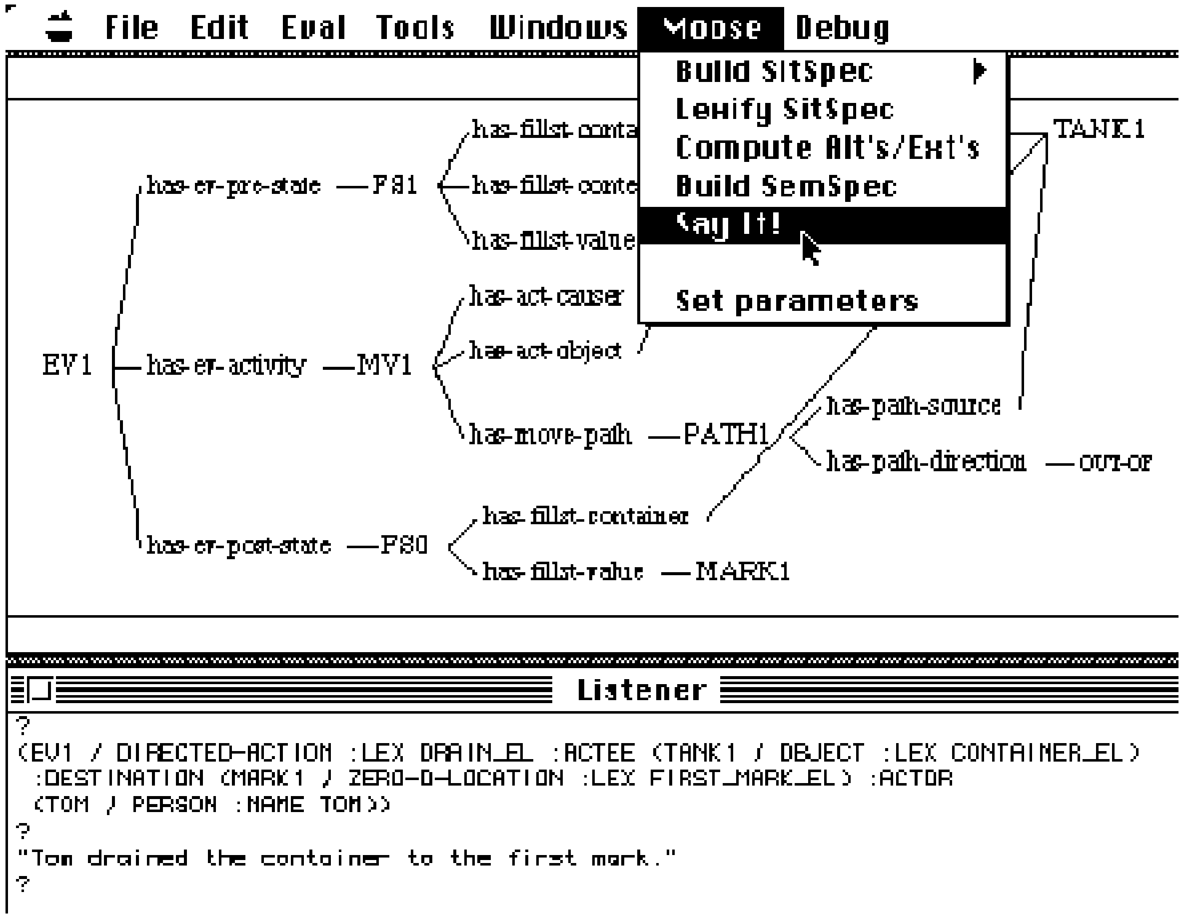

Figure 8.5: Screendump of Moose

(6) 'Say it' gives the current SemSpec to the surface generator (chosen according to the parameter setting), which converts it to an English or German sentence.

Once a SitSpec has been constructed, MOOSE provides a graphical representation of it on the screen.[2] Each SitSpec node is "clickable", and a menu of node-specific functions allows the user to, for example, inspect the verbalization options for that node, or to build the SemSpec for the subgraph rooted in the clicked node, and give it to PENMAN. That is, individual parts of the SitSpec can be verbalized separately. Furthermore, the menu at a SitSpec node offers to set or reset any of the features 'optional', 'foreground', and 'background' for that node.

The standard way of using the system is to build the SitSpec, assign labels for foregrounding, backgrounding, and optionality, and then to execute the options of the main menu in the order given above, so that the preferred verbalization for the complete SitSpec results. But, as indicated, the operations can also be executed for sub-graphs only, by clicking on the respective root node. Also, a 'debug' menu at any time allows for inspecting various kinds of information like lexicon entries, for manually re-ordering the *vo*s at some SitSpec node so that a different verbalization can be produced, and for a few other tasks.

The algorithms used in MOOSE have been described above. A noteworthy point is that we make use of LOOM to interface the lexicon with the knowledge base. In a 'context' (a kind of namespace) separate from the standard knowledge base, 'interface instances' are defined for every lexical entry. The type of the instance is that of the root node of the denotation, and the other information (full denotation, covering-list, PSemSpec, connotation features, and salience assignment) are annotated at the interface instance. In the first step of the generation process, when the lexical options for SitSpec nodes are sought, we first execute a LOOM retrieval of those interface instances with a type that is the same as or more general than that of the SitSpec node. In this way, the set of lexical candidates for matching the denotations is reduced from the very beginning.

The user interface of MOOSE is very useful for developing and debugging purposes: Changes made to any of the participating knowledge sources (domain model, lexicon, alternation/extension rules) can be checked quickly for their effects on the verbalization process.

8.4 Embedding MOOSE in larger applications

A generator for single sentences can be useful in some applications, but in general, sentence generation will be one module of a larger system for producing paragraph-size text. Then, the input is no longer a single SitSpec but a set of SitSpecs, which are linked by appropriate relations. TECHDOC and systems similar to it use a 'rhetorical tree' with propositions at the leaves and relations like CONDITION or PURPOSE at internal nodes. Propositions have so far been single predicate–argument structures; replacing them with SitSpecs and developing a more elaborate scheme of linearizing the tree into a sequence of SemSpecs will significantly improve the performance of the overall system. TECHDOC already has mechanisms for computing referring expressions and choosing clause connectives; these have to be integrated and enhanced.

When paragraph-size text is produced, a SitSpec does not necessarily correspond to a single sentence anymore; it can be aggregated with another SitSpec into a complex sentence. The decisions on sentence scope and on choosing appropriate connectives depend on the specific combinations of rhetorical relation and SitSpecs. Our method of sentence generation can "scale up" to encompass such decisions as follows. Given a tree of SitSpecs and relations, the first

[2]Thanks to Mick O'Donnell for making his graphing tool available to me.

step is finding the verbalization options for every node of every SitSpec, just like MOOSE does now. In addition to "purely lexical" entries, we associate verbalization options also with the nodes that correspond to relations linking SitSpecs to one another. These options then specify the clause boundaries and connectives, and the syntactic constraints on the elements, if any. The PENMAN Upper Model already offers the possibility of producing some connected clauses, which is a good basis to start from. Just for illustration, a *vo* for a CONDITION node could have a denotation and PSemSpec like the following:

```
DEN: (condition (SITUATION A)
                (SITUATION B))
PSS: (x / rst-condition  :domain A  :range B)
```

Notice that both `A` and `B` are to be SITUATIONS in our ontology, which range over STATE, ACTIVITY, and EVENT. RST-CONDITION is one of the Upper Model relations for clause complexes, and the resulting sentence will follow the pattern *If A, then B*. A range of such *vo*s for every type of relation node is to be provided. Sentence boundaries are specified within a *vo* by associating a sequence of two PSemSpecs with it. In parallel to our sentence generation scheme, salience information can be attached to the *vo*; e.g., the clause that is placed in front of the other is made more salient. Also, connotation features can be applicable to connectives just as well: For example, *thereafter* is a more formal version of *afterwards*.

In short, if declarative knowledge about sentence planning is encoded in the verbalization options for nodes representing rhetorical relations, our method of sentence generation can in principle be used for paragraph generation as well.

On a level beyond the single clause, some interesting further divergences between English and German need to be dealt with. Stede and Weis [1993] give a catalogue of phenomena for German, English, and French, among them many on the level of clause linking. Similarly, Zydatiss [1990] compares English and German maintenance instructions and finds interesting differences, e.g., for expressing SEQUENCE and PURPOSE relationships.

Chapter 9

Generating paraphrases

In this chapter, we demonstrate with a range of examples how our generation system introduced in the last chapter can derive different verbalizations from the same SitSpec. All examples discussed in sections 9.1 through 9.3 are generated by MOOSE: The SitSpecs are instantiations of the implemented domain model, and MOOSE maps them to the sentences as shown. Section 9.4 then summarizes the features of our approach and how they relate to the tasks of generating the lexical paraphrases introduced in chapter 4.

In all our sentence generation, we abstract from decisions on referring expressions, as noted earlier. We implicitly assume an utterance situation where both speaker and hearer are directly confronted with the objects in question, so that definite determiners may always be used. We also abstract from deciding on tense by assuming a standard reporting situation in which everything is said in the past tense. These two settings of definiteness and tense can be set to default values in PENMAN, so the features need not be given in the input SemSpecs.

We now give examples of verbalizing each type of SITUATION, but will concentrate on EVENTS, since they have been in the focus of attention in the thesis.

9.1 Verbalizing states

9.1.1 Binary states

A binary state relates some OBJECT to the value of some attribute of that object; specifically, we have introduced TEMPERATURE-STATE and PRESSURE-STATE in chapter 5. A standard way of verbalizing such relationships is using a copula (in English *to be*, in German *sein*) with the OBJECT as subject and the value as direct object: *The water was warm. The pressure was high.* The Upper Model abstraction over such realizations is the concept PROPERTY-ASCRIPTION, which is a relation between an arbitrary THING and a QUALITY. For instance, the SemSpec for *The water was cold* as well as *Das Wasser war kalt* is this:

```
(x1 / property-ascription
   :domain (x2 / object :lex water_el)
   :range (x3 / sense-and-measure-quality :lex cold_el))
```

The lexical entry that produces this structure is the same for both English and German:

```
NAM: X-is-Y
LAN: E/G
DEN: (binary-state (OBJECT A)
                   (VALUE B))
```

```
COV: (binary-state)
PSS: (x / property-ascription  :domain A  :range B)
```

If the values are modelled as a numeric scale, though, the verbalization requires more specific entries that produce, for instance, *The water was 23 degrees* (in German, interestingly, *Das Wasser hatte 23 Grad*, literally 'the water had 23 degrees') or *The temperature of the water was 23 degrees*. These entries are quite similar, and we do not show them here.

9.1.2 Ternary states

For the ternary states, there is no general verbalization pattern applying uniformly to all of them.

Locations A LOCATION-STATE in our model relates a "located object", the LOCATUM, to a LOCATION, and the LOCALIZER specifies the kind of spatial relationship between the two. To verbalize the simple relationships needed in our domain (nothing as complex as *behind*, *below*, etc.), we need to use the UM relation NONORIENTING, whose domain is the LOCATUM and whose range is the LOCATION. Importantly, the LOCATION needs to be of the right UM DIMENSION type, which depends on the LOCALIZER. We have already given the example of the *Danube* type shift in section 7.2.1. The analogous lexical entry for expressing something being contained in a three-dimensional location is as follows:

```
NAM: in-location
LAN: E/G
DEN: (loc-state (LOCATUM A)
                (LOCATION B)
                (LOCALIZER 'in)
COV: (loc-state 'in)
PSS: (x / nonorienting  :domain A  :range (ts B three-d-location))
AER: (location-limit)
```

When combining this entry with the entries for *tank* and *water*, the UM-type of the LOCATION *tank* is shifted from OBJECT to THREE-D-LOCATION, and the verbalization option for *The water was in the tank / Das Wasser war in dem Tank* results.

By analogy, for the localizer `'on`, we have an entry that is the same except for the type shift `(ts B one-or-two-d-location)`, and for `'at-point`, the term is `(ts B zero-d-location)`.

Connections CONNECTION-STATES relate a CONNECTOR to a CONNECTEE, and a symbolic value indicates how loose or tight the connection is. The typical verbalizations express the specific type of connection, and sometimes also the topological properties of the objects, as in *The cap was screwed onto the opening / Der Deckel war auf die Öffnung geschraubt.*

Let us illustrate here the verbalization with verb participles, using the example of the unspecific verb *to connect*, as in *The plug was connected to the socket* (the German sentence is equivalent). The SemSpec for this sentence is

```
(x1 / directed-action  :lex connect_el
    :actee (x2 / object :lex plug_el)
    :destination (x3 / object :lex socket_el))
```

Observe that, contrary to the SemSpecs we have seen so far, there is no `:actor` present in the expression. That reflects the systemic-functional perspective of the situation: there is a process going on (the connecting) and it affects something (the plug), but it is not known who the causer is. Accordingly, the participle *connected* denotes exactly the second half of the denotation of the full verb *to connect*, which expresses a transition from a 'disconnected' state to a 'connected' state. In German, the situation is the same with the verb *verbinden* and its participle *verbunden*, so we look only at the English verbalization here. If the VALUE is `'loose` or `'tight`, it can be expressed with an adverb (see below). if it is `'disconnected`, the verb does not apply at all. Hence, this value is excluded in the denotation:

```
NAM: connect_pass
LAN: E/G
DEN: (connection-state (CONNECTOR A)
                       (CONNECTEE B)
                       (VALUE (not 'disconnected)))
COV: (connection-state)
PSS: (x / directed-action  :lex connect_el  :actee A  :destination B)
AER: (manner)
```

A verbalization using this lexical entry will cover the CONNECTION-STATE as well as the two parts, once they are filled in. The VALUE is not covered, though, and adding it to the verbalization is done by an extension rule that adds a `:manner` field to the PSemSpec:

```
NAM: manner
LAN: E/G
DXT: (connection-state (VALUE A))
COV: ()
ROC: ()
NRO: (:manner A)
```

Assuming that the STATE has either of the two values `'tightly-connected` or `'loosely-connected`, we need two adverbs, which happen to have the same lexical entry for both languages. For *tightly*, it is this:

```
NAM: tightly
LAN: E/G
DEN: (connection-state (VALUE 'tightly-connected))
COV: ('tightly-connected)
PSS: (lex / sense-and-measure-quality :lex (tightly_el / fest_gl))
```

Its `PSS` will replace the `A` in the extension rule, and the overall SemSpec thus corresponds to *The plug was tightly connected to the socket.*[1]

For a CONNECTION with value 'disconnected', in English we produce the analogous *The plug was disconnected from the socket* with the following entry, which replaces `:destination` with `:source`. The manner-extension does not apply here.

```
NAM: disconnect_pass
LAN: E
DEN: (connection-state (CONNECTOR A)
                       (CONNECTEE B)
                       (VALUE 'disconnected))
COV: (connection-state)
PSS: (x / directed-action  :lex disconnect_el  :actee A  :source B)
AER: ()
```

[1]The lexical entry for *tightly* is tailored specifically to the CONNECTION-STATE in our domain; a more thorough treatment of adverbials would need abstractions similar to those we are encoding for the verbs in our approach.

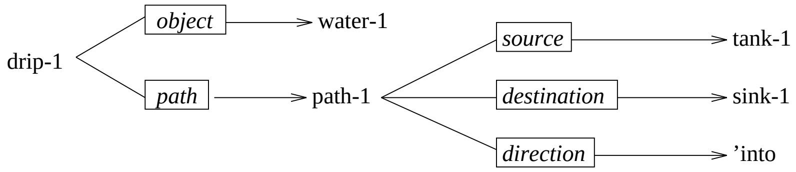

Figure 9.1: SitSpec for water dripping from tank

Fill-states The verbalizations of FILL-STATES are almost congruent in English and German. Simple predications like *The tank was full / Der Tank war voll* are exactly parallel. An extension rule can optionally insert the CONTENT of the container. Here, however, we have an interesting divergence between the languages. In English, the standard form is *The tank was full of water*. German uses the preposition *mit*, which is largely translation-equivalent to *with*. But the English *The tank was full with water* conveys the information that the water is somewhat out of place, only unexpectedly in the tank. In German, we would convey this with *Der Tank ist voller Wasser*. Thus, to get the 'unmarked' reading that neutrally describes the state of the tank, we need different prepositions in the two languages.

If the value is neither 'full' nor 'empty', then an appropriate verbalization is *The tank is full up to the nth mark / Der Tank ist voll bis zur nten Marke*. In the SemSpec, this extent of the FILL-STATE has to be given to the role `:limit`. To capture both usages of *full*—with or without a qualifying prepositional phrase—in the same lexical entry, we use a default, whose function was explained in section 7.1.1. If matched to a FILL-STATE SitSpec with VALUE 'max-mark', the verbalization will be *The tank was full*; otherwise, the mark will also be verbalized, and the overall result is as stated above.

```
NAM: full
LAN: E
DEN: (fill-state (CONTAINER A)
                 (VALUE (< B 'max-mark >))
                 (LOCALIZER 'in))
COV: (fill-state 'in)
PSS: (x / property-ascription :domain A
                              :range (lex / sense-and-measure-quality
                                            :lex full_el)
                            (< :limit B >))
```

As with verbalizing the CONNECTION-STATE, we can also use a participle and say *The tank was filled to the second mark with water*. The participle of *to fill* is defined in the same way as that of *to connect*, and the default and the extension rule are the same as for *full*.

9.2 Verbalizing activities

With ACTIVITIES, verbalization becomes more interesting. To illustrate the function of inheritance and covering, we show one example in detail here: a situation in which water dripped from a tank into a sink. Figure 9.1 gives the SitSpec. Recall that in the domain model we have a general MOVE concept as PROTRACTED-ACTIVITY, and it has several sub-concepts representing manners of movement, among them DRIP.

We take three verbs to be applicable in this situation and show their denotations:

- The very general *to move*, expressing that something undergoes movement.
  `(move (OBJECT A))`

- The more specific *to drain*, expressing that some liquid moves out of some container.
  `(move (OBJECT (A liquid)) (PATH (SOURCE (B container))))`

- *To drip*, more specific than *to move*, and different from *to drain*, expressing that some liquid moves along a path in a dripping manner. We assume that in the domain model MOVE subsumes DRIP.
  `(drip (OBJECT (A liquid)) (PATH B))`

From the denotations we can infer that *to move* is applicable in any of the situations where *to drain* or *to drip* is applicable, whereas *to drain* and *to drip* are not in a hyponymy relation. This is because *to drain* requires a container as the source emitting the liquid, whereas *to drip*, while more specific in the manner of movement, is less restrictive with respect to the path. The source can be unknown, but there must be *some* path expression, e.g., *Water dripped onto the floor.*

After matching the lexemes against the SitSpec, we have the following three verbalization options for the ACTIVITY, where the denotations and covering-lists are instantiated to the name of the ACTIVITY (the entries for *move* and *drain* were in part already shown in figure 7.7):

```
NAM: move
LAN: E
DEN: (drip-1 (OBJECT A))
COV: (drip-1)
PSS: (x / nondirected-action :lex move_el  :actor A)
AER: (durative-causative  path-source  path-destination  bounded-path-destination)

NAM: drain
LAN: E
DEN: (drip-1 (OBJECT (A liquid))
          (PATH (SOURCE (B container))))
COV: (drip-1)
PSS: (x / nondirected-action :lex drain_el  :actor A  < :source B >)
AER: ((locative/clear-intransitive  resultative-causative)  durative-causative
      path-destination)

NAM: drip
LAN: E
DEN: (drip-1 (OBJECT (A liquid))
          (PATH B))
COV: (drip-1)
PSS: (x / nondirected-action :lex drip_el  :actor A  :path B)
AER: (substance-source)
```

A range of alternation and extension rules are then applied and the extended denotations matched against the SitSpec, and several of them succeed. The various extensions of *to drain* were already shown in figure 7.6. Of the resulting *vo*s, we show here only the base forms extended by the PATH extensions:

```
NAM: move2
LAN: E
DEN: (drip-1 (OBJECT A)
```

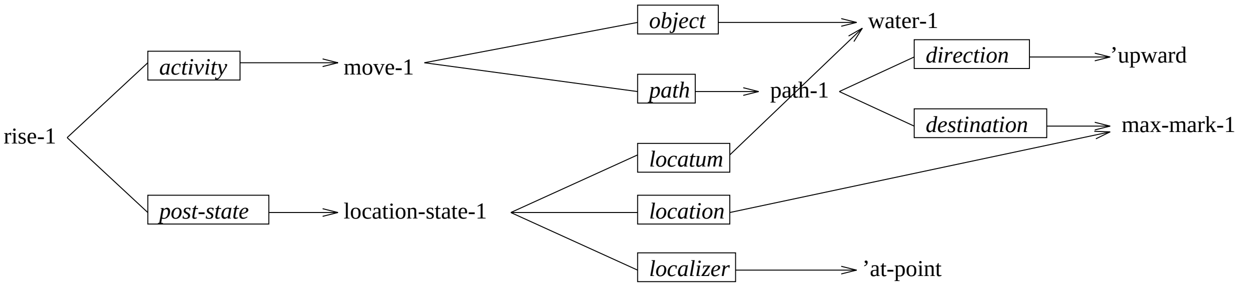

Figure 9.2: SitSpec for water rising in a tank

```
            (PATH (SOURCE B)
                  (DESTINATION C)))
COV: (drip-1)
PSS: (x / nondirected-action :lex move_el  :actor A  :source B  :destination C)
AER: ()

NAM: drain2
LAN: E
DEN: (drip-1 (OBJECT (A liquid))
            (PATH (SOURCE (B container))
                  (DESTINATION C)))
COV: (drip-1)
PSS: (x / nondirected-action :lex drip_el  :actor A  :source B  :destination C)
AER: ()
```

These two together with the *vo* for *drip* given above result in three very similar SemSpecs and their respective verbalizations:
The water moved from the tank into the sink.
The water drained from the tank into the sink.
The water dripped from the tank into the sink.
In this case, the sentences have the exact same behavior with respect to salience distribution, and therefore a choice could be made only on the grounds of preferring the most specific verb. Since *to drip* has the most specific root concept (DRIP), it would be the preferred verb. (In fact, with slightly more elaborate representations it would be possible to account for the fact that *drip* emphasizes the MANNER of movement, whereas *drain* emphasizes the fact that there is some SOURCE of the movement, and *move* is completely neutral.)

9.3 Verbalizing events

Rising water Turning now to complete events, we begin with a small example demonstrating the option of incorporating an element into a more specific verb. The situation is that the water level in a tank rose up to the 'max'-mark on a scale. To simplify slightly, we abstract from the *level* and assume a SitSpec where the water is the object undergoing movement. Figure 9.2 shows this.

To rise denotes movement along a path whose direction is 'upward', and the lexeme covers the move ACTIVITY as well as the direction. Details of the PATH can be added as circumstances.

```
NAM: rise
```

```
LAN: E/G
DEN: (move-1 (OBJECT A)
             (PATH (DIRECTION 'upward)))
COV: (move-1 'upward)
PSS: (x / non-directed-action :lex (rise_el steigen_gl) :actor A)
AER: (path-source path-destination)
```

Once the path–destination extension has been produced, we straightforwardly arrive at the verbalization *The water rose to the max-mark*. Imagine now that in the SitSpec, the value `'upward` has a 'foreground' label attached to it. In this case, we would not want to incorporate it into the verb; to be emphasized, it needs to be a separate constituent. The alternative is to say *The water moved upward to the max-mark*. We have shown the *vo* for *to move* above; it would combine with the path to give *The water moved to the max-mark*, which still leaves the direction uncovered. This coverage is again provided by an extension rule that adds the adverb *upward*, whose lexical entry is as follows:

```
NAM: upward
LAN: E/G
DEN: (path (DIRECTION 'upward))
COV: ('upward)
PSS: (x / nonscalable-quality :lex (upward_el aufwaerts_gl))
```

Once the extension rule has been executed and added the `:manner` to the SemSpec, everything is covered, and hence the paraphrase *The water moved upward to the max-mark* is denotationally equivalent to *The water rose to the max-mark*.

Re-installing the cap Consider now a similar case, where we arrive at language-specific verbalizations and can make a stylistic choice. An example from the Honda manual quoted earlier is the instruction *Replace the cap*, given at the end of the instructions for changing engine oil. The German version is *Den Deckel wieder anbringen* (lit. 'the cap again install'). In English, *to replace* is ambiguous between a sense similar to *substitute* or *exchange* and one where the prefix *re-* indicates that a previous state is restored. The latter sense is the one needed here. In German, it is not possible to incorporate the restoration facet into the verb; instead it has to be expressed with an adverb like *wieder* ('again'). If the English sentence were to use the verb *to install*, there would be a choice of saying either *reinstall* or *install again*.

Given that the feature of restoration is coded in the SitSpec as a role attached to the activity, say INSTALL,[2] then the mechanism explained in the previous example operates here in the same way. Only, for German there would be no incorporation option. There is a choice between two synonyms, though: *Wieder* is a regular, 'core' adverb, and *abermals* is a highfalutin expression with the same denotation. For instance, if a speaker does not like doing something (e.g., installing the oil filler cap), then he or she can indicate the displeasure of having to do it once more by using *abermals*. (There is a similar distinction in English between *again* and *once more*, which we ignore here.) Unlike *wieder*, *abermals* tends to precede the direct object, a fact that the surface generator would be in charge of knowing. To place additional emphasis on it, it can be thematized. In short, the system can produce the variants given below and make its choice on the basis of salience or, in German, of the connotations.
Tom replaced the cap.
Tom installed the cap again.

[2] This is not difficult, because a planning module would have the information on what actions reverse the effects of others and thus can add the role to the concept

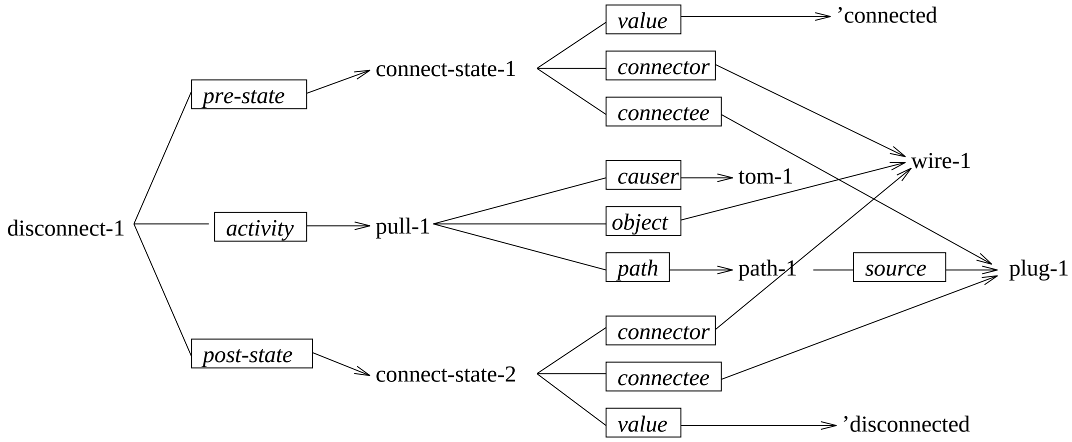

Figure 9.3: SitSpec for Tom disconnecting the wire

Tom brachte den Deckel wieder an.
Tom brachte abermals den Deckel an.
Abermals brachte Tom den Deckel an.

Pulling the wire Next, we consider the difference between focusing on the activity and on the state transition. Recall the example of a person disconnecting the spark plug wire from the spark plug, given in chapter 3, where the original bilingual text (which was in the form of an instruction, though) gave two sentences that are not generally translation-equivalent:
Tom disconnected the wire from the plug.
Tom zog das Zündkabel von der Zündkerze ab. ('Tom pulled the wire off the spark plug')
Disregarding a variety of other possible paraphrases, we just show how these two originate. Figure 9.3 gives the SitSpec underlying both verbalizations.

For the English sentence, the lexical entry of *to disconnect* is a complex one, because the verb is, in its most basic form, already TRANSFORMATIVE and CAUSATIVE.

```
NAM: disconnect
LAN: E
DEN: (event (PRE-STATE (V1 connection-state (CONNECTOR A)
                                            (CONNECTEE B)
                                            (VALUE V2 (not 'disconnected))))
            (ACTIVITY (V3 (CAUSER C)))
            (POST-STATE (V4 connection-state (VALUE 'disconnected))))
COV: (event V1 V2 V3 V4 'disconnected)
PSS: (x / directed-action :lex disconnect_el :actor C :actee A < :source B >)
AER: ()
```

The entry covers the PRE-STATE and the POST-STATE, as well as the ACTIVITY and the VALUES of the states. The SemSpecs of the remaining elements replace the variables in the `PSS`, and the sentence results, either with or without mentioning the SOURCE, because it is optional in the case frame. Notice that all the SitSpec nodes are indeed covered, because the elements of the ACTIVITY are also elements of the STATES, except for the CAUSER, which is contained in the case frame of the verb.

The alternative English sentence *Tom pulled the wire off the spark plug* can be produced in the same way, and the generator can make its choice on the grounds of a desired emphasis setting.

In German, *ziehen* is largely translation-equivalent to *to pull*; it is DURATIVE and thus denotes only the ACTIVITY from the SitSpec. It does, however, undergo extensions that morphologically add the prefix *ab-* to the verb (similar to the particle *off* in English) and then it denotes in addition the fact that the object pulled is afterwards disconnected from its original location. The Aktionsart thus becomes TRANSFORMATIVE+CAUSATIVE; hence *abziehen* also covers the POST-STATE. Our system does not handle morphological derivation, though, so we treat *abziehen* as a separate entry.

```
NAM: abziehen
LAN: G
DEN: (event (PRE-STATE (V1 connection-state (CONNECTOR A)
                                            (CONNECTEE B)
                                            (VALUE (not 'disconnected))))
            (ACTIVITY (pull (CAUSER C)
                            (OBJECT A)
                            (PATH (SOURCE B))))
            (POST-STATE (V2 connection-state (CONNECTOR A)
                                             (CONNECTEE B)
                                             (VALUE 'disconnected))))
COV: (event pull V1 V2 'disconnected)
PSS: (x / directed-action :lex abziehen_gl :actor C :actee A < :source B >)
AER: ()
```

The case frame is analogous to that of *disconnect*, and the variables are replaced in the same way, so that we arrive at the SemSpec for *Tom zog das Zündkabel von der Zündkerze ab.*

Notice that in this example, the denotations of verbs such as *to disconnect* and *to pull off* are very different; yet MOOSE can produce appropriate sentences using either of them. This results from our encoding fine-grained event representations and verb denotations, which allow for covering the same input SitSpec in quite different ways.

Filling the tank In chapter 8, we have explained the production of the sentence *Jill filled the tank with water.* Two variants of this sentence were also given in chapter 6 (figure 6.5), and we will now sketch their production.

Jill poured water into the tank until it was filled results from the same SitSpec, and is given as the preferred sentence when the ACTIVITY in the SitSpec is marked with a 'foreground' label. *Jill poured water into the tank* is the verbalization of the subgraph rooted in the ACTIVITY, which is produced in analogy to the sentence discussed above in section 9.2. *The tank was filled* {*with water*} correspondingly is the verbalization of the POST-STATE, which we have explained in section 9.1. The missing link is the connective *until*, which applies only to PROTRACTED-ACTIVITIES (such as *to pour*) but not to MOMENTANEOUS ones (such as *to turn green*). The Upper Model concept expressing the relationship between the activity and the state that ends it is ANTERIOR-EXTREMAL; in English, it yields a sentence of the structure *A until B*, and in German the corresponding *A bis B* (with different word orders in the English and German subordinate clauses *B*, which are handled by the generation grammars).

```
DEN: (event (ACTIVITY (protracted-activity A))
            (POST-STATE B))
```

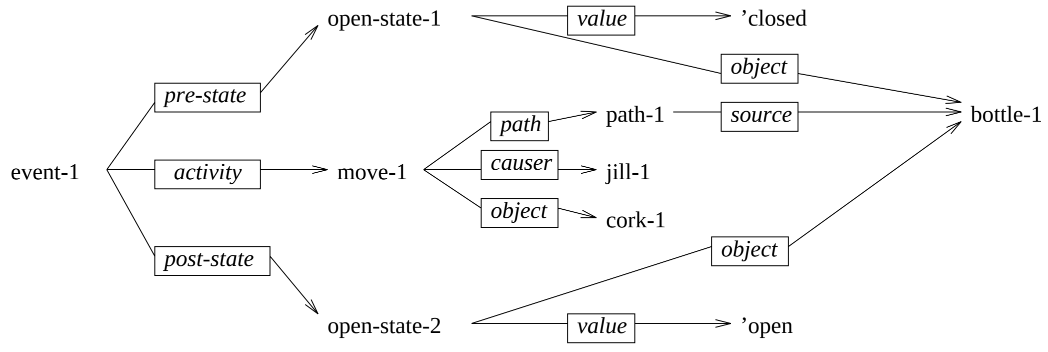

Figure 9.4: SitSpec for Jill uncorking the bottle

```
COV: (event)
PSS: (x / anterior-extremal :domain A  :range B)
```

This entry matches the EVENT node of the SitSpec and combines with the two clauses mentioned above to yield the complete sentence.

In chapter 7, we have mentioned variants of this sentence as resulting from different salience assignments. To reiterate just one of them here: If ACTIVITY and POST-STATE have no salience label attached to them, but the node corresponding to *water* does, then the system will produce *Jill poured water into the tank until it was filled* as the preferred verbalization, since *water* is assigned the prominent position of `:actee` here. In contrast, *Jill filled the tank with water* makes it an adjunct (an `:inclusive` in the SemSpec), which is a non-prominent position, and hence the sentence would not be preferred.

Uncorking the bottle Finally, we consider an example that demonstrates the role of concept subsumption as a source of paraphrase. The situation is that of Jill uncorking a wine bottle, which was already introduced earlier. Concentrating on English examples, we demonstrate how verbalizations as diverse as the following are produced:
Jill opened the bottle.
Jill removed the cork from the bottle.
Jill uncorked the bottle.
The SitSpec was already shown in figure 5.4 and is repeated in figure 9.4. To understand the connections between the verbalizations, it helps to refer to figure 5.9, which shows the subsumption relationships between LOCATION-STATE, CONNECTION-STATE, and TANK-OPEN-STATE. For our purposes, we assume that the concept BOTTLE is a specialization of TANK.

Consider first *to open*, and assume that, to obtain the causative reading, we have already applied the resultative–causative extension to its basic form. The causative reading is shown below.

```
NAM: open
LAN: E
DEN: (event (PRE-STATE (V1 tank-open-state (CONTAINER A)
                                           (VALUE 'closed)))
            (ACTIVITY (V2 (CAUSER B)))
            (POST-STATE (V3 tank-open-state (CONTAINER A)
                                            (VALUE 'open))))
```

```
COV: (event V1 V2 V3 'closed 'open)
PSS: (x / directed-action :lex open_el  :actor B :actee A)
AER: ()
```

Both *to remove* and *to uncork* are already inherently causative, as their lexical entries demonstrate:

```
NAM: remove
LAN: E
DEN: (event (PRE-STATE (V1 location-state (LOCATUM A)
                                          (LOCATION B)))
            (ACTIVITY (V2 (CAUSER C)))
            (POST-STATE (V3 location-state (LOCATUM A)
                                           (LOCATION (not B)))))
COV: (event V1 V2 V3 (not B))
PSS: (x / directed-action :lex remove_el  :actor C :actee A < :source B >)
AER: ()

NAM: uncork
LAN: E
DEN: (event (PRE-STATE (V1 location-state (LOCATUM (cork A))
                                          (LOCATION (bottle B))))
            (ACTIVITY (V2 (CAUSER C)))
            (POST-STATE (V3 location-state (LOCATUM (cork A))
                                           (LOCATION (not B)))))
COV: (V1 V2 V3 A (not B))
PSS: (x / directed-action :lex remove_el  :actor C :actee B)
AER: ()
```

To remove denotes that somebody moves a LOCATUM, which had occupied a LOCATION, so that afterwards the LOCATUM is no longer in the LOCATION. Importantly, the node `(not B)` in the POST-STATE is covered by this verb, because that is exactly what it expresses: move something *away from* a location.

To uncork is a specialization of *to remove*, as far as the denotation is concerned, but the case frame is quite different. Notice first the selectional restriction: *uncork* is a way of removing that applies only to corks in the openings of bottles. And the verb also covers the cork, because the fact that *a cork* is removed is an inherent part of the meaning. As a consequence, the LOCATUM cannot occur in the case frame; instead the `:actee` must be the bottle, and there is no optional `:source`.

When comparing the denotations to the SitSpec, with the subsumption relationships defined in figure 5.9 in mind, it becomes clear that the denotations of all the verbs indeed match the SitSpec. After all, TANK-OPEN-STATE is subsumed by LOCATION-STATE, and the roles are also in the required subsumption relationships. After instantiating the denotations and producing SemSpecs, they produce the three sentences we have listed above.

Imagine now that an additional attribute were attached to the CORK in the SitSpec, for example the information that it is moldy. Assuming that the only way of verbalizing this attribute is an adjective, then there is only one sentence to describe the situation: *Jill removed the moldy cork from the bottle*. Sentences that use *open* and *uncork* would still be correct and well-formed, but the system notices that they do not cover the node MOLDY (one cannot say, e.g., *uncork the moldy cork from the bottle*); hence they are incomplete.

A similar effect could be established if the SitSpec had a 'foreground' label attached to the CORK: the incorporation in *to uncork* would not be a preferred verbalization.

9.4 Solutions to lexicalization problems

Throughout the last chapters, we have demonstrated that our approach to NLG can produce a variety of lexico-semantic paraphrases. In this section, we collect these individual phenomena and relate them to the list of lexico-semantic problems, given in chapter 4, that we set out to solve. Different features of our approach enable the solution of different problems, and we categorize the following presentation along these features.

Collect lexical options Only by determining *all* lexical options beforehand it is possible to evaluate different possibilities of covering the SitSpec nodes, and specifically to compare various incorporation options. In general, there can be numerous ways of distributing the units of meaning across the lexemes, and they can differ in their values of the preferential dimensions; to find a preferred one, it is necessary to have all options available.

Decompose word meaning By decomposing denotations into fine-grained representations, so that there can be overlap between lexeme denotations and semantic units, we get different 'packing', or incorporation variants, as in *to rise / to move upward*.

Another example given in chapter 4 was *to go by plane / to fly*, and we can add as a third variant *to take the plane*. When modelled as a MOVE with an INSTRUMENT role filled by a PLANE, then the lexeme *to go* denotes only MOVE and ensures that the instrument is verbalized with a *by*-PP. *To take*, in the sense intended here, also covers MOVE, but expresses the instrument as a direct object (the difference is encoded in the PSemSpecs); *to fly* covers the combination of MOVE and the instrument PLANE. The DESTINATION of the trip can in any case be added with a *to*-PP, which is provided by an extension rule (in other frameworks an 'adjunct rule').
John went to Moscow by plane.
John took the plane to Moscow.
John flew to Moscow.

Language-specific incorporations, as in *swim across the river / traverser la rivière à la nage* or *Deckel wieder anbringen / replace the cap* are handled in the same way. For the first example, it is necessary to represent MOVE, MANNER, and PATH separately in the SitSpec; then the denotations and covering-lists of lexemes will produce appropriate packings for the target language selected.

Link word meaning to background knowledge and exploit subsumption of denotations By matching lexemes against the SitSpec not for identity but for subsumption, we get more or less specific nouns and verbs e.g., *tank / windshield wiper fluid tank* or *to empty / to drain*, as candidates for verbalization. The subsumption matching automatically takes care of what we discussed as different 'grain size' in chapter 4: If language A has a specific lexeme to express some part of a SitSpec, but language B does not, then a more general lexeme is found as a candidate without further ado. If additional information, which the specific lexeme of language A incorporates, needs to be expressed, then the covering mechanism will ensure the production of, e.g., an appropriate modifier in language B.

In chapter 4 we mentioned the *conventionalized* specificity of lexemes (illustrated with the example of *to remove*, which is often found as an equivalent of more specific German verbs). This could be handled with our mechanisms by adding an appropriate (and probably genre-specific) preference-value that favors the specific German verbs of physical movement on the one hand, and the more-abstract English verbs on the other. In the absence of other preferential factors, the system will then choose *to remove* when producing English, and *abnehmen* (to take off), *herausziehen* (to pull out), etc., when producing German.

Separate denotation from connotation Lexical entries are complex structured entities, and we keep denotation and connotation separate in the representation of word meaning. This enables the production of paraphrases with synonyms or near-synonyms. They have the same denotation, hence all arrive in the pool of verbalization options, and the preference evaluation can choose one on the basis of stylistic features. For example:
Tom drew the water off the container. (somewhat colloquial)
Tom drained the water from the container.
Tom discharged the water from the container. (highfalutin)

Defaults We have provided a default mechanism for word meaning that accounts for the possibility of conditionally incorporating information into a lexeme. If a tank is being filled up to the maximum, then *to fill* conveys that fact, and the POST-STATE need not be verbalized separately, because it is covered by the verb. If the POST-STATE is different from the maximum, it will not be covered, and therefore the system verbalizes it separately, i.e., in a prepositional phrase like *to the third mark*. This mechanism can be used in all those cases where a word has default information associated with it, which can be overridden. *To eat* conveys that the object is some kind of food, and if it is marked as 'optional', it need not be expressed: *Jill ate* is a complete sentence. But if Jill for some reason ate a twenty-dollar bill, the object would violate the default type and would not be covered by the verb.

Event structure Specifically, by representing the internals of event structure in our SitSpec and linking it to the Aktionsart of verbs, we get variations like *fill the tank with water / pour water into the tank until it is full*, where the first emphasizes the result and the second the activity. Another example of this kind is *disconnect the wire / das Kabel abziehen*, which we discussed in this chapter. The verbs in these clauses have very different denotations, yet they verbalize different aspects of the same event and can thereby appear as equivalent in the instruction manual they are taken from. A similar case from chapter 4 is *Rolläden hochziehen / open the shutters*.

Separate denotation and PSemSpec We distinguish two levels of 'meaning' in lexical entries and separate the denotation from the PSemSpec. By relating these two levels with co-indexed variables, we get variants that link the SitSpec elements in different ways to PSemSpec participant roles. In particular, the alternation rules can re-write PSemSpecs and thereby are responsible for variations like *to give the paper to Mary / to give Mary the paper*. Similarly, different readings of the same verb, which sometimes correlate with different role assignments, are produced by these rules: *Water filled the tank / The tank filled with water / Jill filled the tank with water.*

Also, certain cases of different language-specific constructions are reflected on the level of SemSpecs. The example *Twist the cap until it stops / Drehen Sie den Deckel bis zum Anschlag* results in part from the lexeme-matching phase (the nominalization, *Anschlag*, has no parallel in English), in part from unifying PSemSpecs, and in part from syntactic knowledge that comes in at the end.

The problems discussed by Bierwisch [1983], briefly summarized in section 3.5, could also be tackled with our two levels: A noun like *school* would have a constant PSemSpec, because its contribution to the sentence is always the same, but it could have different denotations for the building, institution, and idea senses. Possibly, and this is the idea of Bierwisch's work, one of these could be derived from the other by general, productive rules.

Factor out syntactic knowledge A number of variation phenomena are finally to be handled by the surface generator, which is in charge of knowing about case assignment and constituent ordering patterns, amongst others. Thus, a language-specific syntactic divergence, as in the English/Spanish example *I like Mary / Me gusta Maria* (see section 4.2) would result from the surface generator. The two verbs have the same denotation (at least for our purposes—there could be some fine-grained differences) and their PSemSpecs assign actor and actee in the same way; the surface generator then knows from morphosyntactic features that the Spanish verb assigns dative case to the actor.

Similarly, the difference between *im Gegenuhrzeigersinn drehen* and *twist counterclockwise* can be confined to the surface generator. The SemSpecs for the clauses would be identical, and PENMAN knows that German prefers to express the direction of twisting with a prepositional phrase in German and with an adverb in English. Producing the difference in constituent order is also within the competence of PENMAN.

The 'head switching' phenomenon, as in *Peter likes to swim / Peter schwimmt gern* (see section 4.2), appears to be a syntactic matter, but the root of the problem is rather deep. With cases like these, it needs to be decided on some grounds that either LIKE or SWIM is the underlying activity and the other a modifying OBJECT role. Assuming that LIKE is the root concept, the lexical entries are in charge of knowing that English *to like* expresses the OBJECT with an infinitive verb if it is of type ACTIVITY, and with a noun phrase otherwise. The SemSpec can then be built accordingly. For German, the adverb *gern* would cover the root concept LIKE, and needs to have the information that the associated OBJECT, if it is an ACTIVITY, can be expressed as a clause to which *gern* is to be attached.

Chapter 10

Summary and conclusions

10.1 Summary of the work

Motivations Natural language generation is a truly interesting task only if it offers mechanisms to derive a range of different verbalizations from the same input representations, with a number of well-defined choice parameters. This statement holds from both the perspective of theoretical research and from that of finding fruitful, practical applications. For theoreticians, an expressive language generator can be a useful testbed for studying paraphrases, comparing their similarities and differences, and relating them to utterance situations in which one paraphrase or another might be the more appropriate. And for 'real-world' applications, NLG has to prove that it can perform better than both retrieving canned text and mapping data to language in a trivial one-to-one fashion. The strength of *generating* language can only be in 'tailoring' the text to particular contexts and audiences—in situations where the same message needs to be phrased in different ways under different circumstances. As a prerequisite, it is important that a generator have a wide range of paraphrases at its disposal.

Crucially, the aspect of *multilinguality* extends both the theoretical and the practical perspective. Thorough cross-linguistic comparisons (or 'contrastive studies') are necessary for multilingual generation, and a running text generator is an excellent vehicle for testing the quality of theories and results. Correspondingly, multilinguality opens up a range of potential new applications. Building the resources for mapping from represented knowledge to language requires great effort, and the gain is considerably higher when the output is needed in several languages, so that translation work can be avoided. An important goal for multilingual generation is to share resources and representations between languages wherever possible; and since we are already concerned with monolingual paraphrases, the aim is not to introduce any additional machinery at all for multilingual generation. Instead, producing output in multiple languages is to be seen as a straightforward extension of the monolingual paraphrase task.

Finally, from the viewpoint of knowledge representation, it is important that the structure of the input representation need not mirror the linguistic structure of one specific natural language. The categories that are useful for the reasoning purposes of some application program are not necessarily the same as the categories needed for generating language—the need to restructure the initial input representation when mapping it to language must be reckoned with.

Architecture The issues just discussed were the basic motivations for developing the approach to NLG presented here. We identified one central shortcoming of previous generation work as a major obstacle to making progress: little attention was given to lexical semantics and to separating the various kinds of knowledge involved. Instead, most generators used to assume a simple mapping between input 'concepts' and lexical items as their realizations, and as a consequence, their paraphrasing capabilities were typically quite limited. Mehl [1994] summarized aptly by saying that the usual correspondence between concepts and words as well as between conceptual roles, semantic case frames, and syntactic valency, produces difficulties in the case of machine translation and multilingual generation, since government-frames are language-specific; and that it furthermore becomes difficult to choose between synonyms or near-synonyms with different syntactic features. In addition, we can add a third problematic case, where reasoning requirements for the knowledge base give rise to categorizations (for concepts and roles) different from those needed for verbalization.

To overcome these limitations, we have proposed here a generation architecture in which the *lexicon* is the central resource for mapping a language-neutral conceptual representation to a language-specific semantic sentence representation; this mapping step can involve a restructuring of the input. To accomplish a flexible mapping, lexical entries are rich in information and structure; we divide them into the following parts:

- Denotation and covering-list: the link to the underlying knowledge representation that defines the applicability conditions for the lexeme;
- Partial SemSpec: the contribution that the lexeme can make to a semantic sentence specification;
- Connotation: stylistic features that make lexemes more or less preferable in a specific context;
- Foregrounding and backgrounding: the relative salience that a verb can assign to the various elements of a situation;
- Morphosyntax: features needed for grammatical surface realization.

In addition, we have proposed a number of alternation and extension rules that derive additional verbalization options. Importantly, the lexical knowledge is specified declaratively and in such a way that language-specific information is separate from language-neutral, general information. Where the target languages are parallel, large parts of lexical entries can be shared. In fact, when generating a sentence, the process is language-specific only at two stages: at the very beginning, the lexicon of the target language is used to determine the pool of verbalization options, and at the very end, the surface generation module uses language-specific grammars and Upper Models, although shared representations are used here as well.

Keeping the different realms separate is essential for a modular, domain-independent generator: when the system is moved to a new application, only the denotations of the lexemes (and the lexical rules working on the denotations) need to be adapted to the new representations, while the other parts will remain largely unchanged. Specifically, the Upper Model and the partial SemSpecs remain stable: a lexeme can *denote* something else in a new domain, but its behavior in a sentence will stay the same.

On the basis of these declarative representations, a process that evaluates lexicalization constraints and preferences finds an appropriate verbalization of an input representation. The central constraint for a lexeme to be used is that its denotation be right, i.e., that it can correctly express some part of the input. The preferences, on the other hand, involve the desired stylistic 'color' of the sentence, and the degrees of salience that are to be attributed to the different elements. In practice, more parameters are to be added; for example, the degree of brevity desired, or the use or avoidance of specialized terminology from a particular sub-language. The two parameters discussed in the thesis illustrate the functionality of the approach.

The system is built around the PENMAN sentence generator and the associated idea of representing the linguistic ontology in an 'Upper Model'. However, we have shifted the role of the UM to be one of lexical types rather than a "conceptual" hierarchy, and the input to PENMAN in our system is a fully lexicalized structure. In particular, we demonstrated the weakness of the UM in encoding the *valency* of verbs and proposed to extend the constraints encoded for the processes in the UM with a verb case frame in the form of a partial SemSpec. Correspondingly, and to accomplish the re-structuring necessary for producing many paraphrases as well as multilingual output, the domain model is no longer subsumed by the UM (in contrast to the original conception of PENMAN) and now forms a separate taxonomy with its own ontological categories.

Events To demonstrate the suitability of the approach, we have chosen one particular generation problem and worked in detail on the representation of *events* and the task of verbalizing them in various ways, emphasizing different aspects. Events are interesting both for theoretical and practical reasons.

Only recently, linguistic research on aspectual structure began to emphasize the need for explicitly representing the internal structure of events, so that the problems of 'aspectual composition' can be systematically explored: how do different descriptions of the same event, using clauses with different verbs and possibly connectives, relate to one another? How can it be explained that the same verb can be used to express quite different kinds of events, and how does this relate to the syntactic environment in the sentence?

From the perspective of applications, the question of verbalizing events is central when the input to the generator is produced by *planning* systems (as in [Mellish and Evans 1989], [Meteer 1994]). Such systems reason in terms of states and operators that transform them, and mapping their output to language requires quite fine-grained representations, as Mellish and Evans [1989, p. 246], for example, point out:

> In common with most plans in traditional AI work, NONLIN plans only encode very weak information about causality and temporal relationships. For instance, when there is an action that achieves an effect, there is no way to tell from the plan whether we are dealing with an instantaneous action, an extended action that terminates as soon as the effect is achieved, or an extended action where the effect is achieved sometime during the execution. A natural language like English provides ways of distinguishing between these cases:
> *Turn on the switch and the light will be on.*
> *Pour in the water until the bucket is full.*
> *Prepare a chicken curry so that the chicken scraps are used up.*

We made a step towards bridging this gap by defining our SitSpecs according to two different requirements. An ontology of events, or general *situations*, was developed by starting from ontologies that are used in work on aspect, and then integrating a compositional representation of complex events. These conform to the basic scheme of traditional planning techniques (pre-state, operator, post-state), and at the same time can be verbalized in different ways, by emphasizing either the operator or the post-state.

We have then given an account of several *verb alternations* in terms of the Aktionsart feature of the configurations, and proposed lexical rules for systematically deriving more complex verb configurations from others. This task was greatly facilitated by operating on the level of SemSpec, whose roots in the Upper Model allow us to abstract from language-specific syntax.

In general, natural language generation can be a useful tool for researching event verbalizations: Fine-grained, paraphrase-neutral representations are required, and rules for their systematic mapping to linguistic output need to be devised.

10.2 Comparison to related work

The 'intellectual roots' of our approach on the side of lexical semantics have been discussed in chapter 3. As for language generation, several earlier ideas influenced the shape of our framework; they have been mentioned during the review of lexicalization in chapter 2. In this section, we point back to some closely related approaches and draw comparisons, some of them in more detail than others.

10.2.1 The role of the lexicon in NLG

The idea of accessing the lexicon very early in the sentence generation process is a common approach—see the survey by Cumming [1986]. Typically, though, the lexical entries merely mirrored the concepts in a simple one-to-one fashion. Earlier attempts on overcoming these limitations and strengthening the role of the lexicon to a central resource for making decisions were the pattern-based generators of Hovy [1988a] and Jacobs [1985, 1987].

Hovy's PAULINE was very strong with respect to producing variants of text in accordance with communicative goals, but paid the price of a rather complicated and to a large extent procedural generation scheme. In contrast, we have stressed the utility of an explicit transition between representation levels, based on declarative specifications (which, in addition, opens the door to multilinguality).

Jacobs's [1985] PHRED had the weakness of encoding *all* information in separate patterns associated with words, thus failing to account for any grammatical generalizations. KING [Jacobs 1987] improved on that and is an approach that resembles ours in several respects: it also emphasizes the role of representing knowledge to reduce the need for procedural code, and exploits inheritance for capturing generalizations. Further, Jacobs advocates viewing "actions as views of complex events" and illustrates the approach with the popular example of treating GIVING and TAKING as different views of the same underlying TRANSFER event. The focal point of KING is different from ours, though: it is concerned with *metaphorical* relationships, with extending a view-relation from one domain to another. But the more important difference is in the way KING represents the various kinds of knowledge: It makes the point of assembling *all* knowledge—conceptual, lexical, phrasal, and syntactic—in the very same inheritance network. While this achieves a certain elegance of description, it at the same time makes it hard to extend the system, and to relate the various paraphrases that can be produced to the reasons for their being preferred. That is why our approach opted for separating the various kinds of information in distinct declarative representations, and introduced a separate level of sentence semantics to account for generalizations on that level. Thus, when all knowledge is amalgamated into a single inheritance network, there cannot be a clear transition between separate levels of representation, and consequently it is difficult to extend the approach smoothly to a multilingual framework.

10.2.2 Word–concept linking

The task of associating concepts from the background KB with surface words has, obviously, been dealt with in NLG for a long time. But earlier systems typically used a rather strict association between input concepts and lexemes, so that deciding on words was hardly a matter of choice. In particular, most systems could not re-structure the input, as the mapping between conceptual roles and semantic participant roles was inflexible.

Our notion of flexibly linking concepts to words and collecting lexical candidates as the first step in generation was inspired by Miezitis [1988], who used a spreading-activation algorithm to determine the similarity between conceptual input and word meaning. Such a mechanism depends on pre-set numerical weights and thresholds, which are difficult to motivate. In order to have clearly predictable matching results based on purely declarative representations, we have replaced spreading activation with a procedure of subsumption checking between denotations and input representation. Furthermore, we entered territory uncharted by Miezitis and embedded our 'lexical option finder' into an overall sentence generation framework.

Other proponents of linking words and concepts more flexibly were Horacek [1990a] and Nogier and Zock [1992]. However, they both map from concepts directly to syntactic objects, and thus the same criticism applies that was made above in discussing KING: Neglecting the role of lexical semantics, and not granting it a separate level of description misses generalizations that can be used to derive parpahrases (as in our implementation of verb alternations), and it does not lend itself to multilingual generation, because the entire work of mapping from conceptual representation to natural language needs to be re-done for every additional target language.

Comparing our approach to the idea of discrimination nets, as they have been employed in many variants in generation, is a far-reaching issue because in some sense almost *every* lexical choice mechanism used in NLG so far can be seen as a discrimination net, as long as the lexicon is more fine-grained than the conceptual representation. However, there are some crucial advantages of our approach. To the idea of matching words against conceptual configurations we add the subsumption check and thereby automatically arrive at a more general word in the case that a specific one is not available in a particular language. Encoding the subsumption relationships in a decision tree (which a discrimination net amounts to in practice) would be an extremely cumbersome task. Further, the discrimination net is always attached to a single concept and thus does not solve the problem of mapping whole configurations of concepts and roles to a single lexeme.

As discussed in section 2.4, the DIOGENES system [Nirenburg and Nirenburg 1988] employs a 'meaning-matching metric' to determine the best possible lexeme and thereby differs from the standard discrimination net approach. Similar to MOOSE, the lexicon entries in DIOGENES, expressed in a frame language, pair a conceptual pattern with linguistic information—but again, the latter is of syntactic nature, and there is no distinction between a conceptual and a semantic level of representation. For illustration, this is the entry DIOGENES has for *boy* (from [Nirenburg and Nirenburg 1988, p. 473]):

```
(make-frame boy
  (is-token-of (value person.CL))
  (sex (value male) (importance 10))
  (age (value  (2 15)) (importance 4))

  (lexeme (value "boy"))
  (syntactic-info (lexical-class noun))
  (para-collocation
      (antonym girl adult)
      (synonym lad kid child)
      (hypernym person))
  (syn-collocations-in (value boy.syn)))
```

The conceptual pattern, in the upper portion, gives the super-type of the (lexical) concept defined here and two slots, each of which has an 'importance value' attached to it. In finding candidate lexemes, these values are used by the meaning-matching metric that determines which word is 'closest' to the input representation being matched. This resembles the notion of preference in MOOSE, but notice that DIOGENES uses importance values even for the 'sex' slot in the 'boy'-frame, that is, it does not treat the feature 'male' as an absolute requirement for using the word. Consequently, given an unfortunate combination of various importance values, the word could in principle result when a the concept of a female person is being lexicalized. In MOOSE, on the other hand, we emphasize the difference between constraints and preferences, and strictly use numerical matching only for 'secondary' features.

On the linguistic side, given in the lower part of the entry, there is little information in frames for nouns. For verbs, there is a slot indicating its transitivity class, but it is not clear how exactly the mapping between conceptual roles and syntactic arguments is to take place. Various collocational relationships are stated but it is also hard to discern how these are used in the system. All in all, DIOGENES starts out with some good ideas on flexibility in lexical choice, but there are few generalizable results that could be adapted to other purposes and systems.

10.2.3 Fine-grained lexical choices

We have emphasized that making progress on the question of actual lexical *choice* was not a goal of this thesis; nonetheless it is useful to compare our system to those systems that were concerned with that task and that had an impact on our design decisions.

The language generator demonstrating the largest variety in relating communicative goals to specific lexical choices is probably PAULINE [Hovy 1988a]. We noted, though, that the mechanisms for arriving at the choices in PAULINE are relatively obscure, and we pointed instead to the need for declarative representations that clearly distinguish the different realms of word meaning, so that a choice mechanism can pay attention to those aspects that are relevant in a situation of utterance. (Another system that performs goal-directed lexical choices is ADVISOR II, based on the FUF generator, which we will describe in the following subsection.)

A different approach to lexicalization is proposed by Wanner [1994], who uses lexical functions [Mel'čuk 1988] as a central instrument for sentence planning and lexicalization. The emphasis is on inter-lexical relationships, the 'collocations', and their influence on discourse production. These are taks that we have not dealt with here; instead, we concentrated on the relationship between knowledge base and lexical semantics and investigated, amongst others, the kind of situation-specific paraphrases whose production requires some inferential abilities, which purely lexical knowledge cannot yield.

10.2.4 Paraphrasing

FUF Our concern for deriving many paraphrases was shared by Elhadad [1993], who developed the FUF generator and the application ADVISOR II on top of it, which was described in section 2.3.2. ADVISOR II generates short paragraphs advising students on whether to take particular courses, and thus uses *argumentative intent* as the central factor in choosing words. Quite a variety of semantic and syntactic paraphrases can be produced. For instance, when the system decides to persuade a student to take an AI course, it can choose between sentences such as *AI covers many interesting topics* and *Many AI topics should interest you.* If, on the other hand, it wants to dissuade the student from taking the course, it says, for example, *AI requires many assignments.*

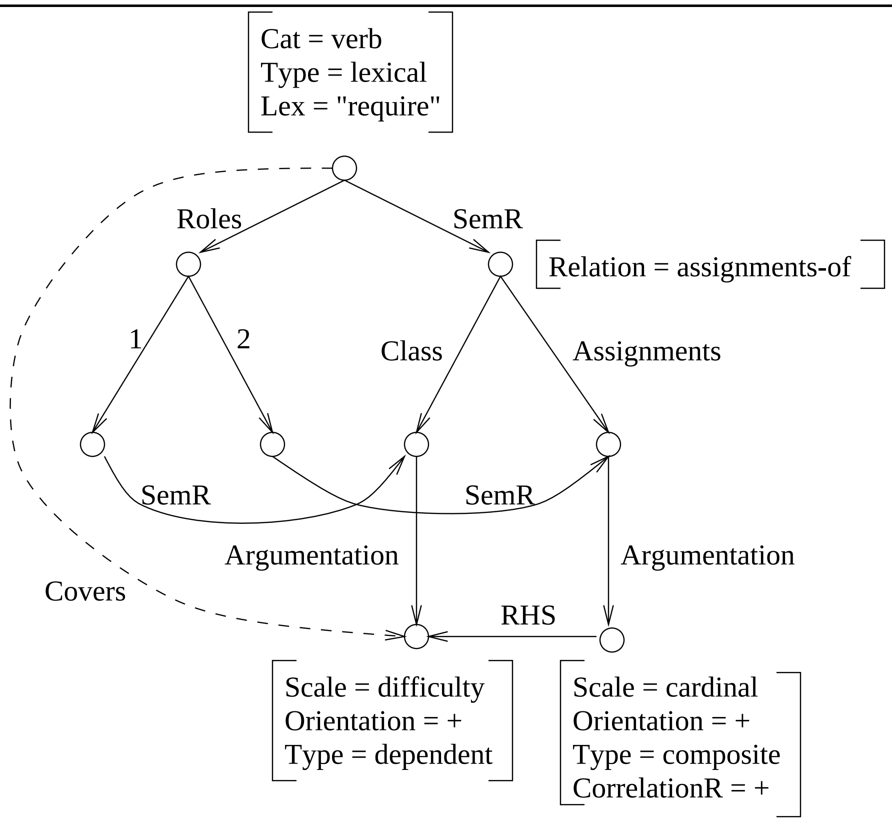

Figure 10.1: Lexicon entry for *to require* from ADVISOR II

The central strength of the NLG strategy used in ADVISOR II is its ability to realize the argumentative intent by different means on different levels of description. Content selection, sentence structuring and connective choice, phrase structuring, and lexical selection can all contribute to conveying a positive or negative opinion on a particular proposition.

ADVISOR II uses lexicalization in much the same way as MOOSE does: lexemes provide the interface between a conceptual representation and a linguistic level of representation, and the lexical choices are made prior to syntactic realization decisions. Thus, lexicon entries look somewhat similar to ours; figure 10.1 shows the ADVISOR II entry of *to require*. 'SemR' links provide the connection between syntactic and semantic objects (feature structures), and hence correspond to our variable co-indexing. The 'RHS' relationship at the bottom represents an argumentative intent to be expressed, and the 'covers' arc from the top node to the RHS feature structure indicates that the syntactic object covers that intent. Without going into further detail, we can see the basic dichotomy between (in our terminology) a denotation and the linguistic information; but the latter is of syntactic nature in ADVISOR II—the arcs labelled '1' and '2' indicate syntactic argument positions. Hence, the system has no intermediate level of representation that would correspond to our SemSpecs. Thus it is not possible to make use of any linguistic generalizations holding on that level, as for example our alternation and extension rules do.

Furthermore, notice that the 'argumentation' features are 'covered' by the syntactic object in the same way as the semantic objects: There is no separation between denotation and connotation that would correspond to our distinction between constraints and preferences; lexicalization in ADVISOR II does not have a preferential component, which we see as indispensable when more than just one connotational dimension is to be accounted for.

The central difference between the FUF approach and ours is, again, in the number of representation levels; Elhadad integrated all paraphrasing capabilities into the surface generator; hence FUF maps the conceptual input directly to linguistic output, and all information is represented in the same feature structure format. This can be seen as a gain in homogeneity, but implies a lack of modularity—Elhadad acknowledges that the system is strongly domain-dependent and difficult to adapt to new tasks.

In contrast, MOOSE was designed with portability as one design goal. In our lexical entries, we have separated the components in such a way that only denotations (and sometimes connotation features) need to be changed when the domain changes—the PSemSpec and the syntactic features will usually stay the same. Furthermore, since our denotations are grounded in the background KB, and subsumption is accounted for in the matching phase, a general word such as *to remove* has a very general denotation in our model, which will in fact generalize across many domains and hence be reusable.

Finally, the heavy inter-weaving of all information into a single generation process in ADVISOR II would make it very tedious to extend the system to more than one language, since all generation work would have to be redone for each language. In contrast, MOOSE produces SemSpecs with a language-neutral mechanism and employs declarative, language-specific lexical resources and Upper Models (which both can also be extensively shared among languages).

GOSSiP A second approach that is concerned with paraphrases is the work of Iordanskaja, Kittredge and Polguère [1991]. They work in the theoretical framework of the Meaning-Text Model (MTM), which is designed as a linguistic theory of language production and, as an important aspect, of paraphrasing. We have sketched their framework in section 2.3.1; it emphasizes the syntactic processes in deriving paraphrases, for example those based on support verbs (e.g., *to use Emacs, to make use of Emacs*). This is a topic we have not addressed in our work, since our focus is on the interface between a conceptual representation and language-specific semantic representations. The MTM leaves the conceptual level aside and explicitly starts with a semantic representation. GOSSiP and MOOSE are concerned with different aspects of the overall problem and therefore difficult to compare.

10.2.5 Event verbalization

From the viewpoint of event verbalization, the SAGE system of Meteer [1994] has objectives similar to ours. Meteer is concerned with verbalizing descriptions from plans that are produced automatically by executing procedures that are associated with the specific goals that the system pursues. Whenever a planning procedure runs, it creates an instance of an event concept, which then serves as input to the generation module. The instances are mapped first to Meteer's level of *text structure* [Meteer 1992] and from there in several steps to linguistic utterances. The mapping from the domain concepts to the text structure level is performed by 'mapping tables' attached to the concepts. In MOOSE, on the other hand, we have stressed that the mapping is a central aspect of lexical semantics and thus defined lexicon entries so that they perform task, so that no extra instrument is needed. Moreover, Meteer's work is not concerned with other aspects of lexical choice, nor with multilinguality.

Within the framework of systemic-functional grammar, Wanner and Bateman [1991] follow an idea similar to our approach: they suggest building SPL expressions as input to PENMAN from a representation of a situation, as explained in section 2.3.1. They are interested in a

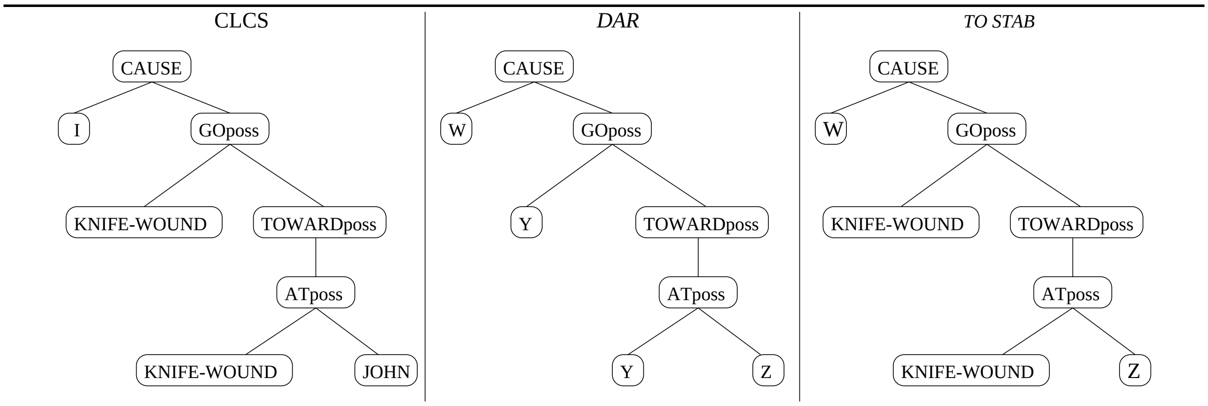

Figure 10.2: Sample CLCS and lexicon entries (abridged) from [Dorr 1993, pp. 224, 227]

variety of verbalizations similar to ours, but the level of their input expressions is not designed with the goal of their automatic production by a reasoning system in mind. The mapping from the situation representation to SPLs by means of system networks and lexical functions is quite complicated, and it is not clear how much of the process has actually been implemented. Besides, the approach does not deal with multilinguality.

10.2.6 Multilinguality and the lexicon

Multilinguality, as we have noted in chapter 1, has only very recently become a field of significant interest in NLG—in fact, TECHDOC was among the pioneers in this area. An older system is FoG [Goldberg, Driedger, Kittredge 1994], which generates English and French weather reports. Here, the lexical choices made for the two languages are almost exactly parallel, due to the nature of the domain and its fixed terminology.

Other research in multilingual generation has centered on developing Upper Models for other languages and finding ways of sharing portions of them among the target languages [Bateman et al. 1991, Bateman et al. 1994], but this work is not directly concerned with lexical matters.

Two projects on multilingual generation have recently started in Europe: DRAFTER [Delin et al. 1994] and GIST [Lavid 1995]. Their overall approaches are similar to that of TECHDOC (DRAFTER is even dealing with the same domain), but it is too early to compare their ways of handling multilingual lexicalization.

UNITRAN and PRINCITRAN In the field of interlingual machine translation, the work on UNITRAN [Dorr 1993] and its successor PRINCITRAN [Dorr and Voss forthcoming] is concerned with problems quite similar to ours. UNITRAN is an MT system specifically designed to handle a number of divergences found between English, Spanish, and German. The system uses an interlingua that is a linguistic rather than a conceptual level of representation and handles the divergences with syntactic mechanisms, based on language-specific parameters that are used in mapping between the interlingua and the languages. Therefore, the focus of the system is clearly different from that of MOOSE, but nonetheless we can draw a comparison of the techniques used.

The interlingua in UNITRAN is based on the lexical-conceptual structures (LCS) developed by Jackendoff [1990], parts of which we have used in modelling movement events. Dorr's version of the representation is a so-called CLCS, which is a tree whose nodes are either primitives that are linguistically relevant (such as CAUSE, WITH-instrument, or IN-location) or concepts representing entities of the world, corresponding to our domain model concepts. As an example, figure 10.2 shows on the left-hand side the CLCS representing the event of an agent ('I') causing knife wounds to move to a person named John, which in English can be expressed as *I stabbed John*.

Concentrating on the generation half of the overall MT problem, the task is to map a CLCS expression to the selected target language. UNITRAN does this by matching the lexical-conceptual structures associated with lexical entries (these are called RLCS) against the CLCS and then performing syntactic operations to combine them into a well-formed sentence. These operations we cannot examine here in detail; the crucial point is that the lexical RLCS can have language-specific parameters associated with some of its nodes, which will steer the syntactic construction into the proper direction.

Continuing the example shown in the figure, the English sentence *I stabbed John* does not have a straightforward Spanish translation, because this language lacks a simple verb corresponding to *to stab*. Instead, one uses the construction *Yo le di puñaladas a Juan* ('I gave knife-wounds to John'). This is the reason why the CLCS looks the way it does, and our approach in MOOSE would in fact use the same representation strategy: The target language that exhibits the most fine-grained decomposition of the event determines the grain-size of the underlying representation. Other languages, which use less fine-grained words for the same event can then pack several conceptual units into a single lexeme.

In producing the Spanish sentence, the RLCS of the verb *dar* ('to give') matches the CLCS. The RLCS is shown in the middle of figure 10.2, but the syntactic parameters have been omitted. The entry has a table associated with it that gives for every variable (here: W, Y, Z) the syntactic position of the corresponding constituent. The system then lexicalizes the nodes corresponding to the variables and performs the syntactic operations in accordance with the entries in the argument table of the verb. The Spanish preposition *a* covers the subtree rooted in TOWARDposs, and the corresponding phrase is added to the sentence. Similarly, the word *puñaladas* ('knife wounds') is inserted.

For the English sentence, *to stab* already incorporates the knife wounds, hence there are only two variables to be replaced, and the resulting sentence structure is simpler because *John* is expressed as direct object.

UNITRAN's procedure of traversing the verb's RLCS and recursively lexicalizing the arguments resembles the strategy of forming SemSpecs in MOOSE. But, once again, the levels of description are different—MOOSE first builds a SemSpec as a principled level of representation. Thus, to produce the sentence pair discussed, MOOSE would construct two different SemSpecs on the grounds of finding different lexical options in each target language, and the construction of the SemSpecs would proceed in much the same way as just described for UNITRAN:

```
(x1 / directed-action :lex stab_el
    :actor (speaker / person)
    :actee (x2 / person :name John))

(x1 / directed-action :lex dar_sp
    :actor (speaker / person)
    :actee (x2 / object :lex punalada_sp)
    :destination (x3 / person :name juan))
```

The surface generators, if we had one for Spanish, would produce the target sentences from these representations.

In UNITRAN, there is no difference between lexical semantics and pre-linguistic knowledge, because there is no background knowledge base in which word meaning is grounded. Accordingly, the system cannot recognize the (situation-specific) equivalence of sentence pairs like *Disconnect the wire / Ziehen Sie das Kabel ab* or *Remove the cap from the tank / Open the tank*. UNITRAN's strength is in handling syntactic divergences, but it does not deal at all with the interface between lexical semantics and background knowledge. As a consequence, the lexical entries for a particular language can be heavily determined by the other languages dealt with by the system. For example, the English entry for *to stab* looks the way it does only because of the divergence with the Spanish translation. In case there was another target language involved, which used yet another structure for *to stab* (such as *I pinched holes into John with a knife*), then designing appropriate representations for all languages would become very problematic.

Due to the approach taken towards the MT task, paraphrasing is not a matter of concern in UNITRAN: For the most part, it suffices to produce one appropriate rendering of the sentence in the target language. In generation, on the other hand, the ability to tailor text to different readers is an important goal, and thus the task of paraphrasing and fine-grained word choice is significant in MOOSE.

In later work, Dorr and Voss [forthcoming] propose a multi-level approach to MT, consisting of syntactic representations for source language and target language, a common interlingual structure similar to the CLCS, and a separate knowledge-base level, which is designed to help in parsing and building the CLCS and to ground the primitives used in the CLCS. However, the link between the interlingua and the KB is sketched only vaguely. As a specific difference to our approach, there is no account of event structure on the KB level; instead, the difference between resultative and durative verbs (in our terminology) is encoded with primitives in the IL, in a Jackendoffian manner. Thus, it seems very difficult to employ any reasoning about events, or to systematically derive verb configurations, as our alternation and extension rules do. However, from the commonly taken perspective on MT, only limited reasoning and derivation capabilities are necessary, so again the different purpose is responsible for a different design.

10.3 Contributions of the thesis

10.3.1 Lexical semantics for NLG

Previous approaches to NLG have linked words as syntactic objects directly to units of the conceptual input. Accordingly, lexical information typically consisted of syntactic features and a conceptual denotation only. In contrast, we have developed a rich representation of word meaning that decomposes the information into separate parts, each of which is useful for one or more purposes.

Separating the denotation from the connotations of a word [DiMarco, Hirst, Stede 1993] is instrumental for word choice on purely stylistic grounds, where words refer to the same object or event, but differ in tone. The denotation is the link to the reasoning system and reflects the distinctions needed there; but it also reflects distinctions that are linguistically relevant (in any of the target languages). Specifically, we have represented the internals of *event structure* (e.g., [Pustejovsky 1991a]), following the principle of introducing a feature whenever it is relevant for the semantics of a target language (e.g., [Jackendoff 1990]), or for a lexical difference between languages (e.g., [Talmy 1985]). Since the event structure is represented at the 'deep' level of representation, its mapping to linguistic representations can account for variation in emphasizing different aspects (e.g., [Kunze 1991]) and in making elements of the situation more or less prominent (e.g., [Talmy 1988]).

By factoring out the contribution that a word makes to sentence meaning (PSemSpec), we create the possibility that a word can denote different (but related) things yet contribute a constant partial SemSpec to the sentence; hence, semantic *type shifts* are possible (e.g., [Bierwisch 1983]). Also, we have been able to specify lexical rules that implement *verb alternations* (e.g., [Levin 1993]) by simultaneously rewriting the denotation and PSemSpec of a verb. These rules operate on the basis of the *Aktionsart* of a verb configuration, for which we have given a definition in terms of the denotation. Furthermore, the separation of denotation, covering-list, and PSemSpec has allowed us to specify *defaults* in word meaning in a clear fashion: a word can incorporate a semantic unit if a particular value is present; otherwise, it is not incorporated and needs to be expressed separately.

The separation of the different parts of lexical meaning facilitates the task of specifying in detail the similarities and differences between words—within a language and across languages. Words can share the same denotation and differ only in other respects, for instance their argument linking (reflected in the PSemSpec) or their connotations. Apart from providing descriptive instruments for comparing words, our account of lexical semantics also invites investigations of efficiently storing monolingual and multilingual dictionary information by sharing parts of lexical entries wherever possible.

10.3.2 System architecture for NLG

We have proposed a new system architecture for NLG that is based on two levels of representation, SitSpec and SemSpec. Chapter 6 introduced them and argued that they are both useful levels of abstraction for generation purposes. (The level of SitSpec was developed in this work, while SemSpec is an adaptation of a level used by a previous sentence generator (PENMAN), but some important modifications were made to its precise role.) SitSpecs are instantiations of *domain knowledge*, and we have in chapter 5 developed a domain model that suits the purposes of reasoning on the one hand, and multilingual generation on the other hand.

An explicit transition between these two levels of representation has been specified on the basis of declarative representations. In this transition, the lexical entries of a particular target natural language serve as the bridge between the two levels, and in contrast to previous approaches, they map the input representation not directly to a syntactic specification, but first to a lexicalized sentence-semantic structure. On this level it is possible to capture certain semantic generalizations, as we have shown with a set of rules that derive some important verb alternations.

Using the transition from SitSpec to SemSpec, our system can produce, on the basis of well-defined declarative representations, a wider range of paraphrases than previous systems. Because in our system word meaning is grounded in the background KB, we can produce paraphrases that require *inference* capabilities and are specific to particular contexts (e.g., *open the tank, remove the cap from the tank*). In previous systems, this grounding was not provided, and they were thus limited to paraphrases based on purely lexical or syntactic knowledge.

While the problem of then actually choosing the 'best' paraphrase in a specific situation of utterance has not been resolved here, we have accounted for the integration of choice parameters. Specifically, we have dealt with salience assignment and connotations, and our system responds flexibly to different target specifications of these features, by making appropriate lexical choices or selecting a verb alternation that accommodates the needs. The system is designed in such a way that further choice parameters can be added.

In contrast to previous approaches to NLG, our system lends itself to multilingual generation: at the beginning of the generation process, the lexicon for the target language is selected and thus serves as a specific “semantic looking glass” for the underlying SitSpec; the lexicon drives the production of language-specific SemSpecs. And then, language-specific generation grammars are equipped with the syntactic knowledge to produce well-formed sentences.

10.3.3 Implementation

The architecture described has been implemented as a prototype system, MOOSE, which produces English and German sentences from SitSpecs in a sample domain of containers and liquids, a sub-domain of that used in the TECHDOC system (which has to do with automobile maintenance). The domain model was developed to allow for reasoning with the representations on the one hand, and for providing the SitSpecs for the generator on the other. The implementation uses LOOM, a popular knowledge representation language, so that our generator can be interfaced to practical applications.

The protoype has a user interface that gives support for building a SitSpec from domain model concepts and then provides a graphical representation of the SitSpec on the screen. The user can then set the target specifications for the generation process by mouse-click, such as specifying SitSpec nodes to be placed in the foreground or in the background, or making them optional for the verbalization. Similarly, the target values for connotational features can be set. Thus it is possible to generate variants of sentences interactively and to experiment with the effects of the parameter settings.

For moving beyond the experimental prototype, we have sketched how the system can be integrated into a larger application such as TECHDOC. TECHDOC has so far used the same semantic representation as input for the language-specific sentence generation modules. In the sample domain of instruction manuals, this is often not problematic, but is too simplistic in the cases of divergences such as those described in chapter 4. Thus, TECHDOC was not able to appropriately generate sentence pairs like *Disconnect the spark plug wire / Das Zündkabel abziehen* or *Twist the cap until it stops / Drehen Sie den Deckel bis zum Anschlag*; after supplementing it with the mechanisms developed in this thesis, sentences of that kind will be produced. Similarly, TECHDOC will benefit from other paraphrasing capabilities such as choosing a verb according to language-specific conventions: *to remove / herausziehen, abziehen*, etc.

10.4 Directions for future research

Having summarized what the thesis has achieved, we now list some of the issues that it has left unresolved because they are beyond the scope of the present work, and which would be very interesting to explore next.

The computational search problem Both the matching procedure for finding verbalization options and the search procedure for constructing the SemSpec have been implemented only in prototype, simply to demonstrate the functionality of the approach. Now, both algorithms should be optimized by employing additional knowledge about the search space. In SemSpec construction, for example, it is important not to grow the pool of verbalization options more than necessary. Words that definitely have the wrong connotations should be ignored right away and not, for instance, trigger further alternation rules to be applied to them. Similarly, before applying an alternation at all, it could be estimated whether its result will be useful.

Building SitSpecs automatically SitSpecs were defined with the goal in mind of ultimately having them produced automatically from higher-level plans. Some first experiments indicate that LOOM production rules are an adequate instrument for expanding abstract descriptions, e.g., the basic steps of an oil change, into more fine-grained specifications. When the knowledge is carefully distributed across the domain concept taxonomy, a generic 'exchange-contents-of-container' plan can be automatically expanded to a specific and detailed sequence of SitSpecs, which describes the actions as they pertain to some particular tank. That is, an action like opening a tank has the same pre-conditions and post-conditions for any tank, but its precise execution depends on the specific type; the knowledge about this type would rest with the specific tank-concept in the taxonomy. In LOOM, production rules are an integral part of the overall reasoning system, and hence they are sensitive to the subsumption relationships.

Choice criteria The aim of this thesis was to make a wide range of lexical paraphrases available to a sentence generator, and we have stressed that the actual criteria for choosing the most appropriate paraphrase in an utterance situation is a matter of future research. In section 2.3 we have reviewed work on such choice criteria. Our system so far accounts only for salience and connotations, but the architecture is flexible enough to have additional choice factors integrated at the stages of preference evaluation or PSemSpec unification.

Aspectual composition We have begun to explore the possible variety of event verbalizations by representing the events not as single predicates but as entities composed of various parts. The next steps here are to work systematically on the question of 'aspectual composition' (or 'Aktionsart projection', as we have called it). Specifically, when temporal information is integrated into the representation of situations, the different ways of verbalizing it depend on the overall Aktionsart, as it is projected from the verb to the clause. Dorr and Gaasterland [1995] present an approach of generating event descriptions, including appropriate connectives, from time-stamped representations. The same line can be followed in our framework by adding temporal attributes to some or all of the three elements of an event and then providing rules for systematically verbalizing this information in tandem with PSemSpec unification.

Lexical variety and syntactic variety We have focused on lexical paraphrases here and said relatively little about syntactic variation, both within and between languages. While the surface generation grammars are in charge of syntactic decisions, and thus are outside the immediate realm of this thesis, the interactions between lexical and syntactic decisions need to be given more attention. Obviously, the machine translation literature is relevant here, as well as work in contrastive linguistics. As an example for the languages we have been dealing with, Mehl [1995] analyzes the various ways of rendering English gerund constructions in German.

Lexis in systemic linguistics This thesis project started with the assumption of using the PENMAN generator and working in the framework of systemic-functional linguistics. However, central weaknesses on the lexical side of PENMAN were noted, and so we developed a generation approach that performs many generation decisions before PENMAN is activated, i.e., before entering the systemic grammar. The work is thus not really dependent on PENMAN any more but is relatively neutral with respect to the front-end generator. Correspondingly, the work is at this stage not directly tied to systemic linguistics. An interesting next step on the theoretical side is evaluating the results of this thesis from the perspective of systemics; also, the mechanisms of PENMAN need to be scrutinized for improving its lexical capabilities. That will involve clarifying and most probably modifying the role of the lexical information in the systemic lexicogrammar.

Lexical inheritance One source of lexical variation explored in this work was the choice between specific and general lexemes—for nouns and verbs alike. As it happens, opening up this door also opens a new research problem: where to *stop* lexical inheritance. For example, while it is in general necessary and useful to have the choice between *opening the bottle* and *removing the cork from the bottle*, one would not talk about *disconnecting the cork from the bottle*. NLG has traditionally eschewed these issues by considering only the most specific lexeme at any point. But Reiter's [1990] research on basic-level categories has shown that the most specific term is not always the best; sometimes it is not even acceptable. And our examples of situation-dependent paraphrases have demonstrated the need for lexical inheritance, too. However, one now needs good rules for barring general lexemes from percolating down the taxonomy too far, and/or for preferring lexemes of appropriate specificity.

The 'generative lexicon' The last point relates to the more general issue of organizing the lexicon intelligently. We have indicated that parts of lexical entries can often be shared among similar lexemes (e.g., lots of synonyms will have exactly the same denotation—and the matching phase of the generator should match these only once), and these observations should lead to a taxonomic organization with shared representations. Pustejovsky [1991b] makes proposals in this direction when advocating his 'generative lexicon', which, as pointed out earlier, aims at spreading the semantic load of the lexicon more evenly between verbs and other parts of speech. Our approach is so far heavily verb-centered and needs to be extended towards accounting, inter alia, for the semantics of nouns more thoroughly. Pustejovsky's research as well as Bierwisch's [1983] ideas, which were illustrated in section 3.5, can provide fruitful insights here.

Bibliography

[Appelt 1985] D. Appelt. *Planning natural language utterances.* Cambridge, UK: Cambridge University Press, 1985.

[Bach 1986] E. Bach. "The algebra of events." In: *Linguistics and Philosophy* 9, 1986

[Bateman et al. 1990] J. Bateman, R. Kasper, J. Moore, R. Whitney. "A general organization of knowledge for natural language processing: the Penman upper model." Technical report, USC Information Sciences Institute, 1990.

[Bateman et al. 1991] J. Bateman, C. Matthiessen, K. Nanri, L. Zeng. "The re-use of linguistic resources across languages in multilingual generation components." In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-91).* Sydney, 1991.

[Bateman et al. 1994] J. Bateman, B. Magnini, F. Rinaldi. "The generalized {Italian, German, English} Upper Model." Workshop on ontologies, European Conference on AI (ECAI), Amsterdam, 1994.

[Bennett et al. 1990] W. Bennett, T. Herlick, K. Hoyt, J. Liro, A. Santisteban. "A computational model of aspect and verb semantics." In: *Machine Translation* 4 (4), 1990.

[Bierwisch 1983] M. Bierwisch. "Semantische und konzeptuelle Repräsentation lexikalischer Einheiten." In: R. Růžička, W. Motsch (eds.): *Untersuchungen zur Semantik* (studia grammatica XXII). Berlin: Akademie-Verlag, 1983.

[Brachman and Schmolze 1985] R. Brachman, J. Schmolze. "An overview of the KL-ONE knowledge representation system." In: *Cognitive Science* 9 (2), 1985.

[Buchberger and Horacek 1988] E. Buchberger, H. Horacek. "VIE-GEN: A generator for German texts." In: D.D. McDonald, L. Bolc (eds.): *Natural language generation systems.* New York: Springer, 1988.

[Busemann 1993] S. Busemann. "A holistic view of lexical choice." In: H. Horacek, M. Zock (eds.): *New concepts in natural language generation.* London: Pinter, 1993.

[Bussmann 1983] H. Bussmann. *Lexikon der Sprachwissenschaft.* Stuttgart: Kröner, 1983.

[Carlson 1981] L. Carlson. "Aspect and quantification." In: *Syntax and Semantics* 14, 1981.

[Carter 1987] R. Carter. *Vocabulary: Applied linguistic perspectives.* London: Allan & Unwin, 1987.

[Collins 1987] *The new Collins dictionary and thesaurus.* London/Glasgow: Collins, 1987.

[Cruse 1986] D.A. Cruse. *Lexical semantics.* Cambridge, UK: Cambridge University Press, 1986.

[Crystal and Davy 1969] D. Crystal, D. Davy. *Investigating english style.* London: Edward Arnold, 1969.

[Cumming 1986] S. Cumming. "The lexicon in text generation." Technical report ISI-RR-86-168, USC Information Sciences Institute, 1986.

[Dale 1989] R. Dale. "Cooking up referring expressions." In: *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics (ACL).* Vancouver, 1989.

[Dale 1992] R. Dale. *Generating referring expressions.* Cambridge, MA: MIT Press, 1992.

[Danlos 1987] L. Danlos. *The linguistic basis of text generation.* Cambridge, UK: Cambridge University Press, 1987.

[Delin et al. 1994] J. Delin, A. Hartley, C. Paris, D. Scott, K. Vander Linden. "Expressing procedural relationships in multilingual instructions." In: *Proceedings of the Seventh International Workshop on Natural Language Generation.* Kennebunkport, Maine, 1994.

[DiMarco and Hirst 1993] C. DiMarco, G. Hirst. "Usage notes as a representation of lexical near-equivalence for lexical choice." In: *Proceedings of the 9th Annual Conference of the University of Waterloo Centre for the New OED and Text Research.* 1993.

[DiMarco et al. 1993] C. DiMarco, G. Hirst, M. Stede. "The semantic and stylistic differentiation of synonyms and near-synonyms." In: Working notes of the AAAI Spring Symposium on Building Lexicons for Machine Translation. Stanford University, March 1993.

[Dorr 1993] B.J. Dorr. "Interlingual machine translation: a parameterized approach". In: *Artificial Intelligence* 63, 1993.

[Dorr and Gaasterland 1995] B.J. Dorr, T. Gaasterland. "Selecting tense, aspect, and connecting words in language generation." In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI).* Montréal, 1995.

[Dorr and Voss forthcoming] B.J. Dorr, C.R. Voss. "A multi-level approach to interlingual MT: Defining the interface between representational languages." Submitted.

[Dowty 1979] D. Dowty. *Word meaning and Montague grammar.* Dordrecht: Reidel, 1979.

[Elhadad 1991] M. Elhadad. "Generating adjectives to express the speaker's argumentative intent." In: *Proceedings of the Fifth National Conference on Artificial Intelligence (AAAI)*, 1991.

[Elhadad 1993] M. Elhadad. "Using argumentation to control lexical choice: a functional unification implementation." Ph.D. thesis, Columbia University, 1993.

[Elhadad and McKeown 1990] M. Elhadad, K.R. McKeown. "Generating connectives." In: *Proceedings of the 13th International Conference on Computational Linguistics (COLING).* Helsinki, 1990.

[Emele et al. 1992] M. Emele, U. Heid, S. Momma, R. Zajac. "Interactions between linguistic constraints: procedural vs. declarative approaches." In: *Machine Translation* 7 (1-2), 1992.

[Evens 1988] M.W. Evens. *Relational models of the lexicon.* Cambridge, UK: Cambridge University Press, 1988.

[Fiedler and Huang 1995] A. Fiedler, X. Huang. "Aggregation in the generation of argumentative texts." In: *Proceedings of the Fifth European Workshop on Natural Language Generation.* Leiden, Netherlands, 1995.

[Fillmore 1968] C.J. Fillmore. "The case for case." In: E. Bach, R.T. Harms (eds.): *Universals in linguistic theory.* New York: Holt, Rinehart and Winston, 1968.

[Fillmore et al. 1988] C.J. Fillmore, P. Kay, M.C. O'Connor. "Regularity and idiomaticity in grammatical constructions: The case of *let alone.*" In: *Language* 64 (3), 1988.

[Garza-Cuarón 1991] B. Garza-Cuarón. *Connotation and Meaning.* Berlin: Mouton de Gruyter, 1991.

[Goldberg et al. 1994] E. Goldberg, N. Driedger, R. Kittredge. "Using natural-language processing to produce weather forecasts." In: *IEEE EXPERT* April 1994.

[Goldman 1975] N.M. Goldman. "Conceptual generation." In: R.C. Schank (ed.): *Conceptual information processing.* Amsterdam: North-Holland, 1975.

[Granville 1984] R. Granville. "Controlling lexical substitution in computer text generation." In: *Proceedings of the 10th International Conference on Computational Linguistics (COLING).* Stanford, 1984.

[Grote 1993] B. Grote. "Generierung in der systemisch-funktionalen Grammatik: Die Behandlung von Präpositionen." Master's thesis, Universität Trier, 1993.

[Grote et al. 1995] B. Grote, N. Lenke, M. Stede. "Ma(r)king concessions in English and German." In: *Proceedings of the Fifth European Workshop on Natural Language Generation.* Leiden, Netherlands, 1995.

[Gruber 1965] J.S. Gruber. "Studies in lexical relations." Ph.D. thesis, MIT, Cambridge, MA, 1965.

[Halliday 1985] M.A. Halliday. *Introduction to functional grammar.* Edward Arnold, London, 1985.

[Halliday and Matthiessen forthcoming] M.A. Halliday, C. Matthiessen. *Construing experience through meaning: a language-based approach to cognition.* Berlin: de Gruyter, forthcoming.

[Harnad 1990] S. Harnad. *Categorial perception.* Cambridge UK: Cambridge University Press, 1990.

[Hawkins 1986] J.A. Hawkins. *A comparative typology of English and German.* London/Sydney: Croom Helm, 1986.

[Healey 1968] A. Healey. "English idioms." In: *Kivung* 1 (2), 1968.

[Heeman and Hirst 1995] P. Heeman, G. Hirst. "Collaborating on referring expressions." In: *Computational Linguistics* 21 (3), 1995.

[Helbig and Buscha 1991] G. Helbig, J. Buscha. *Deutsche Grammatik: Ein Handbuch für den Ausländerunterricht.* Berlin: Langenscheidt, Verlag Enzyklopädie, 1991.

[Helbig and Schenkel 1973] G. Helbig, W. Schenkel. *Wörterbuch zur Valenz und Distribution deutscher Verben.* VEB Verlag Enzyklopädie, Leipzig, 1973.

[Henschel 1993] R. Henschel. "Merging the English and the German Upper Model." Technical Report, GMD/Institut für integrierte Publikations- und Informationssysteme, Darmstadt, 1993.

[Hirst 1995] G. Hirst. "Near-synonymy and the structure of lexical knowledge." In: Working notes of the AAAI Spring Symposium on Representation and Acquisition of Lexical Knowledge. Stanford University, 1995.

[Horacek 1990a] H. Horacek. "The architecture of a generation component in a complete natural language dialogue system." In: R. Dale, C. Mellish, M. Zock (eds.): *Current research in natural language generation.* London: Academic Press, 1990.

[Horacek 1990b] H. Horacek. "Some useful search techniques for natural language generation." In: H. Marburger (ed.): *(Proceedings of the German workshop on artificial intelligence (GWAI-90).* Berlin/Heidelberg: Springer, 1990.

[Hovy 1988a] E.H. Hovy. *Generating natural language under pragmatic constraints.* Hillsdale, NJ: Lawrence Erlbaum, 1988.

[Hovy 1988b] E.H. Hovy. "Generating language with a phrasal lexicon." In: D.D. McDonald, L. Bolc (eds.): *Natural language generation systems.* New York: Springer, 1988.

[Hovy and Nirenburg 1992] E. Hovy and S. Nirenburg. "Approximating an interlingua in a principled way." In: *Proceedings of the DARPA Speech and Natural Language Workshop.* Hawthorne, NY, 1992.

[Iordanskaja et al. 1991] L. Iordanskaja, R. Kittredge, A. Polguère. "Lexical selection and paraphrase in a Meaning-Text generation model." In: C.L. Paris, W.R. Swartout, W.C. Mann (eds.): *Natural language generation in artificial intelligence and computational linguistics.* Dordrecht: Kluwer, 1991.

[Jackendoff 1983] R. Jackendoff. *Semantics and Cognition.* Cambridge, MA: MIT Press, 1983.

[Jackendoff 1990] R. Jackendoff. *Semantic structures.* Cambridge, MA: MIT Press, 1990.

[Jackendoff 1993] R. Jackendoff. "Parts and boundaries." In: *Cognition* 41, 1991.

[Jacobs 1985] P.S. Jacobs. "PHRED: A generator for natural language interfaces." In: *Computational Linguistics* 11 (4), 1985.

[Jacobs 1987] P.S. Jacobs. "Knowledge-intensive natural language generation." In: *Artificial Intelligence* 33, 1987.

[James 1980] C. James. *Contrastive analysis.* London: Longman, 1980.

[Kameyama et al. 1991] M. Kameyama, R. Ochitani, S. Peters. "Resolving translation mismatches with information flow." In: *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics (ACL).* Berkeley, 1991.

[Kantrowitz and Bates 1992] M. Kantrowitz, J. Bates. "Integrated natural language generation systems." In: R. Dale, E. Hovy, D. Rösner, O. Stock (eds.): *Aspects of automated natural language generation – Proceedings of the 6th International Workshop on Natural Language Generation.* Berlin/Heidelberg: Springer, 1992.

[Kasper 1989] R. Kasper. "A flexible interface for linking applications to Penman's sentence generator." In: *Proceedings of the DARPA Workshop on Speech and Natural Language Processing.* University of Pennsylvania, 1989.

[Katz and Fodor 1963] J.J. Katz, J.A. Fodor. "The structure of a semantic theory." In: *Language* 39, 1963.

[Kittredge et al. 1988] R. Kittredge, L. Iordanskaja, A. Polguère. "Multilingual text generation and the Meaning–Text Theory." In: *Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages.* Pittsburgh, 1988.

[Kukich 1983] K. Kukich. "Design and implementation of a knowledge-based report generator." In: *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics (ACL),* 1983.

[Kunze 1987] J. Kunze. "Some remarks on case relations." In: *Proceedings of the Third Conference of the European Chapter of the Association for Computational Linguistics (EACL).* Copenhagen, 1987.

[Kunze 1991] J. Kunze. *Kasusrelationen und semantische Emphase* (studia grammatica XXXI). Berlin: Akademie Verlag, 1991.

[Lavid 1995] J. Lavid. "From interpersonal options to thematic realization in multilingual administrative forms." In: Working notes of the IJCAI workshop on multilingual text generation. Montréal, 1995.

[Lenke 1994] N. Lenke. "Paraphrasen — Lösungen für antizipierte Leserprobleme bei der automatischen Textgenerierung." Doctoral dissertation, Universität Duisburg, 1994.

[Levin 1993] B. Levin. *English verb classes and alternations.* Chicago: University of Chicago Press, 1993.

[Lyons 1977] J. Lyons. *Semantics.* Cambridge, UK: Cambridge University Press, 1977.

[MacGregor and Bates 1987] R. MacGregor, R. Bates. "The loom knowledge representation language." Technical Report ISI/RS-87-188, USC Information Sciences Institute, 1987.

[Marcus 1987] M. Marcus. "Generation systems should choose their words". In: Y. Wilks (ed.): *Theoretical issues in natural language processing.* Las Cruces: New Mexico State Univ., 1987.

[Matthiessen 1991] C. Matthiessen. "Lexico(grammatical) choice in text generation." In: C.L. Paris, W.R. Swartout, W.C. Mann (eds.): *Natural language generation in artificial intelligence and computational linguistics.* Dordrecht: Kluwer, 1991.

[Matthiessen and Bateman 1991] C. Matthiessen, J. Bateman. *Text generation and systemic functional linguistics: experiences from English and Japanese.* London: Pinter, 1991.

[McDonald 1991] D.D. McDonald. "On the place of words in the generation process." In: C.L. Paris, W.R. Swartout, W.C. Mann (eds.): *Natural language generation in artificial intelligence and computational linguistics.* Dordrecht: Kluwer, 1991.

[McKeown et al. 1990] K.R. McKeown, M. Elhadad, Y. Fukumoto, J. Lim, C. Lombardi, J. Robin, F. Smadja. "Natural language generation in COMET." In: R. Dale, C. Mellish, M. Zock (eds.): *Current research in natural language generation.* London: Academic Press, 1990.

[McKeown et al. 1993] K.R. McKeown, J. Robin, M. Tanenblatt. "Tailoring lexical choice to the user's vocabulary in multimedia explanation generation." In: *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics (ACL).* Columbus, OH, 1993.

[Mehl 1994] S. Mehl. "Synonyme bei der Erzeugung, Generierung und Produktion natürlicher Sprache." In: *Proceedings of the 2nd German conference on natural language processing (KONVENS).* Vienna, 1994.

[Mehl 1995] S. Mehl. "Interaction between syntax and semantics: the case of gerund translation." In: *Proceedings of the Fifth European Workshop on Natural Language Generation.* Leiden, Netherlands, 1995.

[Mel'čuk 1988] I.A. Mel'čuk. *Dependency syntax: theory and practice.* Albany, NY: State University of New York Press, 1988.

[Mellish and Evans 1989] C. Mellish, R. Evans. "Natural language generation from plans." In: *Computational Linguistics* 15 (4), 1989.

[Meteer 1992] M. Meteer. *Expressibility and the problem of efficient text planning.* London: Pinter, 1992.

[Meteer 1994] M. Meteer. "Generating event descriptions with SAGE: a simulation and generation environment." In: *Proceedings of the Seventh International Workshop on Natural Language Generation.* Kennebunkport, Maine, 1994.

[Miezitis 1988] M.A. Miezitis. "Generating lexical options by matching in a knowledge base." Technical Report CSRI-217, University of Toronto, 1988.

[Miller 1991] G.A. Miller. *The science of words.* New York: Scientific American Library, 1991.

[Moens and Steedman 1988] M. Moens, M. Steedman. "Temporal ontology and temporal reference." *Computational Linguistics* 14 (2), 1988.

[Naess 1975] A. Naess. *Kommunikation und Argumentation.* Kronberg, 1975.

[Nirenburg and L. Levin 1992] S. Nirenburg, L. Levin. "Syntax-driven and ontology-driven lexical semantics." In: J. Pustejovsky, S. Bergler (eds.): *Lexical semantics and knowledge representation.* Berlin/Heidelberg: Springer, 1992.

[Nirenburg and Nirenburg 1988] S. Nirenburg, I. Nirenburg. "A framework for lexical selection in natural language generation." In: *Proceedings of the 12th International Conference on Computational Linguistics (COLING).* Budapest, 1988.

[Nogier and Zock 1992] J.F. Nogier, M. Zock. "Lexical choice by pattern matching". In: *Knowledge Based Systems* 5 (3), 1992.

[Novak 1988] H.-J. Novak. "Generating referring phrases in a dynamic environment." In: M. Zock, G. Sabah (eds.): *Advances in natural language generation Vol. 2.* London: Pinter, 1988

[Novak 1991] H.-J. Novak. "Integrating a generation component into a natural language understanding system." In: O. Herzog, C.-R. Rollinger (eds.): *Text understanding in LILOG.* Berlin/Heidelberg: Springer, 1991.

[Novak 1993] H.-J. Novak. "Ontology and lexical choice." In: H. Horacek, M. Zock (eds.): *New concepts in natural language generation.* London: Pinter 1993.

[*OALD* 1989] *Oxford advanced learner's dictionary of current English.* Oxford: Oxford University Press, 1989 (4th edition).

[Osgood 1980] C.E. Osgood. *Lectures on language performance.* New York: Springer, 1980.

[Parsons 1990] T. Parsons. *Events in the semantics of English: A study in subatomic semantics.* Cambridge, MA: MIT Press, 1990.

[Pattabhiraman and Cercone 1991] T. Pattabhiraman, N. Cercone. "Salience in natural language generation." In: *Proceedings of the IJCAI workshop on decision making throughout the generation process.* Sidney, 1993.

[Penman 1989] "The Penman Documentation: the Primer, the User Guide, the Reference Manual, the NIGEL Manual." Unpublished Documentation of the Penman Sentence Generation System, USC Information Sciences Institute, 1989.

[Pinker 1989] S. Pinker. *Learnability and cognition: the acquisition of argument structure.* Cambridge, MA: MIT Press, 1989.

[Pustejovsky 1991a] J. Pustejovsky. "The syntax of event structure." In: *Cognition* 41, 1991.

[Pustejovsky 1991b] J. Pustejovsky. "The generative lexicon." In: *Computational Linguistics* 17 (4), 1991.

[Pustejovsky and Nirenburg 1987] J. Pustejovsky, S. Nirenburg. "Lexical selection in the process of language generation." In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics (ACL).* Stanford, 1987.

[Quirk et al. 1972] R. Quirk, S. Greenbaum, G. Leech, J. Svartvik. *A Grammar of Contemporary English.* Harlow: Longman, 1992 (20th edition).

[Rappaport and B. Levin 1988] M. Rappaport, B. Levin. "What to do with theta-roles." In: W. Wilkins (ed.): *Syntax and semantics 21: thematic relations.* New York: Academic Press, 1988.

[Reiter 1990] E. Reiter. "Generating descriptions that exploit a user's domain knowledge." In: R. Dale, C. Mellish, M. Zock (eds.): *Current research in natural language generation.* London: Academic Press, 1990.

[Reiter et al. 1995] E. Reiter, C. Mellish, J. Levine. "Automatic generation of technical documentation." In: *Applied Artificial Intelligence* 9, 1995.

[Reithinger 1992] N. Reithinger. *Eine parallele Architektur zur inkrementellen Generierung multimodaler Dialogbeiträge.* St. Augustin: infix, 1992.

[Robin 1990] J. Robin. "Lexical choice in natural language generation." Technical report CUCS-040-90, Columbia University, 1990.

[Rosch 1978] E. Rosch. "Principles of categorization." In: E. Rosch, B. Lloyd (eds.): *Cognition and categorization.* Hilldale, NJ: Lawrence Erlbaum, 1978.

[Rösner and Stede 1991] D. Rösner, M. Stede. "Zur Rolle von Wissensrepräsentation und Textstruktur bei der automatischen Generierung technischer Dokumente". Technical report 91029, FAW Ulm, 1991.

[Rösner and Stede 1992] D. Rösner, M. Stede. "TECHDOC: A system for the automatic production of multilingual technical documents". In: G. Görz (ed.): *KONVENS 92 – Proceedings of the First German Conference on Natural Language Processing.* Berlin/Heidelberg: Springer, 1992.

[Rösner and Stede 1994] D. Rösner, M. Stede. "Generating multilingual documents from a knowledge base: The TECHDOC project." In: *Proceedings of the International Conference on Computational Linguistics (COLING).* Kyoto, 1994.

[Sanders 1973] W. Sanders. *Linguistische Stiltheorie.* Göttingen: Vandenhoeck & Ruprecht, 1973.

[Saussure 1915/1966] F. de Saussure. *Course in general linguistics.* C. Bally, A. Sechehaye (eds.) New York: McGraw Hill, 1966.

[Schank 1975] R.C. Schank. *Conceptual information processing.* New York: Elsevier-North Holland, 1975.

[Schmitt 1986] P. Schmitt. "Die 'Eindeutigkeit' von Fachtexten: Bemerkungen zu einer Fiktion." In: M. Snell-Hornby (ed.): *Übersetzungswissenschaft – eine Neuorientierung.* Tübingen: Francke, 1986.

[Schwarze 1979] C. Schwarze. "Réparer–reparieren: a contrastive study." In: R. Bäuerle, U. Egli, A. von Stechow (eds.): *Semantics from different points of view.* Berlin/Heidelberg: Springer, 1979.

[Somers 1987] H. Somers. *Valency and case in computational linguistics.* Edinburgh: Edinburgh University Press, 1987.

[Sowa 1984] J. F. Sowa. *Conceptual structures: information processing in mind and machine.* New York: Addison-Wesley, 1984.

[Stede 1993] M. Stede. "Lexical choice criteria in language generation." In: *Proceedings of the 6th Conference of the European Chapter of the ACL (EACL).* Utrecht, 1993.

[Stede 1994] M. Stede. "A contrastive analysis of some contrastive discourse markers." In: B. Schmitz, J. Quantz (eds.): Ambiguität als Universalie. Proceedings of a workshop held at the Annual Conference of the DGfS. Technical report, TU Berlin, FB Informatik, 1994.

[Stede 1995] M. Stede. "Lexicalization in natural language generation: A survey." In: *Artificial Intelligence Review* 8, 1995.

[Stede and Grote 1995] M. Stede, B. Grote. "The lexicon: Bridge between language-neutral and language-specific representations". In: Working notes of the IJCAI workshop on multilingual text generation. Montréal, 1995.

[Stede and Weis 1993] M. Stede, U. Weis. "Kontrastive Untersuchung sprachlicher Phänomene in dreisprachigen Handbuchtexten." In: B. Grote, D. Rösner, M. Stede, U. Weis: From knowledge to language – three papers on multilingual text generation. Technical report 93017, FAW Ulm, 1993.

[Storrer 1992] A. Storrer. *Verbvalenz.* Tübingen: Niemeyer, 1992.

[Talmy 1985] L. Talmy. "Lexicalization patterns: semantic structure in lexical forms." In: T. Shopen (ed.): *Language typology and syntactic description 3: grammatical categories and the lexicon.* Cambridge, UK: Cambridge University Press, 1985.

[Talmy 1988] L. Talmy. "The relation of grammar to cognition." In: B. Rudzka-Ostyn (ed.): *Topics in Cognitive Linguistics.* Amsterdam: John Benjamins, 1988.

[Tesnière 1959] L. Tessniére. *Èlèments de syntaxe structurale.* Paris: Klinsieck, 1959.

[Trier 1931] J. Trier. *Der deutsche Wortschatz im Sinnbezirk des Verstandes.* Heidelberg, 1931.

[Vander Linden and Scott 1995] K. Vander Linden, D. Scott. "Raising the interlingual ceiling with multilingual text generation." In: Working notes of the IJCAI-workshop on multilingual text generation, Montréal, 1995.

[Vendler 1967] Z. Vendler. *Linguistics and Philosophy.* Ithaca, NY: Cornell University Press, 1967.

[Wanner 1994] L. Wanner. "On lexically biased discourse organization in text generation." In: *Proceedings of the International Conference on Computational Linguistics (COLING).* Kyoto, 1994.

[Wanner and Bateman 1990] L. Wanner, J. Bateman. "A collocational based approach to salience-sensitive lexical selection." In: *Proceedings of the Fifth International Workshop on Natural Language Generation.* Dawson, PA, 1990.

[Ward 1991] N. Ward. "A flexible, parallel model of natural language generation." Technical report UCB/CSD-91/629, UC Berkeley, Computer Science Division, 1991.

[White 1994] M. White. "A computational approach to aspectual composition." Ph.D. thesis, University of Pennsylvania, 1994.

[Wierzbicka 1980] A. Wierzbicka. *Lingua mentalis: the semantics of natural language.* Sydney: Academic Press, 1980.

[Weisgerber 1950] J.L. Weisgerber. *Vom Weltbild der deutschen Sprache.* Düsseldorf: Schwann, 1950.

[Wunderlich 1991] D. Wunderlich. *Arbeitsbuch Semantik*. Königstein: Athenäum, 1991 (2nd edition).

[Zydatiss, 1990] W. Zydatiss. "A contrastive analysis of a German instructive text and its English translation." *Lebende Sprachen* 4, 1990