Discourse Analysis and the Need for Functionally Complex Grammars in Parsing

Christian Matthiessen, Michael O'Donnell & Licheng Zeng
Department of Linguistics, University of Sydney*
Version: Wed, Dec 3, 2003

1. Introduction: discourse analysis as an NLP task

Natural language processing (NLP) is concerned with building communicating systems -- systems that can understand text (for example, in order to be able to answer questions), systems that can generate text in pursuit of some communicative goal, systems that can translate text from one language to another, and systems that can abstract and summarize information from text. Here the natural language system takes on the role of an ordinary language user taking part in some act of communication. Consequently, the highest level of meaning is foregrounded -- that is what is to be understood in text understanding, presented in generation, translated in translation and distilled in abstracting. In NLP research, we may not yet have the techniques for reaching a full coverage of this level -- for instance, machine translation may take place at the level of grammar, or at the highest level with some interlingual representation of individual sentences but without translation of discourse semantic patterns -- but this level is still the long-term goal; and it constitutes the measure of success of any natural language system. Lower levels are backgrounded as automatic intermediate steps in the language processing; and so are the patternings at all levels of representation. However, the situation is different with a natural language **meta-processing** system -- a system whose task is not to function as a natural language user but as a 'linguist'.

Thus we can contrast a language processing system for understanding text with a meta-processing system for analysing discourse.

(i) A text understanding system has to interpret strings graphologically or phonologically, interpret the graphological/ phonological representation lexicogrammatically, interpret the lexicogrammatical representation semantically, integrate the semantic representation with the rhetorical development of the text, the 'knowledge base', the 'user model' and determine the current communicative goals. A likely measure of the

* The research discussed here is in part supported by Fujitsu, Japan; and the development of of a discourse analyser is a Fujitsu project led by Guenter Plum. The other members of the research team whose work we draw on are Arlene Harvey, Chris Nesbitt and Guenter Plum. They have contributed substantially; the paper would not have been possible without their contributions.

success of a text understanding system is its ability to generate answers in some form.

(ii) Ideally, a discourse analysis system would go through the same steps as a text understanding system but it also has to be able to foreground the analyses at the various levels and identify and evaluate patterns. For instance, a text understanding system might use the information derived from the thematic analysis of a clause to identify the place where the new information the clause represents is to be integrated in the knowledge base of the system, but a discourse analysis system would also try to determine how the current theme selection relates to the history of previous theme selections, whether these theme selections create a method of development of a particular kind, and so on. And the discourse analyser might evaluate the success of the text in these terms.

This is an ideal discourse analyser -- a text understanding system with the added capability of examining the analysis at various levels to make explicit how the text works or does no works at these levels. To put this in terms of knowledge, the understanding system has to 'know' language; the discourse analyzer also has to 'know about' language. For practical purpose, it is currently difficult to develop a comprehensive discourse analyser with full text understanding capability. On the one hand, text understanding is still being developed and full understanding is not yet achievable. On the other hand, developing the resources for a text understanding system is still very labour intensive -- in particular, the task of building comprehensive models of general and domain knowledge. Consequently, it seems more realistic to aim for discourse analysers with only partial text understanding capability. Depending on the application of the discourse analyser, this needn't be a serious drawback.

Applications include

(1) using a discourse analyser as a linguistic research resource to investigate particular texts, texts belonging to particular registers ('sublanguages') such as stock market reports or research papers in physics, or the system behind the texts analysed. Such a resource will obviously aid working linguists, but it can also be of fundamental importance in developing natural language systems.

(2) using a discourse analyser to evaluate particular texts. In many contexts of instruction and documentation, texts produced either directly or through translation need to be checked for communicative effectiveness, both globally and locally. However, the flow of texts produced may be too great to handle manually; and the linguistic expertise to carry out the task may not be available. One solution is to build a discourse analyser capable of spotting problems with individual grammatical units and of producing 'profiles' that can be used by editors in determining whether the texts profiled need any revisions.

We are currently engaged in constructing a discourse analyser whose primary task is to aid in the evaluation of text; but it is also intended as a research tool of the type described in (1). Here we will briefly discuss the design of such a system and then focus on the demands on the grammatical parser. We will suggest that a grammar serving in a discourse analysis system has to be reasonably comprehensive in its

coverage and has to yield descriptions that are maximally rich in information. The latter is needed to provide as much support as possible for higher-level analysis. In short, the task of discourse analysis is greatly facilitated by an extensive functional grammar. We will briefly characterize such a grammar and then deal with the issue of how a large functional grammar can be managed in parsing.

2. Global organization of a discourse analyzer

Let us first describe the design for the discourse analyzer as a text understanding system with the addition of a capability of examining the analytic results at the various levels of analysis. After our brief sketch, we will indicate which components have already been implemented and the degree to which other components are intended to be fully developed. The text understanding system can be divided into resources (or knowledge sources) and processes (or procedures, such as parsing procedures). The resources are specified declaratively and are, in principle, the same as the resources used in a text generation system. We are in fact drawing on the resources of the Penman text generation system (see e.g. Mann & Matthiessen, 1983; Penman documentation) in the development of the discourse analyzer. The processes use the resources to analyse products at various levels; these products are then available as input to further processing in the understanding system. They can also be accessed by a linguistic examiner for 'meta-processing' such as the identification of referential and thematic patterns. A very simple version of the overall model is shown in Figure 1. (Although not shown in the diagram, the process of linguistic examination draws not only on the analytic products but also on its own resources -- knowledge *about* language: cf. Section 4 below.)

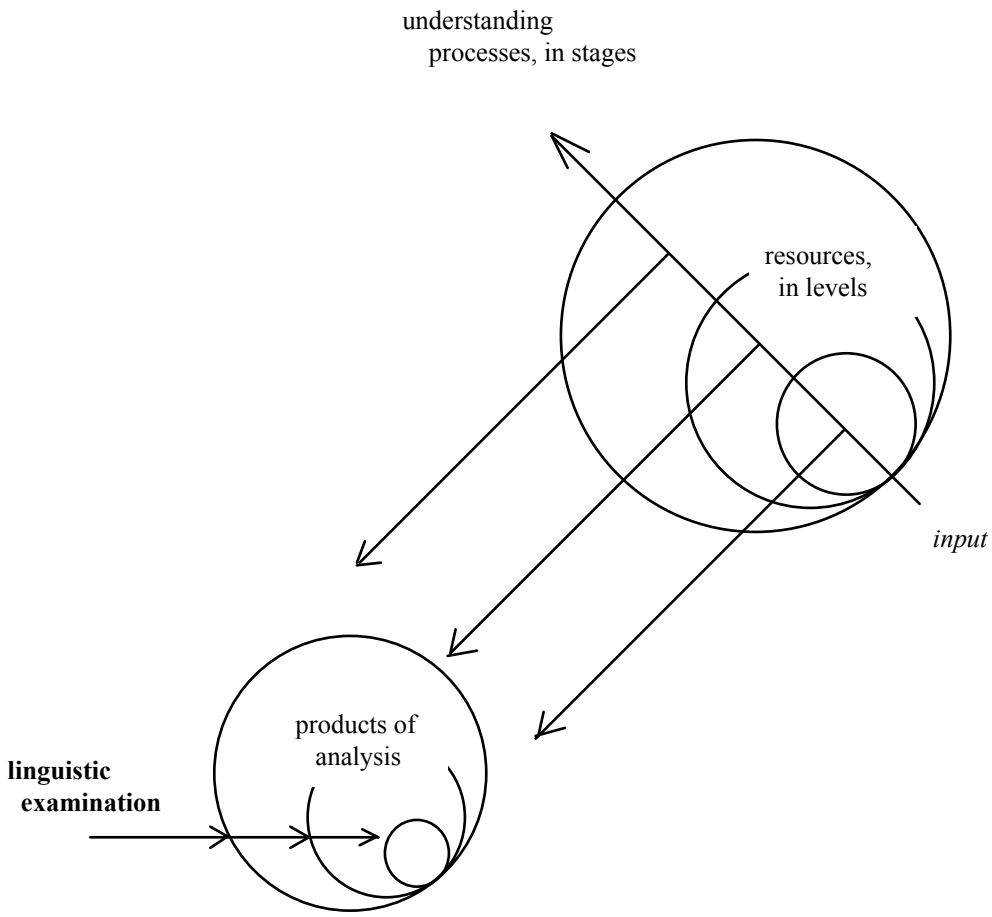


Fig. 1: A discourse analyzer

Both the processes of understanding and those of examination depend crucially on the organization of the resources. The organization is based on stratal theories of language such as stratificational theory and systemic-functional theory; in particular, it is based on Halliday's tri-stratal theory of language (e.g. Halliday, 1978, 1984, 1985; Matthiessen & Halliday, to appear) that also informs the Penman system. The levels or strata are subsystems of language, related by realization (inter-stratal mapping). They are (discourse) semantics, lexicogrammar (grammar, including both syntax and morphology, and vocabulary), and graphology (or phonology). Semantic choices and structures are realised as grammatical choices and structures, which are in turn realised as graphological choices and structures. The understanding process has to arrive at the specifications within a given level and each level has its own subprocesses, such as a bottom-up, breadth-first parsing process for the lexicogrammatical level; but the understanding process also has to derive higher-level specifications from lower-level ones. The role of the text parser is to move from the graphological form of the text, towards a semantic representation of that text. Let us illustrate this by means of the local representations of the string *The accountant left from La Guardia:* see Figure 2, which shows the three levels of the 'products of analysis' in Figure 1. The levels of specification are as follows:

Graphological Specification: At the lowest level, it is specified graphologically. This specification includes not only graphological constituency (letters, graphemic words, sentences) but also structure signals such as the full stop and graphemic prosodies such as italics and bold face (see Sefton, 1990, for discussion of graphology in systemic theory).

Lexico-Grammatical Specification: At the next level, it is specified lexicogrammatically as a wording -- grammatical structure & lexical items. We will say more about the nature of this specification presently; it can be thought of as a rich 'functional description', of the kind used in systemic-functional grammar (see Section 3 below).

Micro-Semantic Specification: At the next level, it is specified semantically (conceptually). This specification includes the kind of information commonly thought of as instantial knowledge in the knowledge base of the system.[1] We have shown it as an instantial concept frame, 'leave', with three roles -- two participant roles (Actor, filled by the constant 'John Doe', and Place, filled by the constant 'La Guardia') and a time role (the time of the occurrence of the process of leaving). The concepts and roles in this construct instantial general concepts and roles: John Doe instantiates 'accountant', the class used in this particular reference to him, and La Guardia instantiates airport. The general concepts are organized into a subsumption lattice so that it is possible to infer, among other things, that leaving is a type of directed motion, which in turn is a type of action.

[1] The notation used in the Penman text generation system is LOOM, a frame-based inheritance representation with logic notation for instantial knowledge developed at USC/ Information Sciences Institute by W. McGregor and others. A feature-based representation of the kind used for the grammar would also serve our purpose.

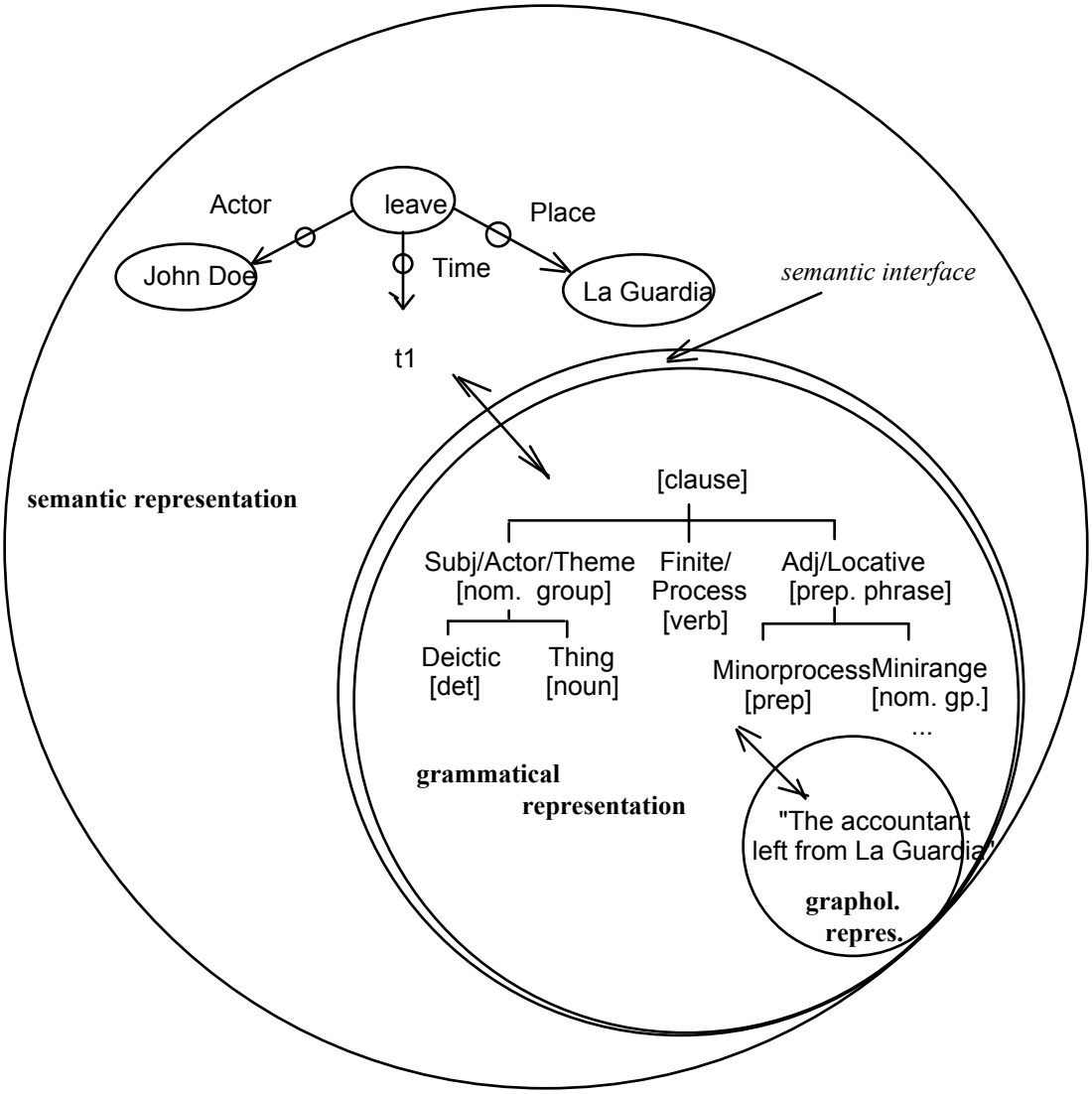


Fig. 2: Stratified representations of products of analysis

The specifications just described are supported by the currently implemented resources. The model provides constraints on the allowable possible structures at each stratum. It also includes sets of inter-stratal mapping relations between each stratum - relations showing how the structures of each stratum constrain the structures of the other. As far as the processes using these resources are concerned, one unification-based parser drawing on the systemic grammar has been developed by Kasper (e.g., 1987, 1988). O'Donnell is currently developing a systemic parser being used in the discourse analyser described here: see further below. The process of mapping from semantics to grammar has been implemented for the generation direction (relying on the semantic interface known as the chooser & inquiry framework, see e.g. Mann, 1983; Matthiessen, 1988). In research initiated at USC/ ISI by Kasper, this is currently being re-represented in a fully declarative form so that it can support analysis as well as generation (see O'Donnell, in prep.).

The scope of the resources and specifications of analysis extend beyond what has been suggested so far in two ways.

(i) They cover the full functional spectrum of meaning -- interpersonal and textual meaning as well as ideational meaning. At the level of semantics, this means that information beyond the ideational knowledge base has to be included: see e.g. Bateman & Matthiessen (1989, in press). The semantic analysis is multi-functional.

(ii) They cover not only micro-units such as clauses and propositions as shown in Figure 2 but also macro-units such as whole text and rhetorically coherent text segments, eg. theme progression, reference chaining etc. The semantic analysis includes micro-semantic analysis and macro-semantic analysis.

Both of these extensions are absolutely critical to a discourse analysis system. The full functional spectrum is represented directly at the stratum of grammar (as well as at the stratum of semantics). The semantic macro-analysis has to be built up from the information yielded by the grammatical micro-analysis and the semantic micro-analysis. We now turn to the lexicogrammatical stratum.

3. The lexicogrammatical stratum

According to systemic-functional theory, lexicogrammar is a resource for construing meanings as wordings: it provides the strategies for realizing meanings as grammatical structures, grammatical items and lexical items. More general meanings tend to be realized by grammatical structures such as Subject ^ Finite vs. Finite ^ Subject and grammatical items such as *the, a, must, can, at, in* whereas more specific meanings tend to be realized by lexical items: the difference between grammar and lexis is one of delicacy (degree of generality/ specificity).

3.1 Grammar organized as a resource for semantics

Lexicogrammar stands in a **natural** relationship to semantics (see Halliday, 1985, for the theoretical point and an account of English grammar that demonstrates the point). The form of language has evolved in the expression of meaning, so it is little wonder that the organisation of the grammatical resources reflect the organisation of the semantics. The semantic naturalness of grammar is a very important fact in the context of discourse analysis: it means that provided we have uncovered the full extent of the organization of lexicogrammar, we also have the potential for revealing a good deal of semantic information -- including information about the organization of discourse.

The basic principle of grammatical organization is that of **choice**: the grammar is organized as a large set of inter-dependent strategic options in wording meanings. These options are semantically interpretable and they reveal how the grammar, both in its global organization and in its local organization, is a projection of semantic organization. An option is represented by means of a **feature** and a set of options, a choice, is represented by a **system** of two or more features in a relation of exclusive disjunction -- only one of them can hold for a given grammatical unit being analysed. A choice is not available globally in the grammar; it is only available under certain more local conditions. These conditions are simply features from other systems and the conditions are represented as a Boolean combination of features -- a simple feature, a conjunction of features, a disjunction of features or some combination of conjunction and disjunction. This is the **entry condition** to a system. A system thus consists of an entry condition and a set of options conditioned by the entry condition:

(i) an entry condition: a Boolean combination of features;

(ii) a set of options: an exclusive disjunction of two or more features.

(For example, 'clause: indicative/ imperative' is a system whose entry condtion is 'clause' and whose set of options is 'indicative' vs. 'imperative'; 'imperative/ declarative: tagged/ untagged' is a system whose entry condtion is the disjunction of 'imperative' and 'declarative' and whose set of options is 'tagged' vs. untagged'; and 'effective & material: doing' is a defective system or gate whose entry condition is the conjunction of 'effective' and 'material' and whose set of options is the simple feature 'doing'.) Systems are thus related through their entry conditions and form networks of systems -- **system networks**. Such a network represents a partial ordering of the set of systems describing the grammar of a language. It is acyclic but it is a lattice rather than a strict taxonomy.[2] Figure 3 shows a fragment of a system network. The ordering of system from left to right in dependency is known as **delicacy;** the system network in the Nigel grammar extends further in delicacy, for instance to include systems dependent on 'mental', 'verbal', 'relational' and 'modal'. (The features that have been underlined constitute the selection expression -- the set of paths through the network -- for an example to be discussed in Section 3.2.)

[2] The system network was introduced by M.A.K. Halliday in the early 1960s. While features have been used in formal syntactic theory since the 1960s, it is only more recently that the need for a theory of feature organization has been emphasized -- an organization such as the subsumption lattice of HPSG.

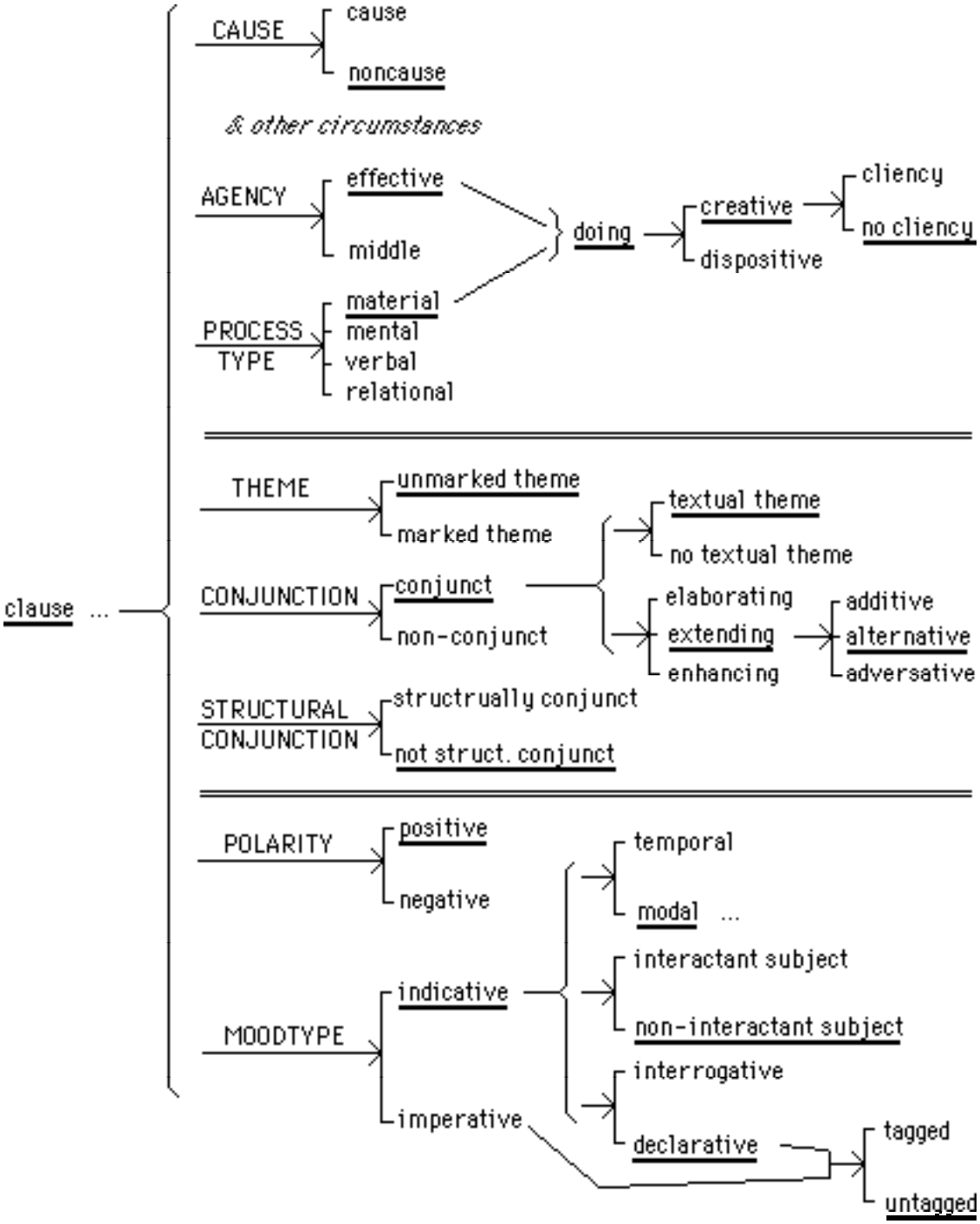


Fig. 3: Example of system network with realization statements

Any feature in the system network may have associated with it one or more specifications of how that feature is expressed or realized in terms of grammatical structure and grammatical and lexical items. These specifications are called **realization statements**. A realization statement occurs in the context of a systemic feature and consists of a realization operator and one or more operands. The realization operator is one of a small set of minimal specifications of wording -- insertion (the presence of a function; symbol: +), ordering (the relative ordering of two

functions; symbol: ^), expansion (a constituency relation between two functions; symbol ()), and preselection (the realization of a function by a unit/ item characterized by a particular feature or set of features; symbol: :). The operands are one or two grammatical functions (such as Subject, Actor, Theme; Finite, Process) and grammatical or lexical features. Relevant realization statements for the features in the system network of Figure 3 are tabulated below; they have been grouped according to metafunction for presentational clarity (the metafunction is not itself a formal part of the realization statement):

meta-function	**feature** [paradigmatic context]	**realization statement** [structural specification]
(general)	clause	+ Process; + Medium; Process: verbal group; + Theme; + Topical; Theme (Topical); # ^ Theme
ideational	material	+ Actor; Actor: nominal group
	doing	+ Goal; Goal: nominal group; Goal/ Medium
	creative	Process: 'creationverb'
interpersonal	indicative	+ Mood; + Subject; + Finite; Mood (Subject, Finite)
	declarative	Subject ^ Finite
	non-interactant subject	Subject: third person; Finite: third person
	modal	Finite: modal auxiliary
textual	conjunct	+ Conjunctive; Conjunctive: conjunction
	unmarked theme & declarative	Topical/ Subject
	textual theme	Theme (Conjunctive); Conjunctive ^ Topical

The system network and the realization statements together constitute the grammar's resources for describing expressions -- for assigning them systemic and structural specifications. These resources are neutral with respect to generation and parsing; they can be used by both generation and parsing procedures. Kasper (1987, 1988c) has shown that it is possible to re-represent the resources using the notation of Functional Unification Grammar (cf. also Kay, 1979; Winograd, 1983: Ch. 6): Systemic Functional Grammar is part of the typological[3] family of feature & function grammars or unification grammars that came into focus in the 1980s.

[3] We write 'typological' because the family is not genetic -- developing from different origins, systemic functional grammar and a number of more formal grammar (GPSG, LFG, HPSG) have come to resemble one another in a number of respects as far as the representational resources are concerned.

Let us now turn to an example of what the result of the analysis of a clause looks like. We will focus on the descriptive result of the process of analysis, not on the parsing procedures that lead to this result. At this point, we are interested in what kind of specifications the components in the discourse analyser can get from the grammar (Sections 3.3 and 4). Having demonstrated the value of producing rich, functional grammatical descriptions, we will turn to the question of how to handle an extensive functional grammar in parsing (Section 5).

3.2 Grammatical descriptions

The analytic specification is both (i) systemic and (ii) structural; and it is multifunctional (ideational, interpersonal, and textual). The example *Alternatively, the DASD dataset can be created || by specifying the NEW operand in a TSS ALLOCATE command,* is a combination of two clauses.

(i) Systemic description. Focussing on the first clause, we can give the most general systemic feature description as shown in Figure 4.[4] The systemic descriptions is called a **selection expression**; it is simply a set of selection paths through the system network. The majority of the features in the selection expression given here appear underlined in Figure 3 above.

[clause: major: independent:

textual:

full clause (i.e., non-elliptical):

non-structural conjunction: extending: alternative &

textual theme &

unmarked (experiential) theme &

receptive: non-agentive

interpersonal:

indicative: (declarative: untagged) & non-attitudinal & non-interactant subject & (modal: (modulation: ability) & low) &

positive

ideational*:*

material & effective: doing: creative: no cliency &

non-location & non-extent & non-cause & non-manner & non-matter & non-accompaniment & non-role]

Fig. 4: Systemic feature description (selection expression)

(ii) Structural description. The structural description is given in Figure 5; it also shows the most general classes of the realizations of the functions ('nominal group', etc.) but more delicate classes (such as 'modal') are not shown. The internal organization of the units realizing clause functions is only shown for the nominal group *the DASD dataset*. (Such internal structures and associated selection expressions are produced in earlier cycles in the bottom-up parsing process we work with.)

---

[4] Output features of gates (defective systems with single output features) are not shown since they can be inferred from the feature description.

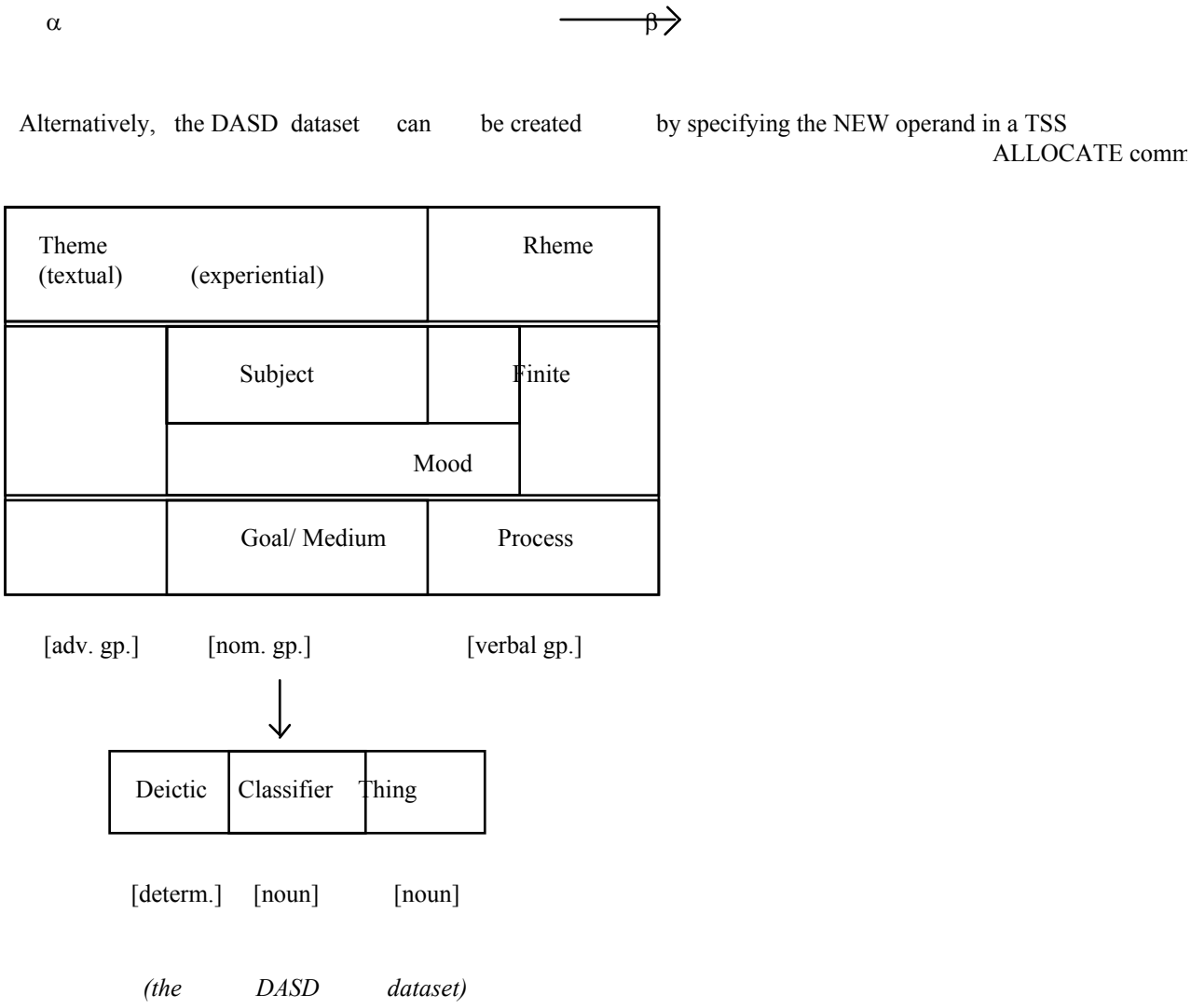


Fig. 5: Structural description

Textually, the Theme of the clause, *Alternatively, the DASD dataset,* indicates how the clause is to be related to its text history, both in terms of rhetorical development (*alternatively*) and in terms of the progression of topics (*the DASD dataset*). We will show this more clearly in Section 4, where we discuss the passage from which this clause has been taken. **Interpersonally**, the clause is 'declarative'. It is also modal, but semantically, it is not clear whether the clause means 'you can create' or 'something can create': taken just by itself, it may or may not be an instruction to the reader.[5]

[5] The dependent clause *by specifying ...* can be taken as evidence that the main clause is an instruction; but the normal principle of recovering the identity of an implicit Subject in a dependent clause by reference to the Subject of its main clause does not work: it is not the DASD dataset that would specify. However, if 'you' had been selected as Subject of the main clause thus clearly making it an instruction -- *alternatively, you can create a DASD dataset by specifying the NEW operand in a TSS ALLOCATE command* -- the thematic organization of the clause would have changed and the relationship to the thematic progression at this point in the discourse would have been lost: cf. Section 4 below.

Ideationally, the clause is a 'creative' 'material' one: it represents a Goal as being brought into existence. The process is clearly part of the domain of the operation of the programme the text is concerned with.

As the grammatical fragments given here, both the resources in Section 3.1 and the descriptive examples here, illustrate, grammar interpreted according to functional theory *looks* semantic; it operates with features such as 'interactant subject', 'modal', 'material', 'creative', 'perceptive', 'theme', 'extending' and functions such as Subject, Mood, Actor, Medium, Senser, Phenomenon, and Theme, but it is important to bear in mind that the categories are still based on the grammatical distribution of forms. In Halliday's (1985: xx) words:

> all the categories employed must be 'there' in the grammar of the language. They are not set up simply to label differences in meaning. In other words, we do not argue: 'these two sets of examples differ in meaning, therefore they must be systematically distinct in the grammar'. They may be; but if there is no lexicogrammatical reflex of the distinction they are not. ... The fact that this is a functional grammar means that it is an interpretation that is based on meaning; but the fact that it is a 'grammar' means that it is an interpretation of linguistic forms. Every distinction that is recognized in the grammar -- every set of options, or 'system' in systemic terms -- makes some contribution to the form of the wording.

Consequently, any grammatical category is recoverable in parsing.[6] Since the grammar specifies descriptions that are semantically fairly transparent, the interface to the semantics is quite straightforward (cf. Section 3.3) and it is possible to envision a semantics that is considerably richer than the traditional kind of semantics.

3.3 Micro-semantic interpretation

The systemic and structural descriptions of the clause are the final product of the stage of grammatical parsing. The next step is to interpret the specifications semantically: see Figure 6. The interpretation is mediated by a semantic interface we won't discuss here. For our purposes, the important points are (i) that the semantic interpretation is multifunctional and (ii) it is the basis for further macro-semantic interpretation. We return to the second point in Section 4 below. The fact that the semantic interpretation is multifunctional means that a clause is interpreted not only in the ideational terms of the knowledge base -- or, more appropriately, the ideation base. It is also interpreted in interpersonal and textual terms: interpersonally, it is related to the interaction base of the system -- a resource usually discussed in NLP under the heading of 'user modelling' -- and textually, it is related to the text base of the system, which is concerned with the rhetorical organization of the text, with identifiable referents, with the distinction between given and new information, etc.

---

[6] The label worn by "grammatical" categories does not define the category. Rather, the category is defined by the relationship it bears to the other terms in the system (cf Hjelmslev, e.g. 1943). The label is just a mnemonic which helps us refer to the category. 'Agent' and 'material' may label categories based on semantic grounds, but they do reflect grammatical criteria.

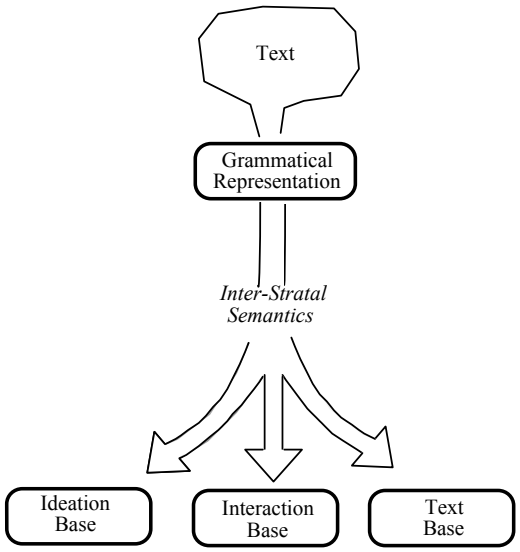


Fig. 6: Semantic interpretation of grammatical specification

Of the three bases, the ideation base ('knowledge base') is most fully developed. The interaction base has not been developed for the discourse analyser or for the Penman text generator. Certain aspects of the text base have been developed for the Penman text generator but have not been incorporated in the discourse analyzer yet. Full use of the ideation base in text understanding presupposes complete domain modelling. A number of domains have been modelled for the Penman text generation system, but we have not undertaken domain modelling for the discourse analyser yet. However, for the purposes of discourse analysis, it is possible to use the most general aspects of the ideation base -- known as the upper model (cf. Matthiessen, 1987; Bateman et al, 1989) -- without full domain modelling.

4. Using information from the grammar in discourse analysis

Having suggested what kind of analyses the grammar can produce, we can now say a few words about the way in which the information from the grammar can be used by the discourse analyzer. The earlier example whose grammatical analysis was given above is taken from a software manual. Let use provide a longer extract to illustrate some of the issues the discourse analyzer can deal with given a rich grammatical analysis:[7]

```
[0] Creating a DASD dataset.

[1] This section describes the knowledge required to create
a DASD dataset.

[2] Temporary and permanent datasets

[3] A DASD dataset can be created by specifying NEW in the
DISP parameter of a DD statement in a job control statement.
```

[7] The text from which this passage is taken was analysed by Arlene Harvey and will be discussed in more detail elsewhere.

```
[4] Alternatively, the DASD dataset can be created by
specifying the NEW operand in a TSS ALLOCATE command. [5]
Two types of DASD dataset can be created, as described
below.

[6] Temporary dataset:

[7] This is a dataset that is used temporarily during the
execution of a batch job. [8] A temporary dataset does not
need to comply with the [product name] standard naming rule.
[9] A temporary dataset, however, cannot be registered in
the [product name] management dataset.

[10] Permanent dataset:

[11] This is a dataset that is saved ...
```

An analysis of this passage based on systemic functional grammar will produce a rich description similar to those discussed in Section 3.2 and illustrated in Figures 4 and 5. That is, the analytic description is both systemic and structural. In addition, through micro-semantic interpretation (Section 3.3), it is possible to establish associations of nominal groups and processes to referents and concepts in the knowledge base, as well as to various semantic structures that the passage instantiates. These analyses, both systemic and structural, are multi-perspectival; they represent features and functions in terms of the ideational metafunction ('knowledge representation'), the interpersonal metafunction ('information exchange'), and the textual metafunction ('information flow').

To illustrate the process of examination, let us present an abbreviated description of each clause in the text[8] and show how the rich parsing result supports the linguistic examiner in characterizing the passage and identifying potential problems. (Themes are underlined to make it easy to scan the passage for its thematic progression. Under the text, textual (THEME), interpersonal (MOOD), and ideational (TRANSITIVITY) analyses are provided in that order.)

Segment [1]

[1]This section	describes	the knowledge "required to create a DASD dataset".
Theme	Rheme	
Subject	Fin/Pred	Complement
Sayer	Proc: verbal	Verbiage

A good deal of information can be inferred from the transitivity functional structure of (1). The process type is 'verbal', which suggests that the clause represents the meta-domain of the manual as a piece of discourse rather than the domain of a particular piece of software. The Sayer is the nominal group this section; and 'section' can be located in a taxonomy of discourse elements (like 'chapter', 'paragraph', 'figure', 'diagram') in the ideation base. The Verbiage is the nominal group *the knowledge*

[8] As the sample description in Section 3.2 shows, the linguistic examiner will have access to considerably more detail than we can show here. In particular, it will have access to the systemic descriptions of the clauses, not only the structural ones.

required to create a DASD dataset. Sometimes the Verbiage is used to represent the subject matter of the discourse. The Verbiage here is such a case (cf. *this section talks about ...*). Textually, this Verbiage of subject matter is non-thematic and is presented in the unmarked position of new information. This is a common discourse strategy for introducing future Theme candidates into the discourse; and, as we will see, this potential is indeed taken up (cf. for example segment 3).

Segment (1) is itself as a whole in a discourse thematic position, which is an aspect of macro-semantic organization (cf. Martin, to appear) and as such it orients the reader of the manual to topics, layout or organization of the following text in the section. Its organization is thus of special significance for the passage as a whole.

From the above points, a number of generic expectations about the subsequent text can be inferred. Let us state these for each metafunction:

Ideational metafunction (knowledge representation): Processes in the text can be expected to be largely of creative, dispositive and relational types, as in a computer manual in general. Mental processes like “think, see, like” do not normally convey knowledge about computer operations. Also, there might very well be definitions and classifications in the passage, as they are important means to conveying knowledge. The examiner should thus watch out for clauses that are 'relational: identifying' for definition, with patterns like “X is Y ...”, and clauses that are 'relational: ascriptive for classification, with patterns like “X consists of Y, X is a Y” etc.. The tense selections can be expected to be largely the simple present tense.

Interpersonal metafunction (information exchange): Since the text is focusing around description (“This section describes...”), declarative clauses can be expected. Interrogative and imperative clauses are inappropriate since they serve to demand information or action from the readers. Clauses may have various kinds of modalizations of probability like “can, might, possible” etc, since there may be uncertainty in knowledge. However clauses are not expected to have a lot of high-degree modulations such as “required to”, “must”. Had “knowledge” in (1) here been replaced by the word “procedures” or “steps”, we would expect the following text to have more imperative clauses and clauses with high-degree modulation.

Textual metafunction (information flow): We expect the text to develop around a topic. The parser tells us that “required to create a DASD dataset” is the qualifier of “knowledge”, and that “DASD dataset” is the new information in the qualifier, therefore we can temporarily assume that “DASD dataset” is the topic and that “DASD dataset” must be the theme of at least one of the subsequent clauses. We can further infer that the chain of references to the “DASD dataset” will be the longest in the following text.

We can now consider segments (2) through (10). Two of these are subheadings in the manual. They are graphologically recognizable as subheadings and their discourse function is to give a piece of information thematic status. They thus predict potential Themes in the following clauses and if this prediction is frustrated, the thematic development of the discourse may be problematic.

Segment [2]

[2] Temporary and permanent datasets

(subheading)

Segment (2) is a subheading. Subheadings usually indicate the topical Theme of the following text, and (2) is roughly the same as our prediction from (1), so the topic is further confirmed. However, while *dataset* was predicted as a potential topical Theme in (1), the classification into temporary and permanent datasets wasn't; and although the subheading predicts this distinction as a potential Theme for the following segments, it is not picked up thematically in these segments, i.e. in (3) and (4). It is picked up in segment (5), but then only indirectly as *two types of DASD dataset.*

Segment [3]

[3.1] A DASD dataset	can	be created
Theme	Rheme	
Subject	Fin	Predicator: passive
Goal		Process: material:creative

[3.2] by specifying	NEW	in the DISP parameter of a DD statement in a job control statement
Predicator	Comp	Adjunct
Proc: verbal	Verbiage	Locative

The analysis of (3) shows that "DASD dataset" is the Theme of the clause, an indication that our assumption about "DASD dataset" as topic has been confirmed. The Finite element of (3), namely, "can" as modality also conforms to assumptions we infer from (1). The Process "create" is also normal. The function structure of (3.1) does not specify the function Actor, so we do not know who can "create DASD dataset". And the message of agent cannot be recovered from the history of analysis of this text by the linguistic examiner. At this point, the linguistic examiner may issue a warning that the Actor of the Process of creating cannot be recovered from context.The bound (dependent) clause "by specifying new..." in (3.2) serves as hypotactic enhancement of the main clause in that it informs us about the means of the operation stated in the main clause.

Segment [4]

[4.1] Alternatively,	the DASD dataset	can	be created
Theme	Rheme		
Conjunctive	Subject	Finite	Predicator: passive
	Goal		Process: material: creative

[4.2] by specifying	the NEW operand	in a TSS ALLOCATE command
Predicator	Complement	Adjunct
Proc: material	Verbiage	Locative

The analysis of (4) is similar to that of (3), and hence is generally appropriate to the passage. In addition, the examiner knows from the presence of the Conjunctive "alternatively" that (4) extends the preceding discourse with a presentation of an alternative to at least (3). Also the examiner notices the second mention of "DASD dataset" in (2) is preceded by a Deictic, so it extends the reference chain of the "DASD dataset" referent.

Segment [5]

[5.1] Two types of DASD dataset	can	be created,
Theme	Rheme	
Subject	Finite	Predicator: passive
Goal		Process: material: creative

[5.2] as	described	below.
	Predicator	Adjunct
	Process: verbal	Locative

The examiner might find (5.1) is a bit problematic. The Theme of (5.1) is the nominal group "two types of DASD dataset", in which "type" is the Head, and "of DASD dataset" is the Qualifier. Since "type" has never been mentioned before in the passage, it is new to the reader. Although given information rather than new information is selected as Theme in the default case, here "two types of DASD dataset" has been given the status of the Theme of the clause, serving as the presentational point of departure. The selection of "two types of DASD dataset" as Theme also puts the thematic progression pattern built up in the preceding discourse into disorder. An alternative strategy would be to introduce "two types of DASD dataset" non-thematically in an existential clause: *There are two types of DASD dataset.* Each of the two types would then be picked up thematically (in segments [6] through [9] and [10] onwards), yielding a very common type of thematic progression that is easy to follow.

Segments [6]

[6] Temporary dataset:

(subheading)

Segment (6) is not problematic: it is a subheading that serves to make the Theme of clause (7) (and subsequent clauses until the next subheading in [10]) very explicit.

Segment [7]

[7] This	is	a dataset	<7A>
Theme	Rheme		
Subject	Finite	Complement	Predicator: passive
Identified	Proc:relational	Identifier	Process: material: creative

[7A] That	is	used	temporarily	during the execution of a batch job
Theme	Rheme			
Subject	Finite	Predicator: passive	Adjunct	Adjunct
		Process: dispositive, passive	Manner	Locative

Clause (7) is a definitional clause, which is expected from the inference based on (1).The linguistic examiner will recognize (7) as a normal identifying clause used as a definition; it can recover that "this", the nominal group serving as Identified in (7), refers to "temporary dataset". So the reference chain on "dataset" is further extended.

The Themes of clauses (8) and (9) create a normal thematic progression pattern. The mentions of "DASD dataset" in (8) and (9) are preceded by "a" as Deictic, since the reference is to a general class. So the linguistic examiner will include them as part of the definition given in (7). This method allows examiner to trace the definition of a concept. It helps the manual editor find out how much information is given in the definition of a concept.

So much for our simulation of how the linguistic examiner evaluates a piece of text based on the rich outputs of a systemic functional parser. Although our illustration is informal and heuristic, we have set up a model of the linguistic examiner based on relational algebra and type persistence enforcement. We are not going to describe the model in detail here; we will just summarize some properties of the model:

- **Critical Language Patterns**. From the illustration above, we see that the way certain lexicogrammatical features and functions are deployed affect the quality of the manual text. That is, combinations of certain values of these linguistic factors are together diagnostic of the communicative success of the text passage being analysed. For example, the identifying relational process where the Identifier is a nominal group with a Qualifier (as in *This is a dataset that is used temporarily during the execution of a batch job*) indicates a well-formed definition. Let's call a group of lexicogrammatical features a critical language pattern (CLP) when they may be indicative of the communicative effectiveness of text. It is obvious then from the illustration above that the linguistic examiner is constantly trying to recognize and evaluate various CLPs from input text and the parsing results from the parser. The recognition of a CLP is a process of matching analysed text with linguistic patterns of a given CLP. The evaluation of a CLP is a process of unifying recognized CLP patterns with discourse "normal forms" that are inferred along the way of linguistic examination. The inference we obtain in the analysis of segment (1) is an example of discourse normal forms. Intuitively, a discourse normal form is a piece of heuristic or linguistically-motivated knowledge about normal discourse built with a CLP. If the evaluation of a CLP by the examiner yields an inappropriate value, a warning will be reported. Obviously the recognized CLPs and evaluated results should be stored as history and temporary result in the process of linguistic examination.
- **Multi-agent cooperation**. The illustration above also shows that in the process of examination, CLPs are coordinating and communicating with each other: while a

CLP is requesting messages from other CLPs in order to make a decision, other CLPs may be waiting for this CLP in order to proceed with their tasks. This problem can be considered an instance of a major problem in Distributed Artificial Intelligence, that is the coordination and communication of multiple intelligent agents to achieve a common goal (Rosenschein, 1986), with CLPs as agents in the system. We can draw an analogy between CLP examination and monitoring the instrument panels on a dash board. Suppose there are several gadgets on a dash board, with each of them constantly monitoring and indicating the change of one aspect. Some gadgets can be very simple; they just indicate the actual change of the objective environment and we can take action simply based on their values. There are some other gadgets that monitor both the changes of other gadgets (meta-monitoring) and objective environment, and the values they indicate are based on complicated computation on several factors. The complicated gadget type is an issue of multi-agent cooperation. As we see, CLPs can be both simple and complicated types of gadgets. For example, in the simulation of CLP examination above, the definition-tracing CLP, which is being monitored by the knowledge-introducing CLP on the appropriateness of describing a piece of software, is at the same time monitoring and waiting for the evaluation of the relational-process CLP which is trying to recognize "X is Y that..." pattern. Although there is not so far a sound and complete solution to the multi-agent problem, we can still borrow some techniques developed to solve concurrency issues, such as using semophore for process synchronization and the access-oriented programming paradigm to organize the sequence of CLP examining.

- **Inference rules**. As shown in the simulation of the analysis of clause (1), the information about discourse normal form that we can obtain from the analysis of clause (1) can be very rich. These messages are not simply anticipated by linguists beforehand and stored in a knowledge base for later use. The linguists provide basic knowledge about discourse normal forms. The linguistic examiner will try to infer more "logically valid" discourse normal forms on the basis of three resources (a) inference rules (b) the knowledge about discourse encoded by linguists (c) representational formalism for CLPs. From the illustration above, we see the result of inference at stage can also be used as assumptions for later stages. Also results of inference can be revoked if controversial facts are present. Another thing to note here is that inference should be able to work both incrementally and completively. By incrementally, we mean new inference is made with each new piece of input from the parser. By completively, we mean the inference can be made only after a chunk of text has been processed by the parser and all results of analysis become available to the linguistic examiner at one moment. This implies that inference is made independent of parser result. The more pieces of evidence are available at a given time, the more inferences can be drawn at that time.

- **Data persistence and dynamic type checking**. We introduce these notions as a technical solution to CLP examination. The data persistence can be simplified as a notion of complete separation of value and their types in a programming environment. This separation means that values or data exist as pure symbols in the first instance, they are not classified as a particular data type when they are created in the program. As the values are used in programs, they are dynamically

bound to certain types and attributes. Thus values are considered as extensions or referents of particular data types. One of the consequences of the mechanism is so-called polymorphism, which is close to what we called multi-functional or multi-perspective description, that is, a value can have several types, or to be an element of several sets. For instance, a word of a clause can be Subject, Actor and topical Theme at the same time.The other consequence is that since values exist independently of their types, we can build up rather complicated type systems such as conceptual hierarchies, semantic networks or system networks, and then a value can be dynamically matched to several types in the type system. The advantage of having a complicated type system is that it provides us with an environment in which the type of inference mentioned above can take place. For instance, with a system network it is easy to reason about implications and multiple inheritance. It is not difficult to see that the process of CLP examination can be translated into the process of dynamic type checking, in which a manual text, a list of words, is stored as a list of symbols, various CLPs as well as discourse normal forms are organized into a complicated type system, which will take the form of system network in our implementation. Within this framework, the recognition of CLPs can interpreted as various coordinated processes constantly trying to bind symbols from the manual text to various CLP types in the system network and to maintain all these bindings. In addition, the evaluation of CLPs can be viewed as binding bound CLPs to various types of discourse normal form. Hence our model also includes a type definition language for defining CLPs and discourse normal forms, and a type checker that conducts the linguistic examination.

The resources that the linguistic examiner draws on include, apart from particular grammatical descriptions such as the one given in Figures 4 and 5 above, (i) history-recoverable data structures and (ii) knowledge about the grammar:

(i) one prominent feature of discourse analysis is the constant need to refer back to the analysis of previous discourse. The data structures used by the linguistic examiner must allow it to recover any analysis previously made. For example, in order to keep track of referents in text, each name consumed by the examiner will be compared with the previous mentions of that name.

(ii) in our model of discourse analysis, the knowledge about grammar consists of three parts: type hierarchy of CLPs, discourse normal forms and inference rules.

5. Complexity Problems and Solutions

In the discussion above, we have motivated the use of a large functional grammar, for parsing purposes. A computational grammar of this form exists, developed at Information Sciences Institute (ISI) as part of the Penman text generation system. The grammar is called Nigel (Matthiessen, 1985; Mann and Matthiessen,1985).

There are at present two projects setting up parsers to work off the Nigel grammar (suitably modified for the parsing task). One project directed by Bob Kasper at ISI (Kasper, 1988a, 1988b), and the other involving our research on text understanding and discourse analysis in Sydney.

A grammar like Nigel cannot be handled by just any parser. The size and complexity of the grammar requires special techniques. The rest of this paper will explore some of the techniques that can solve the problem of handling a large complex functional grammar.

5.1 Overview of problems and solutions

To get a better view of the problem, let's look at a typical analysis provided by the Nigel parser. Consider the analysis of *A dataset can be created by a specification of NEW in the DISP parameter* in Figure 4 above. Each unit is assigned a set of grammatical functions. Each unit also has an associated set of grammatical features, (shown below the functional analysis, but essentially integrated in with the functional labelling).

In a phrase-structure analysis, each node of a parse tree would have a single category label. Some semi-functional grammars, such as LFG or FUG, may assign perhaps twenty category labels to the clause, fewer to the lower ranking units. Nigel however, assigns 70 function and class categories to the clause-rank node, and a slightly smaller number to the three nominal groups in the clause.

The Nigel grammar offers a higher degree of choice within each unit, both because of its extensive grammatical coverage, as well as its delicacy in distinguishing distributional difference, and also because of its responsibility to the semantics.

A higher degree of choice means a higher number of alternative analyses at a local level. Combinatorial explosion will thus be a real problem for a Nigel parser, since the increased alternatives at each step of parsing multiply together to give large numbers.

So, the problem of parsing with Nigel is that the time taken to parse a clause may be longer than we desire to wait. We will now consider some of the means we are using to speed up the parsing process.

The solutions can be grouped under 6 headings:

1. Pre-parsing with a Cover-Grammar
2. Limiting Combinatorial Explosion
3. Avoiding Repetition of Analyses (structure sharing)
4. Minimising Resource Access Time (indexation)
5. Shifting Calculation to a pre-analysis stage (pre-compilation)
6. Handling Disjunction in Unification

5.2 Pre-Parsing with a Cover Grammar

Martin Kay (1985) suggests one means for avoiding the complexities of a large functional grammar -- the sentence is first parsed using a simplified grammar, producing an analysis which segments the sentence, but not much else. After this "the result of the parsing process can be unified with the original grammar to eliminate false analyses" (p. 277). He calls the simplified grammar a **cover-grammar**, a grammar which recognises all the sentences recognised by the larger grammar, but in far less detail.

The initial parse using the cover grammar does not suffer from the complexity problems of the larger grammar. And by constraining the structural possibilities of the larger grammar, much of the scope of ambiguity for the larger grammar is resolved.

Bob Kasper (e.g., 1988a) has followed this method independently. His parser for Nigel makes use of a simple phrase-structure grammar to segment the sentence into constituents. Each phrase-structure rule used in the successful analysis has associated with it a set of constraints on the Nigel systemic grammar. These rules set up, in parallel to the phrase structure analysis, a skeleton of the analysis from Nigel (see Figure 7 for a sample rule). The rest of the Nigel grammar is then used to flesh out the skeleton. Figure 8 demonstrates the mapping.

$NGPS_1$ -> $NGPS_2$ ^ PP
(:AND $FD(NGPS_1) = FD(NGPS_2)$
(:or (:and $PORTION(FD(NGPS_1)) = FD(PP)$
$PORTION(FD(NGPS_1))$:[portion-process]
$FD(NGPS_1)$:[portion-qualified]))
(:and $TEMPORAL(FD(NGPS_1)) = FD(PP)$
$TEMPORAL(FD(NGPS_1))$:[temporal-process]
$FD(NGPS_1)$:[temporal-qualified]))
(:and $ACCOMPANIMENT(FD(NGPS_1)) = FD(PP)$
$ACCOMPANIMENT(FD(NGPS_1))$:[accompaniment-process]
$FD(NGPS_1)$:[accompaniment-qualified]))
(:and $LOCATIVEQUAL(FD(NGPS_1)) = FD(PP)$
$LOCATIVEQUAL(FD(NGPS_1))$:[location-process]
$FD(NGPS_1)$:[location-qualified])))

Fig. 7: A sample phrase structure rule with associated Nigel constraint

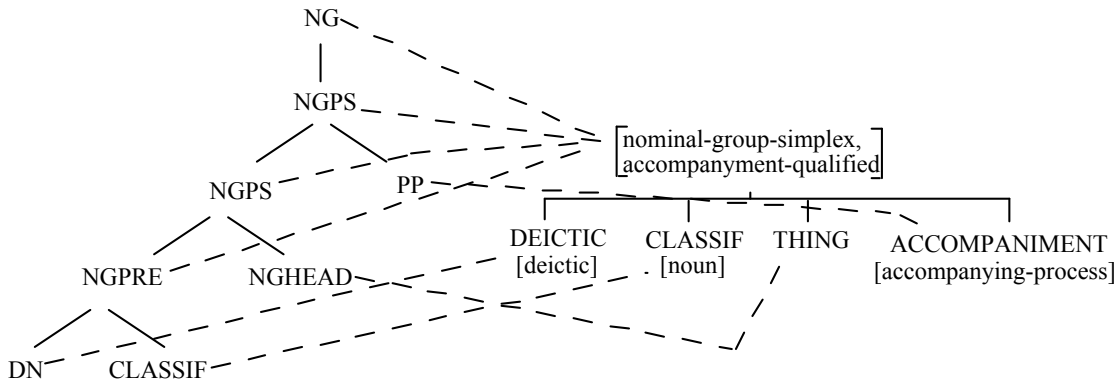


Fig. 8: Mapping From Phrase Structure To Systemic Structure

One improvement over this system would be to implement the cover grammar in the same formalism as Nigel e.g. a systemic cover-grammar. This would avoid the problems involved with using two grammatical formalisms to parse a text. Both the cover parser and the full parser can share the same parsing processes.

The problem with all of these approaches is that they require the maintenance of an extra level of analysis, which is largely unsupported by the language theory. An associated problem is that every time the main grammar is modified, so too must the rules which link this grammar to the cover grammar. For this reason, the parser that forms part of our discourse analyser uses other methods to reduce complexity.

5.3 Data Driven Approach

Small grammars (small in coverage or delicacy) can use a top-down parsing algorithm without much trouble -- the degree of hypothetical structure building will not be too extensive.

But for Nigel, top-down parsing is inappropriate. A top-down approach needs to make many structural hypotheses before reaching the input string, even using a depth first algorithm. If we look at a grammatical analysis provided by Nigel (Figure 8), we can note the degree of structure at just at the clause level (roughly 60 features, each representing one structural alternative chosen from among others). When we add in slightly smaller numbers of hypotheses at each structural rank below, and the possibility of depth recursion, one can see that the degree of hypothesis formulation is excessive.

A dataset	can	be	created	by a specification of NEW in the DISP parameter
[clauses clause full clause-simplex nonconjuncted independent-clause independent-clause-simplex indicative declarative material effective creative modal modality-nonconditional modality-positive nonvolitional possibility ability noninternal-subject-matter no-secondary nonattitudinal manner nonthematic-manner means nonmatter nonagentive passive-process noncliency noncomplemented nonoblique-complemented indicative-noninteractant explicit-declarative-subject untagged goal nominative-subject topical-subject nonplural-subject real auxstem-voice nonrole no-spatial-extent no-spatial-location no-temporal-extent no-temporal-location nonaccompaniment noncause lexical-verb-term-resolution]				
TOPICAL/SUBJECT/ GOAL/MEDIUM	FINITE	AUXSTEM/ VOICE	VOICEDEPENDENT/LEXVERB/ PROCESS	MANNER/MEANS

Figure 7: Demonstrating the detail of a Nigel Analysis

A parser for Nigel needs to be '**data driven**' (bottom up) -- working directly from the input, rather than 'hypothesis driven'. We need to make best use of the information we have, reducing the number of hypotheses we need to make. For the same reason, breadth first parsing is preferred, seeing all the data before building too much structure. Other techniques like top-down filtering make the system even more efficient.

5.4 Avoiding Repetition of Analyses

A backtracking or parallel parser will typically need to repeat the analysis of sub-components of a sentence for each structural alternative at a higher level. For instance, given "The horse raced past the barn fell", a backtracking parsers would process "The horse raced past the barn" as a clause, then fail to handle "fell", then backtrack. The second analysis would correctly handle "The horse raced past the barn" as a single nominal group, and properly handle "fell".

However, in the second attempt, a backtracking parser would re-analyse parts of the sentence which are the same for both analyses. The various possible analyses of "past the barn" would remain the same over both passes

Two techniques have been developed in the community for handling this problem - **well-formed substring tables** (WFST) and **charts**.

A well-formed substring table is "a mechanism enabling a parser to keep a record of structures it has already found, so that it can avoid looking for them again" (Gazdar&Mellish:179). The first time any string of items is analysed, all the possible analyses are stored onto a storage table. Subsequently, when the parser needs to analyse that string again, the parser can recover the prior analyses from the table.

The EDA parser makes use of this technique to reduce parsing complexity from a processing time in the worst-case exponential, to a more reasonable, perhaps n^3 time (yet to be demonstrated).

Chart parsing is a technique which allows all partial analyses of any unit to be stored away. Descriptions of the technique are available elsewhere, so we won't elaborate here. Both the Kasper Parser and the parser described here make use of chart parsing to eliminate repeated analyses.

5.5 Minimising Resource Access Time: Indexing into the Grammar

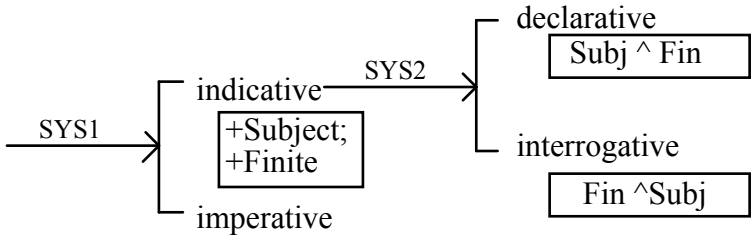
Martin Kay (1985) talks of *pre-compilation* of grammars: the grammar is initially available in a human-produced form, and before parsing starts, it is *compiled* into a machine-internal form. Pre-compilation is used in our discourse analysis system in two ways, described in this and the following section.

As the grammar is loaded in, various indexes into the grammar are stored away. Appropriate indexation allows quicker retrieval of grammatical information, speeding up the parsing process. For example, the following lisp code describes two systems in our human readable form:

```
(system  :name    SYS1
         :inputs  clause
         :outputs ((indicative (insert SUBJ)) (imperative)))

(system  :name    SYS2
         :inputs  indicative
         :outputs ((declarative (order SUBJ FIN))
                   (imperative (order FIN SUBJ))))
```

Graphically, these systems are represented as follows (cf. Figure 3 above), with realization statements included in boxes under the features they realize:



The Data Types used here are as follows:

System Names: SYS1, SYS2
Grammatical Features: indicative, imperative, declarative, interrogative
Grammatical Functions: Subject, Finite
Realisation Operators: insert, (+), order (^), (others...)

When we load these systems into the parser, we store this information into four hash tables (since hash tables are searched quicker than lists).

TABLE NAME	KEY	STORED
System-Table	System Name	The input and output features of that system
Feature-System-Table	Feature	The system for which this table is an output
Feature-Realisation-Table	Feature	The feature's Realisations
Feature-Input-Table	Feature	The systems which have this feature as input

Building multiple indexes into the same data structures means that we can go straight to the information we need, rather than searching through the resources for those parts we need. Other indexes can be built also, if needed. Indexes take some time to build initially, but speed up the parsing itself.

5.6 Shifting Calculation to a pre-analysis stage: pre-compilation

Another form of pre-compilation (one more important for speeding up the grammar) involves doing much of the calculation usually done during analysis in a pre-analysis run. To demonstrate this, let's consider the system network in Figure 3. A system network is an inheritance tree, each node towards the right 'inherits' all the features between it and the root (its 'path'). In more complex system networks (such as those in Nigel), the calculation of paths consumes some time, and the calculation for a given feature may be repeated often.

A pre-compilation technique we use is to pre-calculate the path of each system in the network, and store it as a field in the System table.

Another technique is to compile out a list of the possible function bundles defined by the grammar. As shown in Figure 4, each unit has one or more functions assigned to it. This set of functions is called the unit's *function bundle*. In a bottom-up parse, calculation of the function bundle of a particular unit is a costly business. To get around this, we compile out a list of all possible function bundles before analysis begins. Each function bundle has constraints attached to it (what classes the unit must have which fills the functional slot, what type of unit the unit function bundle can be a constituent of).

Cost and Benefits of Pre-Compilation: There is a trade-off connected to the level of pre-compilation: work done in pre-compilation may be done needlessly, not drawn upon in the actual analysis (as when we compile resources for interrogatives, which may not appear in the text). The corresponding advantage of pre-compilation occurs when the same or similar analysis is repeated. With pre-compilation, the calculations

are only done once. The pre-compiled grammar can be used to parse any number of clauses, and even saved away to a file for later use.

The trade-off can also be seen in terms of the amount of work done before versus during analysis. A parser using pre-compilation is slower to boot up, but once up, will parse clauses faster.

The advantage of this lies in the relative cost of time before and during analysis. For our purposes, text analysis needs to approximate real-time, since a human interactant will be involved (an editor). The same requirement holds for Natural Language Interfaces, and perhaps other parsing applications. By moving calculations from the human-involved stage into a pre-compilation stage (which should not need human supervision), human time is saved.

However, if grammar development is on-going, frequent re-compilation of the resources will be necessary. This could be problematic.

5.7 Handling Disjunction in Unification

A System network can be viewed as a logic system -- a system is a statement of disjunction between a set of features, simultaneous systems introduce conjunction, and delicacy represents logical implication.

One technique for unification with disjunction in a logical expression involves bringing all disjunction to the highest level, resulting in a formula which is a single disjunction of conjunctive forms. This form, called *disjunctive normal form* (DNF), is achieved by multiplying out the disjunction in the logical form:

(AvB) & (CvD) => (A&C) v (A&D) v (B&C) v (B&D)

Now, obviously if too much disjunction is present in the logical form, combinatorial explosion will make the expansion too expensive time-wise. The expansion process requires exponential time in the worst case.

Other approaches which avoid this exponential time have been developed. Kasper (1987, 1989) has developed one such method. Kasper's (1989: 2) algorithm:

> is based on an a normal form that divides each description into definite and indefinite components. The definite component contains no disjunction, and the indefinite component contains a list of disjunctions that must be satisfied.

Techniques such as these allow a far faster unification that is possible using a DNF-based unification.

6. Conclusion

In this paper, we have discussed some central aspects of a discourse analyser. We have shown the need for a functional grammar that can support rich descriptions that can be used by the discourse analyser. The parsing grammar is a systemic-functional one. Compared to traditional and formal theories of grammar, systemic-functional theory expands the domain of grammar along three dimensions:

(i) it is not based on structure in the first instance but on **choice** -- the organization of grammar as a strategic resource. Consequently, any parse

with a systemic-functional grammar will yield not only a structural specification but also a systemic one (features selected from a system network -- a kind of subsumption lattice); and the systemic specification is an important step towards semantics (cf. Halliday, 1966, on 'deep grammar').

(ii) it is explicitly functional in its interpretation of grammar. Structures are not sequences of classes such as nominal group + verbal group + prepositional phrase, but rather configurations of functions. Moreover, it is **multi-functional**. Any structural function derives from one of three simultaneous metafunctional (ideational, interpersonal, and textual); and any grammatical unit such as the clause is simultaneously structured by all three.

(iii) it is not only based on the overt categories of traditional grammar (such as case, tense, number; presence/ absence of an 'object') but also on covert categories -- what Whorf (1956) called cryptotypes. The **crypto-grammar** of English includes the relations of projection and expansion for combining clauses into complexes and the process types material, mental, verbal and relational in the transitivity system of the grammar.

These three expansions of the territory of the grammar all serve to show that the grammar is semantically natural.[9] In Section 4, we illustrated how the linguistic examiner of the discourse analyser can use the information derived from systemic-functional parsing to infer further information about the discourse passage analysed.

Using a grammar capable of supporting comprehensive, rich descriptions raises the issue of complexity. The techniques for managing complexity discussed in Section 5 are some which the two Nigel parsers mentioned here are using to achieve real-time parsing given the complexity of the grammar. Both parsers are still under development,, so it is not yet known whether this goal will be achieved. If nothing else, it is hoped that this section has provided a useful exploration of various time-saving techniques.

References

Bateman, John & Christian Matthiessen.1989. The text base in generation. Paper presented at conference of Language and text research, Xi'an Jiaotong University, Xi'an; to appear in selected papers from the conference, ed. by H. Bluhme. Amsterdam: Benjamins.

Bateman, John & Christian Matthiessen.in press. Systemic Linguistics and Text Generation: Experiences from Japanese and English. London: Frances Pinter.

Fries, Peter H. 1983. On the status of theme in English: arguments from discourse. J S Petofi & E Sozer [Eds.] *Micro and macro Connexity of Texts.* Hamburg: Helmut Buske Verlag (Papers in Textlinguistics 45). 116-152. [republished in Forum Linguisticum 6.1. 1-38]

[9] The theory of grammar also includes the notion that grammar is expanded through grammatical metaphor (Halliday, 1985: Ch. 10). We have not discussed this topic here but it is also important to a discourse analyzer.

Halliday, M. A. K. 1966. Some notes on "deep" grammar. Journal of Linguistics 2.1: 57-67.

Halliday, M.A.K. 1978. Language as social semiotic. The social interpretation of language and meaning. London: Edward Arnold.

Halliday, M.A.K. 1984. Language as code and language as behaviour: a systemic-functional interpretation of the nature and ontogenesis of dialogue. In R. Fawcett, M.A.K. Halliday, S. Lamb & A. Makkai (eds), The semiotics of culture and language. Volume 1. London: Pinter.

Halliday, M. A. K. 1985. *An Introduction to Functional Grammar,* London: Edward Arnold.

Hjelmslev, Louis. 1943. Omkring sprogteoriens grundlaeggelse. Akademisk Forlag.

Kasper, Robert. 1987. *Feature Structures: A logical Theory with Application to Language Analysis*. PH.D. dissertation, University of Michigan

Kasper, Robert. 1988a. "An Experimental Parser for Systemic Grammars", *Proceedings of the 12th Int. Conf. on Computational Linguistics*, Budapest.

Kasper, Robert. 1988b. "Parsing with Systemic Grammar".

Kasper, Robert. 1988c. Systemic Grammar and Functional Unification Grammar. In J. Benson & W. Greaves (eds). Systemic Functional Approaches to Discourse: Selected Papers from the Twelfth International Systemic Workshop. Norwood: Ablex. Also as ISI/RS-87-179.

Kasper, Robert. 1989. "Unification and Classification: An Experiment in Information-Based Parsing" Paper presented at International Parsing Workshop, 1989.

Kay, Martin. 1979. Functional Grammar. In Proceedings of the FIfth Annual Meeting of the Berkeley Linguistic Society.

Kay, Martin. 1985. "Parsing In Functional Unification Grammar" in *Natural Language Parsing*, Dowty D., Karttunen L., and Zwicky A. (eds.) Cambridge University Press, Cambridge, England.

Mann, William C. 1983. Inquiry semantics: a functional semantics of natural language grammar. In the Proceedings of the First Annual Conference of the European Chapter of the Association for Computational Linguistics.

Mann, William C. & Matthiessen, Christian 1983. Nigel: a systemic grammar for text generation. Marina del Rey, CA: USC/ Information Sciences Institute (RR-83-105).

Mann, William C. & Matthiessen, Christian 1985. Demonstration of the Nigel Text Generation Computer Program. In Benson & Greaves (eds.). *Systemic Perspectives on Discourse*, Volume 1. Norwood: Ablex.

Marcus, Mitchell P. 1980. *A Theory of Syntactic Recognition for Natural Language*. Cambridge, Mass.: MIT Press. (MIT Ph.D. dissertation, 1977.)

Martin, James R. to appear. Life as a noun: arresting the universe in science and humanities. Eija Ventola (ed.), [Selected papers from the 16th International Systemic Congress]. The Hague: Mouton.

Matthiessen, Christian. 1985. The systemic framework in text generation: Nigel. In Benson & Greaves (eds.). *Systemic Perspectives on Discourse*, Volume 1. Norwood: Ablex.

Matthiessen, Christian. 1987. Notes on the organization of the environment of a text generation grammar. In Kempen (ed.). Natural Language Generation. Dordrecht: Martinus Nijhoff. Also as USC/ISI/RS-87-177.

Matthiessen, Christian. 1988. A systemic semantics: the chooser and inquiry framework. In Benson, Cummings & Greaves (eds.), Systemic Functional Approaches to Discourse: Selected Papers from the Twelfth International Systemic Workshop. Norwood: Ablex. Also as ISI/RS-87-189.

Matthiessen, C. & M.A.K. Halliday. to appear. Systemic Functional Grammar. To appear in F. Peng & J. Ney (eds), Foundations of syntax: an advanced study of current theories of syntax. Amsterdam & London: Benjamins & Whurr.

Gazdar & Mellish Chris ??? chapter 6: "well-formed Substring Tables and Charts"

Sefton, Peter. 1990. Making plans for Nigel or Defining interfaces between computational representations of linguistic structure and output systems: Adding intonation, punctuation and typography systems to the Penman system. BA Honours Thesis, Department of Linguistics, University of Sydney.

Whorf, B.L. 1956. Language Thought & Reality. Selected Writing of Benjamin Lee Whorf, edited by J. Carroll. Cambridge, MA: The MIT Press.

Winograd, Terry. 1983. Language as a Cognitive Process. Volume 1, Syntax. Addison-Wesley.