The HealthDoc Sentence Planner

Leo Wanner

Department of Computer Science
University of Waterloo
Waterloo
Ontario N2L 3G1
Canada
tel: +1-519-885-1211 ext. 5344
fax: +1-519-885-1208
email: lwanner@logos.uwaterloo.ca

Eduard Hovy

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292-6695
U.S.A.
tel: +1-310-822-1511 ext. 731
fax: +1-310-823-6714
email: hovy@isi.edu

Abstract

This paper describes the Sentence Planner (SP) in the HealthDoc project, which is concerned with the production of customized patient-education material from a source encoded in terms of plans. The task of the SP is to transform selected, not necessarily consecutive, plans (which may vary in detail, from text plans specifying only content and discourse organization to fine-grained but incohesive, sentence plans) into completely specified specifications for the surface generator. The paper identifies the sentence planning tasks, which are highly interdependent and partially parallel, and argues, in accordance with [Nirenburg *et al.*, 1989], that a blackboard architecture with several independent modules is most suitable to deal with them. The architecture is presented, and the interaction of the sentence planning modules within this architecture is shown. The first implementation of the SP is discussed; examples illustrate the planning process in action.

1 Sentence Planning

1.1 Introduction

Most current models of text generation include a phase of content selection and organization, usually performed by a text planner or schema application engine, followed by a phase of grammatical surface-form rendering, performed by a sentence generator.

In practice, it is usually found that the sentence generator requires more detailed linguistic information than text planners or schema appliers can provide [Meteer, 1991; Rambow and Korelsky, 1992; Hovy, 1992; Panaget, 1994; Wanner, 1994]. So further planning is required. Following [Rambow and Korelsky, 1992], we call this additional planning task *sentence planning* (even though some operations may cross sentence boundaries). A sentence planner (SP) must specify one of the various possible alternative phrasings at roughly the sentence and clause level. By transforming and augmenting its input, the sentence planner produces representations detailed enough for the surface generator to operate deterministically. Consider an example where the lack of sentence planning results in an awkward text:

(0) *In some instances, an implant wears out, loosens, or fails. If <u>an implant wears out, loosens, or fails</u>, it will have to be removed.*

More appropriate alternatives can be generated when different sentence planning techniques are used:

(1) **Alternative reference:**
In some instances, an implant wears out, loosens, or fails. If <u>this happens</u>, it will have to be removed.

(2) **Alternative lexical choice:**
In some instances, an implant wears out, loosens, or fails. If replacement is needed, it will have to be removed.

(3) **Removal of redundancy (aggregation):**
In some instances, an implant wears out, loosens, or fails, and [] will have to be removed.

In this paper we describe the sentence planner in the HealthDoc project. HealthDoc [DiMarco *et al.*, 1995] was established in early 1995 with the goal of generating customized patient-education documents. It combines existing generation technology—the sentence generator KPML [Bateman, 1995][1] and its input notation SPL [Kasper, 1989]—and new systems, such as the sentence planner described here. The sentence planner embodies a design that we hope has some general applicability in bridging the gap between text planners and sentence generators. Its input is a specification of the desired output content (a patient document) written in *Text Source Language* (TSL), see Subsection 2.3; its output consists of one or more SPL expressions. Its general operation is to recombine, enhance, and refine TSL expressions until they are adequately specific SPL expressions.

1.1 Sentence Planning Tasks

After analysis of a number of patient-education documents, including those on diabetes, cholesterol, and hormone replacement therapy, we have identified the following most important sentence planning tasks:

- **Fine-grain discourse structuring:** Discourse relations (RST relations, for example) that conjoin clause-size pieces in the TSL are still open to considerable variation of expression, such as the inclusion of a discourse marker or the lexical or implicit communication of the discourse relation. See, for example, [Scott and de Souza, 1990] for treatment of ELABORATION, [Vander Linden and Martin, 1995] of PURPOSE, and [Grote *et al.*, 1995] of CONCESSION.
- **Sentence grouping and sentence content determination:** Individual sentences must be delimited; temporal, spatial, and causal nuances of predicates must be determined, and so on [Meteer, 1991; Pustejovsky, 1995; Stede, 1996].
- **Clause-internal structuring:** The order of clause constituents, taxis, and projectivity of propositions within the clause [Hovy, 1992; DiMarco and Hirst, 1993; Panaget, 1994] must be determined; within each sentence, the thematized and focused elements must be identified [Iordanskaja, 1992]; redundancy must be removed [Dalianis and Hovy, 1996a; Dalianis and Hovy, 1996b].
- **Reference planning (endophoric lexical choice):** The particular form of coreference (including anaphora, deixis, and ellipis) and reference must be chosen in order to maintain discourse cohesion [McDonald, 1978; Tutin and Kittredge, 1992; Dale and Reiter, 1995].
- **Exophoric lexical choice:** As argued in [Nirenburg *et al.*, 1989; Meteer, 1991; Wanner, 1994], lexical choice other than linguistic reference should also be considered a sentence planning task, since lexical units predetermine the syntactic structure of a clause, and since salience may be realized by lexical means.

2 The Sentence Planner

2.1 Constraints and Desiderata

Given the nature of the task (namely, the transformation and/or augmentation of partial sentence specifications into full ones), a set of constraints emerge for the design of the Sentence Planner. We believe these constraints to be fairly general; that is, that sentence planners developed for applications other than HealthDoc will have much the same structure. These

[1] KPML stands for *Komet/Penman MultiLingual* and is a development of the Penman system [Penman Project, 1989].

constraints are the following:

1. The SP must transform an underspecified input of deep semantics into a suitably specified output of shallow semantics.

2. The SP must modularize sentence planning tasks as far as possible, to facilitate design clarity, development, and maintenance. Since the sentence planning tasks listed above are not single-step operations, since they do not have to be performed in strict sequence [De Smedt, 1990; Reithinger, 1992], and since the planner's operation is non-deterministic (early choices may undergo subsequent revision), this suggests that each sentence planning task should be implemented by a separate module or by several modules.

3. The intermediate steps of the SP should be accessible and easily interpretable to people building the SP, to enable cross-module interconnection and debugging.

4. The SP must be extensible, allowing new modules to be introduced as a need for them is identified.

5. The level of sophistication of the knowledge within a module must not be constrained by the SP architecture, so that the modules might be crude initially but then can incrementally be refined *without impeding throughput*. To facilitate this, the rules and knowledge resources employed by the SP modules should be represented as declaratively as possible.

Constraints 1 and 3 suggest that the intermediate form(s) of the data during SP operation be some sort of SPL-in-emergence; that is, a frame continually evolving from the more abstract input to the final SPL output(s). One way to achieve this is to see SP modules as tree-transformation engines (viewing SPL and pre-SPL expressions as trees[2]). This means that their effects must be written as tree-rewriting rules in the following general form (see Section 2.3):

$$[\text{pre-SPL}_1 \rightarrow \text{pre-SPL}_2]$$

[2]Strictly speaking, SPL expressions and their origins are directed acyclic graphs, not trees; this does not affect the design in any way.

Naturally, each module must know which tree transformation rule to apply to any given pre-SPL under any given conditions. A suitable mechanism is provided by system networks, used just as in KPML's grammar [Matthiessen and Bateman, 1991]. Each module contains a feature (system) network that discriminates arbitrarily finely to a desired state. At any point in the network, the selection of a feature with an associated tree-rewriting rule causes application of that rule to the current pre-SPL. Thus tree-rewriting rules are *realization statements* in the SP modules (several other realization operators are also supported).

Constraints 2, 4, and 5 suggest that the SP employ a blackboard architecture. As has already been argued by [Nirenburg *et al.*, 1989], a blackboard is best suited to accommodate the flexible order in which modules can take action. It also supports the addition of new modules without requiring the revision of the interfaces of existing modules.

2.2 Sentence Planner Architecture

The architecture of the HealthDoc sentence planner is shown in Figure 1. Solid arrows indicate data flow; dashed arrows indicate control flow. The components are:

1. A set of modules: Discourse Structuring, Content Delimitation, Sentence Structuring, Aggregation, Exophoric Lexical Choice, and Endophoric Lexical Choice.[3]
2. Knowledge sources: the lexicon (essentially KPML's lexicon extended by collocational [Wanner and Bateman, 1990] and qualia [Pustejovsky, 1995] information), the semantic and lexicogrammatical resources as discussed in [Wanner, 1994], the Penman Upper Model [Bateman, 1990], the HealthDoc Domain Model, and the Reader Model of the patient. Not shown in Figure 1 is the history of choices made in the course of processing.
3. The blackboards: the main blackboard, which contains the latest pre-SPL expression(s) and their derivation history, and the control

[3]We also foresee an Ordering module.

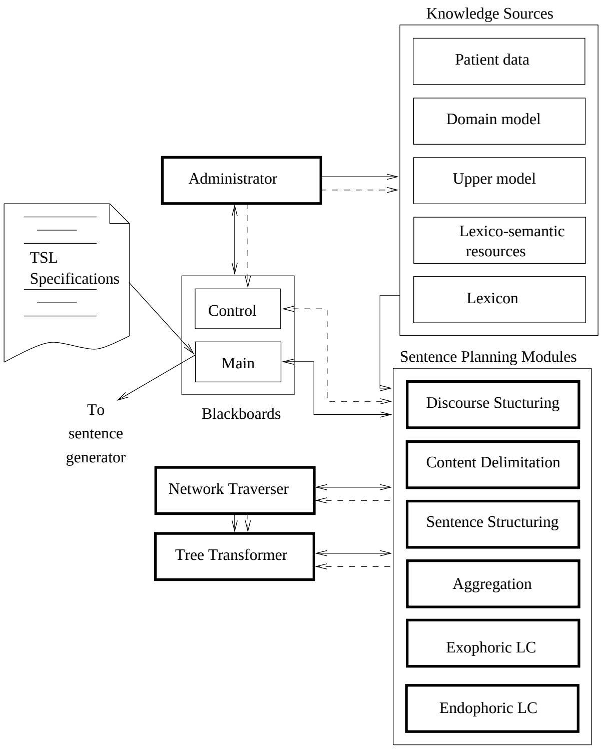

Figure 1: The Architecture of the Sentence Planner in HealthDoc.

blackboard, which contains bookkeeping information such as the flags that signal the status (running/idle/etc.) of each module.

4. The Administrator: the top-level process that invokes modules, updates pre-SPL expressions, manages parallel alternative expressions, etc.

5. The Tree Transformer: the engine that matches the left-hand sides of tree transformation rules to pre-SPL expressions and, when they match, unifies all variables and replaces the matched tree fragments with the right-hand sides of the rules.

6. The Network Traverser: a process that traverses the system network of each module, handles the choice criteria functions (typically, these criteria pertain either to the input pre-SPL or to one of the knowledge resources), and upon reaching tree transformation rules, hands them off to the Tree Transformer.

2.3 The Planning Process

The sentence planning process transforms an input TSL expression(s) into one or more SPL expressions.

Input and output notation. TSL is currently under development. When fully developed, its degree of detail will be variable from one-to-one equivalent with SPL, to an abstract form that contains only the deep semantic frame for a predication (thereby being a notation in which, for example, *commit suicide* has *suicide* as head, rather than *commit*). The flexible degree of detail of the TSL will allow either more semantic or more surface-oriented sentence planning.

For illustration, compare the following two TSLs of varying degrees of abstraction for sentences (1) to (3) above:

A more abstract TSL expression:

```
((D1 / disjunction
      :domain (W / wearout
                 :undergoer (I1 / implant))
      :range (D2 / disjunction
                 :domain (L / loosen
                            :undergoer I1)
                 :range (F / fail
                            :undergoer I1))
      :circumstance (O / occasional))
 (C1 / condition
      :domain (R / remove
                 :patient I1
                 :mode necessity)
      :range D1))
```

Note that coreference information is here encoded by using the same variables.

A more specific TSL expression:

```
((D1 / disjunction
      :domain (W / wearout
                 :actor (I1 / implant
                           :number singular)
                 :tense present)
      :range (D2 / disjunction
                 :domain (L / loosen
                           :actor I1
```

```
                                    :tense present)
                    :range (F / fail
                                    :actor I1
                                    :tense present))
        :nonorienting-action (I2 / three-d-location
                                    :lex instance
                                    :number plural
                                    :determiner some)
        :theme I2)
 (C1 / condition
        :domain (R / remove
        :actee I1
        :modality must)
        :range D1))
```

In the more specific TSL expression, the deep semantic roles have been replaced by surface semantic roles (*:actor, :actee*), and syntactic information (tense) and textual information (theme) have been added. To see the difference between the TSL and SPL, consider the SPL expression for the first sentence in (0)–(3):

SPL output expression:

```
(D1 / disjunction
      :domain (W / nondirected-action
                  :lex wear-out
                  :tense present
                  :actor (I1 / object
                               :lex implant
                               :number plural))
      :range (D2 / disjunction
                  :domain (L / nondirected-act.
                               :lex loosen
                               :tense present
                               :actor I1)
                  :range (F / nondirected-act.
                               :lex fail
                               :tense present
                               :actor I1))
      :nonorienting-action (I2 / three-d-location
                  :lex instance
                  :number plural
                  :determiner some))
```

Overall planning process. The planning process starts with the Administrator, which places a pre-SPL fragment onto the blackboard and activates a module.

Linguistically, it is not possible to prespecify a fixed sequence in which the modules should apply [De Smedt, 1990; Nirenburg *et al.*, 1989]. In some cases, mutual dependencies exist between different linguistic phenomena (i.e., also planning tasks) that cause race conditions or deadlock. We briefly discuss conflict resolution strategies in Section 4. In general, though, we define a (partial) default sequence for the modules: the Discourse Structuring and Sentence Structuring modules run first, and in parallel.[4] They are followed by the Content Delimitation module, and finally by the Exophoric and Endophoric Choice modules, also in parallel. This is in accordance with the increasing delicacy of phenomena the modules deal with. However, we will also experiment with other sequences. A user-specifiable Agenda that determines the ordering of module application has been implemented.

Upon activation, a module removes a pre-SPL from the blackboard, refines and enriches it, and replaces it on the blackboard. The outputs of parallel modules are unified and the unified pre-SPL expression becomes the working copy. The output of non-parallel modules become the working copy immediately. After all modules have run, the constructed SPL on the blackboard is passed to KPML for realization.

In our current implementation, all modules except Discourse Structuring are defined uniformly using system networks.[5] This allows us to adopt an already developed and well-known representation, and the machinery that processes the information encoded in terms of this representation. For example, in [Panaget, 1994; Wanner, 1994], this machinery has been applied to construct an SPL expression from a 'text plan'. Unlike this work, which builds up SPL expressions anew using the text plan as a source of information to guide processing, our SP transforms the text plan *itself* into the SPL. We believe that such a transformation is more transparent to the SP builders, enabling them to in-

[4] "In parallel" here means in arbitrary order.

[5] However, each module is an independent black box. This enables separate and parallel development of each module using any formalism.

spect and manipulate directly the pre-SPL expressions formed during the planning process. The types of planning operations required remain the same in both cases.

To implement transformation rules in the framework of system networks, we define three new realization operators: REWRITE, ADD, and SUPPLANT. Each operator has a different effect on the fragment(s) of pre-SPL that match its left hand side: REWRITE alters its content or structure, ADD adds new information but alters nothing, and SUPPLANT replaces the matched portion with something else (see [Jakeway *et al.*, 1996]). The transformation rules invoke the Tree Transformer when the features they are associated with are chosen. The Tree Transformer then applies the rules to the current pre-SPL expression.

Four general types of transformation are generally required during sentence planning (symbols preceded by *?* are wildcards and match any variable or symbol in the pre-SPL):

1. Augment a pre-SPL expression:

(ADD ((?x / CAUSE) → (:dmarker *causation*)))

in which any fragment of the current pre-SPL containing as head the deep semantic type CAUSE is augmented by the addition of the role *:dmarker* with the filler *causation* (i.e., a cue word signaling causation in the surface form).

2. Modify a pre-SPL expression:

(REWRITE ((?x / RST-RELATION) → (?x / RST-CAUSE)))

in which the head of a pre-SPL fragment is changed from RST-RELATION to RST-CAUSE.

3. Remove a portion of a pre-SPL expression:

```
(SUPPLANT ((?x / CAUSE
                 :role1 y
                 :function (:role1 agent)) →
           (?x / CAUSE
                 :actor y)))
```

in which the intermediate roles *:role1* and *:function* are removed as soon as the process type (and, therefore, also the name of the agent role) are determined, and replaced by appropriate information.

4. Move a portion of a pre-SPL expression:

```
(SUPPLANT ((?x / PROCESS
                 :situation y) →
           (?x / y)))
```

in which the intermediate role *:situation* is removed and its filler moved to occupy the head of the fragment rooted at '?x'. (This occurs when it has been decided that the predicate 'y' — e.g., MOVE—is to be expressed as the verb [*to*] *move*, rather than as the support verb construction *make a move*; now MOVE must be promoted to be the head of ?x in the emerging SPL). Note that 'remove a fragment' always implies 'move a fragment'.

2.4 Inter-Module Conflict Resolution

If a module is activated but is not able to make all the decisions it needs to, or if it makes decisions that are known to be possibly incompatible with those made by other modules later on, there are in general three options for how to proceed:
1. The module must suspend operation until all information required is available.
2. The module must make decisions somewhat blindly and allow backtracking when a decision turns out later to be incompatible.
3. The module must not make the decision but produce all alternatives in parallel, to be winnowed out by later processing.

We do not discuss the first two options since they are standard AI search techniques. The third option is inspired by the treatment of alternatives in statistics-based NLP and the way alternative options are handled in MTM [Mel'čuk, 1981]. In this option, we allow a module to replace an input with *multiple* alternative outputs instead of only one when it cannot make a choice. All such alternative pre-SPLs are spawned and propagated onto the blackboard, so that other modules can work on them all, as parallel alternatives. Should one of the alternatives later turn out to be incompatible with some module's decision, that alternative simply 'dies' and is removed from the blackboard. If, on the other hand, all alternatives remain viable to the end, then the SP has produced more than one valid locution from the input. This option is only feasible if the networks of the modules are not very detailed, i.e., do not lead to an excessive number of alternatives. Although we will experiment with all three modes, our primary intention is to employ mode 1; currently, for implementational reasons, we use mode 2 in its simplest instantiation: when the changes made by a module turn out to be incompatible with those of other modules, the module starts again.

3 The Modules

This section describes the functionality of each module. As an example, we show the creation of the SPL expressions for the sentences (1) to (3). Lack of space prevents us from discussing the criteria and heuristics that are responsible for the individual choices.

3.1 Discourse Structuring Module

The Discourse Structuring module decides upon the way a discourse relation is communicated. So far, three major distinctions are made:
1. **Marker/no-marker**: For example, the CONDITION relation can be marked by *if*, *in case*, etc.
2. **Explicit/implicit**: CONDITION can be communicated explicitly by discourse means and/or lexical means (such as the verb *necessitate*), or implicitly, obtainable via inference.
3. **Nucleus/satellite salience**: In the case of CONDITION, salience can be shifted by change of the order of the condition and effect arguments.

The pre-SPL expression created by the Discourse Structuring module reflects the choices made regarding sentences 1 to 3 above in the type and number of roles to introduce, the role filler information, etc. The following fragment shows the result of the discourse structuring module for the sample sentences:

```
( D1
  (C1 / condition
        :domain (R / remove
                      :actee I1
                      :modality must)
        :range D1)
        :dmarker condition
        :range-ordering first))
```

The fragment D1 is not changed. For fragment C1, the use of a discourse marker has been determined. Also, due to the salience of the condition part of the utterance, the ':range' role will be expressed first.

3.2 Sentence Structuring Module

The Sentence Structuring module determines the structure of the sentences to be encoded in the SPL. This includes:
1. **Sentence boundaries:** If two separate sentences are to be produced, the SPL is split into one per sentence and built up sequentially.
2. **Global sentence structure:** A sentence can be a hypotactic clause complex, a paratactic clause complex, or a simple clause. To determine this, the Sentence Structuring module evaluates whether the predications in the pre-SPL are to be communicated as a sequence or as a composite, complex event. A sequence of events can further be emphasized by a marker.

In our example, the SPL under construction undergoes the following changes:

```
((D1 / disjunction
       :domain W
       :range (D2 / disjunction
                    :domain L
                    :range F
                    :dmarker disjunction)
       :dmarker –
       :nonorienting I2
       :theme I2)
 C1)
```

It has been determined that the first SPL will contain a paratactic clause complex and that there will be a disjunction marker between the L and F fragments. Since a CONDITION that is represented by the roles *:domain* and *:range* is expressed in KPML as a hypotactic clause complex, fragment C1 remains unchanged (note that the aggregation module still has not run to make the changes for (3)).

3.3 Content Delimitation Module

The Content Delimitation module determines the material to be included into each separate SPL expression. At the present stage, this includes the following:
1. **Constituents of a predication**: Constituents that are to be encoded in the pre-SPL. Depending on the contextual and linguistic constraints, roles that are listed in the input might be suppressed and additional ones might be introduced.
2. **Causal, temporal, aspectual nuances**: Nuances of the predication that are to be encoded in the pre-SPL.

The Content Delimitation module primarily introduces realization statements into the pre-SPL expression that constrain the exophoric choice module.

In our example, which starts from a relatively specific TSL, the content delimitation module does not make visible changes: all roles present in the pre-SPL are determined to be realized. Starting from the abstract TSL, intermediate roles *:situation* would have been introduced in the fragments labeled by the variables W, L, F, and R. This syntactically neutral role *:situation* enables the Exophoric Choice module to generate different internal clause structures.

3.4 Aggregation Module

The Aggregation module eliminates redundancy in the pre-SPL by grouping entities that are arguments of the same relation, process, etc., together. The actions of the Aggregation module affect, as a rule, the resulting syntactic structure. In our example, redundancy is apparent in pre-SPL fragments W, L, and F, since their only internal difference is their type, as well as the repetition of the whole D1 in fragment C1.

The actions of the Aggregation module result in the following changes within pre-SPL C1 for sentence (3):

```
(C / conjuntion
     :domain (R / remove
                  :actee I1
                  :lex zeroellipsis
                  :modality must)
     :range D1
     :range-ordering first)
```

3.5 Exophoric Lexical Choice Module

The Exophoric Lexical Choice module chooses lexical units for those entities specified in the pre-SPL that are new in the discourse. More precisely, it makes the pre-SPL more concrete along three lines:
1. **Lexicalization of discourse structure relations**: Discourse relations (and their cue words) may be realizable by lexical means. In our example, the CONDITION marker in (1), (2) is lexicalized as *if*; the DISJUNCTION marker as *or*, and the CONJUNCTION marker as *and.*

2. **Internal clause structuring**: The internal clause structure is predetermined by, among other means, the valency schema of the lexical unit that is chosen to serve as a head of a clause or phrase. With the choice of a head lexical unit, the salience of the arguments is also predetermined (see, e.g., [Wanner and Bateman, 1990]). One of the choices the exophoric lexical choice module makes while generating (2), is the replacement of the fragment D1 in the CONDITION part by *If replacement is needed.* This choice can be made because the KB contains the process of replacement as a possible consequence of an implant being worn out, loosened, or having failed. It is not motivated by the presence of the coreference link in the TSL. The pre-SPL reflects this choice as follows:

```
( D1
  (C1 / condition
       :domain (R / remove
```

```
                    :actee I1
                    :modality must)
      :range (N / need
                    :actee (RE / replacement))
      :dmarker condition
      :range-ordering first))
```

3. **Lexicalization of the entities**: this is traditionally considered to be the task of the lexical choice process. We do not discuss this issue here.

3.6 Endophoric Lexical Choice Module

The Endophoric Lexical choice module chooses lexical units for entities that have already been introduced in the discourse either verbatim or by related entities. If an entity has been introduced verbatim, its next mention can be realized as a personal pronoun, generalized name, definite name, deictic pronoun, or ellipsis. If a related entity has been introduced, the new lexical unit depends on the relation between the two entities; compare [Tutin and Kittredge, 1992].

In our example, if the Exophoric module runs first, the Endophoric module ends up only pronominalizing *implant* in the last clause. If instead the Endophoric module runs first, the SPL produced is (1) rather than (2), i.e., the Endophoric Choice module chooses the phrase *this happens* to refer to *an implant wears out, loosens, or fails.* If we assume this variant, the pre-SPL expression for the second sentence is changed to:

```
(C1 / condition
     :domain (R / remove
                    :actee I1
                    :modality must)
     :range (H / happen
                  :actor (I1 / implant
                              :lex this))
     :dmarker condition
     :range-ordering first)
```

4 Related Research

Related work falls into two areas. The first is sentence planning as a task, including the organization of the planning process. The second covers specific subtasks of sentence planning. Since we have already provided extensive references to work in the second area, and our focus in this paper is not on the detailed presentation of these subtasks, we refrain from discussing it further.

In the first area, our SP appears, at first glance, to closely resemble DIOGENES [Nirenburg *et al.*, 1989]: both systems contain a blackboard with different sentence planning tasks performed by separate modules. However, significant differences exist with respect to processing strategies, including blackboard management and conflict resolution; the assignment of different subtasks to modules; the organization of the modules; and the organization of knowledge resources. This issue is discussed in [Jakeway *et al.*, 1996].

In other related work [Appelt, 1985; Meteer, 1991; Horacek, 1992], several sentence planning tasks are treated within the process of text planning. [Rambow and Korelsky, 1992; Panaget, 1994] have a separate sentence planning component, but they do not separate the specific subtasks of sentence planning into distinct submodules. Necessarily, some subtasks, such as content delimitation, exophoric, and endophoric choice, then play a less prominent role.

5 Conclusion

Individual sentence planning tasks have been the focus of much previous research. A few sentence planners have combined some of these tasks in a single engine. With the HealthDoc sentence planner, we are attempting to build an architecture that supports both the addition of new sentence planning tasks, in the form of new modules, as well as continued growth in sophistication and coverage of individual task performance. This is an ambitious goal. We believe that a blackboard architecture with separate modules, a situation discrimination mechanism such as feature networks, and a continuously transformable internal representation, from TSL input to SPL output using tree transformation operators, to be a promising avenue of research, given the complexity of the problems facing text generators.

Acknowledgments

We would like to thank Bruce Jakeway for implementations as well as the other members of the Sentence Planning Group, Chrysanne DiMarco, Phil Edmonds, and Daniel Marcu, for many fruitful discussions. Special thanks to John Wilkinson, who was one of the "Key Sentence Planners" during the first phase of our work.

References

[Appelt, 1985] D.E. Appelt. *Planning natural language utterances.* Cambridge University Press, Cambridge, England, 1985.

[Bateman, 1990] J.A. Bateman. Upper Modeling: Organizing Knowledge for Natural Language Processing. In *5th International Workshop on Natural Language Generation*, Pittsburgh, PA., 1990.

[Bateman, 1995] J. Bateman. KPML: KOMET-Penman Multilingual Linguistic Resource Development Environment. In *Proceedings of the 5th European Workshop on Natural Language Generation*, Leiden, 1995.

[Dale and Reiter, 1995] R. Dale and E. Reiter. Computational Interpretation of the Gricean Maxims in the Generation of Referring Expressions. *Cognitive Science*, 19(2):233–263, 1995.

[Dalianis and Hovy, 1996a] H. Dalianis and E.H. Hovy. Aggregation in Natural Language Generation. In G. Adorni and M. Zock, editors, *Trends in natural language generation: An Artificial Intelligence perspective.* Springer Verlag, Berlin & Heidelberg, 1996.

[Dalianis and Hovy, 1996b] H. Dalianis and E.H. Hovy. On Lexical Aggregation and Ordering. In *Proceedings of the 8th International Workshop on Natural Language Generation.* Herstmonceux, 1996.

[De Smedt, 1990] K. De Smedt. IPF: An Incremental Parallel Formulator. In Robert Dale, Christopher S. Mellish, and M. Zock, editors, *Current research in natural language generation.* Academic Press, 1990.

[DiMarco and Hirst, 1993] C. DiMarco and G. Hirst. A Computational Theory of Goal-Directed Style in Syntax. *Computational Linguistics*, 19(3):451–499, 1993.

[DiMarco *et al.*, 1995] C. DiMarco, G. Hirst, L. Wanner, and J. Wilkinson. Healthdoc: Customizing patient information and health education by medical condition and personal characteristics. In *Proceedings of the Workshop on Patient Education*, Glasgow, 1995.

[Grote *et al.*, 1995] B. Grote, N. Lenke, and M. Stede. Ma(r)king Concessions in English and German. In *Proceedings of the 5th European Workshop on Natural Language Generation*, Leiden, 1995.

[Horacek, 1992] H. Horacek. An Integrated View of Text Planning. In *Proceedings of the 6th International Workshop on Natural Language Generation*, Trento, Italy, 1992. Springer-Verlag.

[Hovy, 1992] E.H. Hovy. Sentence Planning Requirements for fautomated Explanation generation. In *Proceedings of the Workshop on Explanation Facilities for Model-Based Expert Systems, DIAMOD-Bericht no. 23*, Sankt Augustin, Germany, 1992.

[Iordanskaja, 1992] L. Iordanskaja. Communicative Structure and its Use during Text Generation. *International Forum on Information and Documentation*, 17(2):15–27, 1992.

[Jakeway *et al.*, 1996] B. Jakeway, E. Hovy, and L. Wanner. Specification of the HealthDoc sentence planner. Internal Note, CS Department, University of Waterloo and

USC/ISI, Waterloo and Marina del Rey, 1996.

[Kasper, 1989] R. Kasper. SPL: A sentence plan language for text generation. Technical report, Information Sciences Institute, University of Southern California, 1989.

[Matthiessen and Bateman, 1991] C.M.I.M. Matthiessen and J.A. Bateman. *Text generation and systemic-functional linguistics: experiences from English and Japanese.* Frances Pinter Publishers and St. Martin's Press, London and New York, 1991.

[McDonald, 1978] D. D. McDonald. Subsequent References: Syntactic and Rhetorical Constraints. In *Theoretical Issues in Natural Language Processing—2 (TINLAP).* ACM, New York, 1978.

[Mel'čuk, 1981] Igor Mel'čuk. Meaning-text Models: a Recent Trend in Soviet Linguistics. *Annual Review of Anthropology*, 10:27–62, 1981.

[Meteer, 1991] M.W. Meteer. Bridging the Generation Gap Between Text Planning and Linguistic Realization. *Computational Intelligence*, 7(4):296 – 304, 1991.

[Nirenburg *et al.*, 1989] S. Nirenburg, V. Lesser, and E. Nyberg. Controlling a Language Generation Planner. In *Proceedings of the Joint Conference on Artificial Intelligence*, pages 1524–1530, Detroit, 1989.

[Panaget, 1994] F. Panaget. Using a Textual Representation Level Component in the Context of Discourse or Dialogue Generation. In *Proceedings of the 7th International Workshop on Natural Language Generation*, Kennebunkport, 1994.

[Penman Project, 1989] Penman Project. PENMAN documentation: the Primer, the User Guide, the Reference Manual, and the Nigel manual. Technical report, USC/Information Sciences Institute, Marina del Rey, California, 1989.

[Pustejovsky, 1995] J. Pustejovsky. *The Generative Lexicon.* MIT Press, Cambridge, 1995.

[Rambow and Korelsky, 1992] O. Rambow and T. Korelsky. Applied Text Generation. In *Applied Conference on Natural Language Processing*, Trento, Italy, 1992.

[Reithinger, 1992] Norbert Reithinger. *Eine parallele Architektur zur inkrementellen Generierung multimodaler Dialogbeiträge.* Infix Verlag, St. Augustin, 1992.

[Scott and de Souza, 1990] D. Scott and C. S. de Souza. Getting the Message Across in RST-Based Generation. In R. Dale, C. Mellish, and M. Zock, editors, *Current Research in Natural Language Generation*, pages 47–73. Academic Press, London, 1990.

[Stede, 1996] M. Stede. A generative perspective on verbs and their readings. In *Proceedings of the 8th International Workshop on Natural Language Generation*, Herstmonceux, 1996.

[Tutin and Kittredge, 1992] A. Tutin and R. Kittredge. Lexical Choice in Context: Generating Procedural Texts. In *Proceedings of COLING 92*, pages 763–769, 1992.

[Vander Linden and Martin, 1995] K. Vander Linden and J.H. Martin. Expressing rhetorical relations in instructional text: A case study of the purpose relation. *Computational Linguistics*, 21(1):29–57, 1995.

[Wanner and Bateman, 1990] L. Wanner and J.A. Bateman. A Collocational Based Approach to Salience-Sensitive Lexical Selection. In *5th Natural Language Generation Workshop*, Pittsburgh, PA., 1990.

[Wanner, 1994] L. Wanner. Building Another Bridge over the Generation Gap. In *Proceedings of the 7th International Workshop on Natural Language Generation*, Kennebunkport, 1994.