AN OVERVIEW OF THE PENMAN TEXT GENERATION SYSTEM

William C. Mann
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90291

ABSTRACT

The problem of programming computers to produce natural language explanations and other texts on demand is an active research area in artificial intelligence. In the past, research systems designed for this purpose have been limited by the weakness of their linguistic bases, especially their grammars, and their techniques often cannot be transferred to new knowledge domains.

A new text generation system, Penman, is designed to overcome these problems and produce fluent multiparagraph text in English in response to a goal presented to the system. Penman consists of four major modules: a knowledge acquisition module which can perform domain-specific searches for knowledge relevant to a given communication goal; a text planning module which can organize the relevant information, decide what portion to present, and decide how to lead the reader's attention and knowledge through the content; a sentence generation module based on a large systemic grammar of English; and an evaluation and plan-perturbation module which revises text plans based on evaluation of text produced.

Development of Penman has included implementation of the largest systemic grammar of English in a single notation. A new semantic notation has been added to the systemic framework, and the semantics of nearly the entire grammar has been defined. The semantics is designed to be independent of the system's knowledge notation, so that it is usable with widely differing knowledge representations, including both frame-based and predicate-calculus-based approaches.

1. TEXT GENERATION AS A PROBLEM

AI research in text generation has a long history, but it has had a much lower level of activity than research in language comprehension. It has recently become clear that text generation capabilities (far beyond what can be done with canned text) will be needed, because AI systems of the future will have to justify their actions to users. Text generation capabilities are also being developed as parts of instruction systems [Swartout 83a], data base systems [McKeown 82], program specification systems [Swartout 82, Swartout 83b], expert consulting systems [Swartout 81] and others. A group who recently assessed the state of the art in text generation ([Mann 81a]) concluded that there are four critical technologies which will largely determine the pace of text generation progress in this decade:

1. Knowledge Representation
2. Linguistically Justified Grammars
3. Models of Text Readers
4. Models of Structures and Functions in Discourse

The Penman system has distinct roles for each of these, contributing particularly to 2 and 4.

Penman is intended as a portable, reusable text generation facility which can be embedded in many kinds of systems. By design, it is not tied to a single knowledge domain, to avoid the potential waste of effort inherent in developing single-domain systems whose domain-independent, reusable specific knowledge is not retained. Penman's techniques are adequate to cover the data base domain of McKeown's text generator [McKeown 82], Davey's game transcripts domain [Davey 79], the crisis instructional domain of Mann and Moore [Moore & Mann 79] and others.

2. SYSTEM OVERVIEW

Figure 2-1 shows the principal data flows in Penman. The given goal controls both the search for relevant information (Acquisition) and the organization of that information (Text Planning). Plans are hierarchic, with plans for clauses or sentences at the finest level of detail. These plans include both the logical content to be expressed and how each unit leads the reader's attention through the material. The sentence generator module (Sentence Generation) executes the most detailed level of the plan, thus producing a draft text. The evaluation and revision module (Improvement) evaluates the text, applying measures of quality and comparing the text with the plan for producing it. The module then produces perturbations in the plan to attempt to improve the text. A text is complete when the perturbations suggested by Improvement do not improve the value level identified by the Improvement module.

The major knowledge resources of these modules are also indicated in the figure. In addition to the knowledge notation itself, there is a knowledge base for generic and concrete knowledge of the subject matter and its relation to the world in

This research was supported by the Air Force Office of Scientific Research contract No. F49620-79-C-0181. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research of the U.S. Government.

general, a model of discourse, represented as a collection of patterns and rules which guide text planning, and a model of the reader.

We describe the modules (in order of degree of development rather than in the data flow order described above) in the topical sections below.

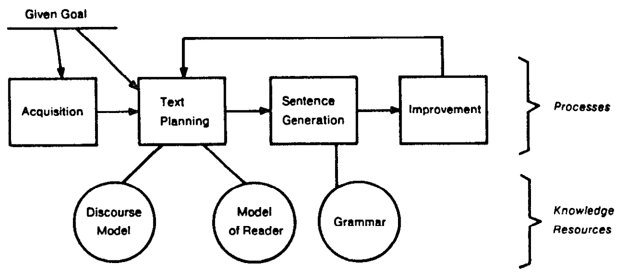

Figure 2-1: Major data flow paths in Penman

3. SENTENCE GENERATION MODULE

The most obvious weakness of text generation systems has been their weak and ad hoc treatment of grammar [Mann 81b]. The one notable exception, Davey's Proteus, produced text in 1973 which remains unsurpassed [Davey 79].

Penman's grammar is called Nigel, named after the child learning English in Halliday's well-known studies [Halliday 75].[1] Nigel is a systemic grammar, like the one in Winograd's SHRDLU, but far more extensive and with an entirely different semantics. Systemic linguistics has greatly influenced many text generation efforts, justifiably, since it is admirably suited for work whose concerns extend beyond the sentence and its structure.

The natural unit of size in a systemic grammar is not a production rule, since systemic grammar does not use production rules. Instead, the unit is the ***system.*** A system is a collection of alternatives called ***grammatical features*** among which one must be chosen if the system is ***entered.*** The grammatical features are not like the category labels found in more conventional grammars. They are not non-terminal symbols representing phrases, and systems do not have to specify the order of constituents in order to have effects.

The grammar is a network of systems. Each system has an entry condition, which is a boolean expression of grammatical features. The entry condition specifies whether the system will be entered, based entirely on choices of grammatical features in other systems. All of the optionality and control of the grammar's generation is in the choices of grammatical features; there is no other kind of optionality or variability. Each pass through the grammar produces a collection of features as its result.

[1]We gratefully acknowledge the past work and present participation of Michael A. K. Halliday, without whom this work would not be possible, as well as the other systemic linguistics who have developed the framework and especially the systemic accounts of English.

There is a separate specification of how syntactic structures are constructed in response to features. This specification, called the ***realization component*** of the grammar, tells how each feature corresponds to a set of operations on a structure-building space which will contain the final result. Constituent order is specified here rather than in the systems.

A collection of chosen grammatical features determines the structure of a syntactic unit, such as a sentence or prepositional phrase. Systemic notation makes it easy to build up such a set of features in parts, out of separately developed sub-collections of features. This fact, that syntactic units can be specified cumulatively rather than category differentiation, turns out to be crucial to the success of the whole framework. Systemic grammars develop their units, especially at the clause and clause-combination levels, by combining ***several independent lines of development, corresponding to several kinds of functional reasoning.***

M. A. K. Halliday, the founder of systemic linguistics, divides the functions of language, i.e., all of its controlled effects, into three ***metafunctions***, which are collections of relatively closely related functions:

1. ***Ideational***: These functions are concerned with the logical and experiential content of language.
2. ***Interpersonal***: These functions are concerned with the stance which the speaker (writer) takes relative to the hearer (reader) and the ideational content. It includes the usual range of speech act theory, but also the speaker's attitudes and representations of his status.
3. ***Textual***: These functions are concerned with the smooth flow, emphasis, ease of comprehension of running text; they become particularly important beyond the level of single sentences.

In Nigel we have extended the notation for choices by associating with each system a ***choice expert***, an explicit process which is able to decide which choice of a feature is correct for any particular set of circumstances (knowledge and text plan). All variability (except word selection) is in the choices, so these choice experts completely determine the resulting language structures.

Choice experts are defined in a notation which makes them independent of the prevailing knowledge representation. This independence is achieved by putting a tight interface around the choice experts, requiring them to obtain all of the information about circumstances by asking questions at the interface, never by searching it out for themselves. The outbound symbolic expressions at the interface are called ***inquiries***, and the semantic approach is called ***inquiry semantics.*** Inquiry responses are atoms, not structures or pointers.[2] The portion of the system outside of the interface is collectively called the ***environment***.

[2]Inquiry semantics has special methods for dealing with the lexicon; lack of space prevents showing them.

An Example of the Use of Inquiry Semantics

As an example, consider the inquiry and response activity at the interface for generating a determiner. In this example, we will treat all of the choosers as an undifferentiated source of inquiries. The example will show how Nigel can obtain the relevant information from its environment without knowing the knowledge notation of the environment. The example is appropriate for selecting the determiner "her" in the sentence "She cancelled her appointment." The example is part of generating a phrase which refers to the appointment.

One of the choosers presents the inquiry (IdentifiabilityQ THING) at the interface, relying on a previously established association of an environment symbol, APPT, with THING. This inquiry says, in effect,

> Does APPT (THING) represent a concept which the speaker expects the hearer to find novel, not previously mentioned or evoked, and thus does not expect the hearer to identify uniquely by reason of the attention which it currently holds, its inherent uniqueness in culture, or its association with an identifiable entity?[3]

Notice that the environment, in this case the part of the system which maintains a model of the hearer, must make an estimate about the hearer's state. The basis for requesting this particular estimate is in linguistic studies of the function of determiners.

The environment's response is the atom 'identifiable'. This establishes that the phrase being developed is expressing something definite which the reader is expected to be able to identify, or possibly create, in his knowledge. A definite determiner will signal this fact to the reader.

After some processing, another chooser asks:

> Is there a specification of proximity within APPT (THING)?

The environment's response is the atom 'noproximity'. At this point, determiners such as "this" and "those," which compete with the possessive determiners, have been ruled out.

The chooser then presents the inquiry (PossessorModificationQ THING), which can be expressed as

> Is there a specification of possessor within APPT (THING)?

The environment responds with the atom 'possessor'. This leads to reserving the determiner slot for some possessive determiner such as "their" or "her," provisionally ruling out the default definite determiner "the."

Having discovered that there is a possessor, it is safe for the grammar to try to evoke a symbol for the possessor. It presents the inquiry (PossessorModID THING). This is a different sort of inquiry, asking for an arbitrary symbol to represent a locus of knowledge in the environment. (It need not be a symbol actually in use in the environment. The grammar will only use the symbol in forming further inquiries.)

The environment responds with the atom 'APPLICANT,' which the grammar associates with the grammatical function DEICTIC in a symbol table it is keeping, the same one which had an association for THING. (We will be assuming below that the table also has associations for SPEAKER and HEARER.)

The grammar then starts to ask questions about the possessor. It presents the inquiry (QuestionVariableQ DEICTIC), which is a question about APPLICANT, expressible as follows:

> Is APPLICANT (DEICTIC) a variable which represents the unspecified portion of a predication?

The environment responds with the atom 'nonvariable'. This rules out determiners such as "whose," as in "You gave him whose money?"

A chooser then presents the inquiry (MultiplicityQ DEICTIC), expressible as

> Is APPLICANT (DEICTIC) inherently multiple, i.e., a set or collection of things, or unitary?)

The environment responds with the atom 'unitary'. This rules out "their."

The next inquiry is (MemberSetQ SPEAKER DEICTIC), which can be expressed as

> Is SELF (SPEAKER) the same as or included in APPLICANT (DEICTIC)?

The response is 'notincluded,' ruling out "my" and one meaning of "our."

The next inquiry is (MemberSetQ HEARER DEICTIC), again with response 'notincluded,' this time ruling out "your."

The next inquiry is (KnownGenderQ DEICTIC), which can be expressed as

> Is the gender, masculine feminine or neutral, of APPLICANT (DEICTIC) known?

The response is 'known'. This finally rules out "the," because the possessor can definitely be expressed in the determiner, so no other expression (such as a prepositional phrase) is needed to express it.

Having established that gender is known, the grammar can ask its value with (GenderQ DEICTIC), the response being 'female.' The grammar selects "her" as the lexical item for the determiner. This selection is the first use of a lexical item in this account.

The knowledge representation side of the interface must implement the domain-relevant subset of the possible inquiries.

[3]Inquiries in Nigel are maintained in an English form as well as a formal form. In these examples, the grammatical function symbol (such as THING) is shown in parentheses, and the corresponding conceptual symbol from the environment (such as APPT) is shown to the left of the parentheses.

The only symbols of the grammar which enter into this implementation are the inquiry operators and the atoms such as 'identifiable' which represent closed-set responses. Modifying the knowledge representation may make it necessary to modify the inquiry operator implementations, but will never make it necessary to change the grammar or its semantics. The implementation task is represented in [Mann & Swartout 83], and the semantics is described in more detail in [Mann 83, Mann 82].

Given the collection of inquiries which the grammar can ask, it is possible to give a precise answer to the question "What knowledge can be expressed grammatically in English?" without presuming a particular knowledge representation or logical formalism.

Nigel is the largest functional grammar of English in a single notation. At the beginning of 1983 it had over 200 systems, each system being roughly comparable to one or a few production rules in other formalisms. There are some gaps in its syntactic capabilities, but nonetheless Nigel is adequate for many text generation tasks. Its diversity can be judged in part by examples: all of the sentence and clause structures of section 4.2 are within Nigel's syntactic range. Nigel is programmed in Interlisp. The inquiry semantics of Nigel is only partly defined at this writing, but we expect that all systems will have choosers before August 1983.

4. TEXT PLANNING MODULE

The text planning module organizes the relevant information into a pattern suitable for presentation. Its operation is based mainly on two regions of Penman's memory: a stored theory of discourse and a model of the reader.

4.1. A Theory of Discourse

Penman's text planning is based on a theory of discourse which regards the text as being organized into regions which act on the reader's state in predictable ways. Regions are composed of other regions, recursively down to the clause level. The composition patterns are stored for the use of the text planner, which is a successive-refinement planner based on Sacerdoti's [Sacerdoti 77]. The planner itself is well-precedented, but this theory and method of organizing the text are new. The composition patterns roughly resemble McKeown's rhetorical schemas [McKeown 82] and are related to Grimes' rhetorical relations [Grimes 75].

Each composition pattern can be seen as an action having preconditions, one or more methods for performing the action, and a set of effects. The entire text is an action, and its parts combine just as actions do into larger actions. Each action is taken because it tends to satisfy a particular set of goals. Decisions on what to include, what sequence of presentation to use, and how to lead the reader are based on the effects (on the reader's state) which the system expects, based on its knowledge of the composition patterns and the reader's state.

The resulting text still contains recognizable recurrent configurations, but--in contrast to sentence structure--these recurrent configurations are not so much recognized by coocurrence relations among the elements. To a much greater degree, the patterns are functional, and are recognized by recognizing joint contribution to a communicative purpose. The patternfulness arises out of reoccurrence of generic purposes, together with recurring selection of ways to satisfy those purposes in communication.

Of course, patterns of desired effects reoccur over long periods of time, and so patterns in text also reoccur. Traditions and habits can be based on these reoccurrences, eventually becoming conventional patterns in language. These conventional patterns never form an adequate (or even suitable) basis for planning text, because as the text pattern becomes a fixed structure it becomes separated from the goals which motivated use of its parts. (So, for example, religious blessings turn into "Adios" and "Goodbye," losing their function as blessings.)

While it may eventually become necessary to incorporate fixed patterns of text arrangement in the design of a text generator, today's technology is not compromised by ignoring them. Instead, the technology can be based on direct original reasoning about how purposes may be satisfied by performing some of the available communicative acts.

4.2. A Model of the Reader

To communicate effectively is to cause conformity of the receiver's state to a description of the desired state. In Penman the governing state descriptions refer to the end state of the reader, the state reached just after the text has been read. (In other kinds of communication, such as entertainment, the transient states may be more important.) So, for example, we may bring the reader into a state of being able to identify prime numbers, or of knowing the name of the king of France. (Entertaining text might be designed to produce a continuous state of amusement.)

We can describe almost the entire reader's state in terms of the elements of knowledge which he holds. In the usual case, the reader's collective knowledge before and after the text is read is a subset of the knowledge which the system holds. We can think of Penman's model of the reader conveniently as a set of independent "colorings" on the knowledge of the system. If we use Red to represent the knowledge of the reader before the text is presented, and we use Green to represent the knowledge he should hold after the text has been read, with other colors which represent intermediate states reached while reading the text, we get a visual analogue of Penman's technique. We recognize that this technique is not adequate for some problems, but it is adequate for many applications. Future systems need to deal with conflict of belief, mutual knowledge and other communication phenomena which the coloring model does not cover [Moore 80, Cohen 77].

The Text Planning Module has not been implemented. The key to implementation is completion of the theory of discourse described in section 4.1. which has been developed extensively but is still in a precomputational stage.

5. ACQUISITION MODULE

Acquisition of the initial stock of information potentially to be expressed is a very domain dependent process; we expect to reimplement it for each application of Penman. Although some selectivity of search can be derived from the given goal, the techniques seem to always be very specific to the application. On the positive side, information acquisition is relatively easy if the knowledge representation in use represents all of the important kinds of knowledge of the host system. The Acquisition Module of Penman has not yet been implemented; experimentation is in a stage which is not committed to a particular expressive task.

6. IMPROVEMENT MODULE

It is surprising how much progress has been made in text generation based on generators which do no more than produce "first draft" text. Neither Davey's generator nor McKeown's attempts to rework the text after generation. The KDS system [Mann & Moore 81] seems to be the only one which has relied heavily on text evaluation and hill-climbing to improve on the quality of the text, using methods which do not require the generator to anticipate the quality of the resulting text.

Penman does not try to anticipate the major determinants of readability in the text it is producing. Sentence length, levels of clause embedding and the like are difficult to anticipate but trivial to measure after the text has been generated. Very simple measures of text quality, including these and also some comparative measures (to see whether the intended content was delivered) seem to be quite adequate as a basis for suggesting helpful revisions in text plans.

The Improvement Module has not been implemented; its design includes particular critic processes, repair proposing processes and repair introduction processes.

7. SUMMARY

The Penman text generation system is designed to be a high quality, portable, multi-domain, multi-representation embeddable module for text generation. It extends the systemic framework, providing a new semantic boundary for grammar, and makes text generation independent of knowledge notations in a new way. Viewed relative to the four critical technologies for text generation research, Penman contributes principally to the form and content of linguistically justified grammars and to models of discourse.

REFERENCES

[Cohen 77] Cohen, P. R., and C. R. Perrault, "Overview of 'planning speech acts'," in *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Massachusetts Institute of Technology, August 1977.

[Davey 79] Davey, A., *Discourse Production*, Edinburgh University Press, Edinburgh, 1979.

[Grimes 75] Grimes, J. E., *The Thread of Discourse*, Mouton, The Hague, 1975.

[Halliday 75] Halliday, M. A. K. , *Learning How to Mean*, Edward Arnold, 1975 .

[Mann 81a] Mann, William C., et al., *Text Generation: The State of the Art and the Literature*, USC/Information Sciences Institute, RR-81-101, December 1981. Appeared as *Text Generation* in April-June 1982 AJCL

[Mann 81b] Mann, W. C., "Two discourse generators," in *The Nineteenth Annual Meeting of the Association for Computational Linguistics*, Sperry Univac, 1981.

[Mann 82] Mann, W. C., *The Anatomy of a Systemic Choice*, USC/Information Sciences Institute, Marina del Rey, CA. RR-82-104, October 1982.

[Mann 83] Mann, W. C., and C. M. I. M. Matthiessen, *Nigel: A Systemic Grammar for Text Generation*, USC/Information Sciences Institute, RR-83-105, February 1983. The papers in this report will also appear in a forthcoming volume of the *Advances in Discourse Processes Series*, R. Freedle (ed.): *Systemic Perspectives on Discourse: Selected Theoretical Papers from the 9th International Systemic Workshop* to be published by Ablex.

[Mann & Moore 81] Mann, W. C., and J. A. Moore, "Computer generation of multiparagraph English text," *American Journal of Computational Linguistics* 7, (1), January - March 1981.

[Mann & Swartout 83] Mann, W. C. and W. R. Swartout, *Knowledge Representation and Grammar: The Case of OWL and Nigel*, USC/Information Sciences Institute, Marina del Rey, CA 90291 , Technical Report, 1983 . (report in preparation.)

[McKeown 82] McKeown, K.R., *Generating Natural Language Text in Response to Questions about Database Structure*, Ph.D. thesis, University of Pennsylvania, 1982.

[Moore 80] Moore, R., *Reasoning about Knowledge and Action*, SRI International, Artificial Intelligence Center, Technical Note 191, 1980.

[Moore & Mann 79] Moore, J. A., and W. C. Mann, "A snapshot of KDS, a knowledge delivery system," in *Proceedings of the Conference, 17th Annual Meeting of the Association for Computational Linguistics*, pp. 51-52, August 1979.

[Sacerdoti 77] Sacerdoti, E., *A Structure for Plans and Behavior*, Elsevier North-Holland, Amsterdam, 1977.

[Swartout 81] Swartout, W. R., *Producing Explanations and Justifications of Expert Consulting Programs*, Massachusetts Institute of Technology, Technical Report MIT/LCS/TR-251, January 1981.

[Swartout 82] Swartout, W., "Gist English generator," in *Proceedings of the National Conference on Artificial Intelligence*, pp. 404-409, AAAI, August 1982.

[Swartout 83a] Swartout, W. R. et. al., "Workshop on automated explanation production," in *ACM SIGART*, ACM, 1983 . To appear.

[Swartout 83b] Swartout, W., The Gist behavior explainer, 1983. (Submitted to AAAI-83.)