Building Underlying Structures for Multiparagraph Texts

Robert Granville
BBN

1. Introduction

Our experience with MACH-III [Kurland *et al* 1992] showed us that there is more to multiparagraph text than stringing together isolated well-formed paragraphs, not surprising since the same is true of multisentential paragraphs and multiword sentences. The underlying structure of the entire text, depicting interparagraph relationships and emphases, must also be determined for successful generation. This of course implies that we need a formalism for representing interparagraph structure. Fortunately, RST [Mann & Thompson 1987] is capable of representing the needed interparagraph structure as well as intraparagraph structure, giving us the framework for exploring how paragraph structure and total text structure interact and how these structures affect the surface text. However, RST does not specify how to build large structures representing multiparagraph text, or even smaller structures representing sentences. This paper presents an algorithm to construct such multiparagraph structures in the context of a critique by the MACH-III system of a student's performance in troubleshooting the HAWK radar. This critique is based on the functional hierarchy tree (FH tree), which is the heart of the expert system component of MACH-III [Kurland *et al* 1989]. First we describe functional hierarchy as a paradigm for organizing expert system knowledge. Then the algorithm for generating text structures based on this functional hierarchy organization is presented in two parts. The first describes how the higher level RST structure defining the overall organization of the paragraphs and their contents is built. This is an elaboration of the algorithm first presented in [Granville 1993]. The second part describes how the individual paragraph RST structures are filled out, resulting in a complete representation of the desired text. This algorithm was developed as part of the *George* system, on-going work extending MACH-III to improve explanation capabilities. The first part of the algorithm, that which builds the high-level RST structures, has been implemented. The second part is the current subject of the *George* effort.

2. Functional Hierarchy and Explanations

Since our algorithm relies heavily upon the functional hierarchy tree structured organization of the MACH-III expert system, we should begin with a brief description. Functional hierarchy (FH) is a new paradigm for organizing expert system knowledge bases, based on the procedural abstraction principles of Liskov and Guttag [Liskov & Guttag 1986]. Functional hierarchy differs greatly from production rules (the customary basis for an expert system) in that functional hierarchy rules define the actions a system can take, rather than the conditions under which actions may take place. The concept of "action" is expanded to include all actions the system takes, including control decisions, rather than just changes to the database, thereby eliminating the need for a separate control structure. These rules are arranged in a hierarchy, where the action of a rule is defined as a combination of other actions. A complete description of the FH paradigm can be found in [Kurland *et al* 1992].

A student action is deemed either to be GOOD, when the action continues in the current branch of the FH tree, SKIPPING, when an action jumps to another branch of the tree before the current branch is completed, OUT OF TREE, when the action has nothing to do with the problem at hand, and therefore isn't represented in the tree, or REDUNDANT, when an action had already been taken by the student, since none of the various actions in this domain ought to be repeated in one session. There is also RETURN for when a student returns to a branch he skipped from, which is the equivalent of GOOD, and SKIPPING-RETURN, which is a return to a skipped branch at the price of leaving the current branch unfinished, equivalent to SKIPPING. MACH-III includes navigational milestones in its output. These milestones do not represent direct actions the student has taken, but mark the entering, returning to, and completion of branches in the FH tree by the actions the student does take.

In writing any text, the intended audience must be identified and assumptions of background knowledge be made. The intended audience for the MACH-III critiques consists of students who have already had several weeks of training in troubleshooting HAWK radar systems. We can assume they are basically familiar with the troubleshooting task and the components that make up HAWK radars. The chief goal of the MACH-III system is to help the students organize their troubleshooting knowledge with simulated hands-on experience to make them more efficient at the required task.

A common component of generation systems is a database containing assumptions of the reader's knowledge known as the *reader model* (or *listener model* for systems that simulate participants in spoken dialogue). In our system, the required text is *not* a model of interactive dialogue, but rather a written monologue. This means that our reader model does not have to be subject to testing and correcting, since such testing and correcting are by nature interactive. Therefore, the system doesn't have to be careful about maintaining a history of how the RST was built, and which assumptions were brought into play when decisions in building the RST were made.

This is in direct contrast with Moore's system [Moore 1989]. In her work, Moore addressed the problem of a listener not understanding a computer generated response in an interactive dialogue, usually due to a mistaken assumption in the system's listener model. It was imperative for Moore's system to be able to identify where in its plan the system failed to make itself understood and the assumptions that caused it to make the erroneous decisions that resulted in the faulty plan. This way it could correct the assumptions and revise the plan, resulting in clearer text for the listener. In order to do this, Moore's system must build its plan to achieve identified goals, and keep careful records of how the plan was built, what decisions were made, and the factors that went into those decisions.

The *George* system does not need an elaborate component to build a plan for two reasons. The first, as mentioned above, is that the desired output text of *George* is a non-interactive monologue of written text, rather than text that models interactive dialogue, and therefore doesn't need information for revising its text in reaction to a listener's misunderstanding. The second is that the database from which *George* has to generate text is the MACH-III functional hierarchy trees. These FH trees are structured purposely to explicitly reflect the very organization we want *George* to explain. Because of these FH trees, we don't have to build plans to determine text structure, and the job of organizing the text, that is, building RST structures, is greatly simplified.

An observant reader will note that a great deal of structure is being imposed on the knowledge base, significantly more than is usual. This raises the question, can we reasonably expect expert systems to have such carefully structured knowledge bases? Clearly, there has to be an intrinsic organization to expert knowledge; there have to be basic relationships about how pieces of information combine to form larger pieces. Knowing you need to test the local oscillator subsystem of a radar unit, but not knowing the subcomponents that make up the local oscillator subsystem will not be sufficient to accomplish the task. Even if you know how to test each basic component in the radar, not knowing how they combine to form larger subsystems would preclude testing these larger subsystems.

If there has to be an organization to knowledge, the next question is when should that organizing be done. Traditionally, no organizational structure for the facts has been required of the expert system. Responsibility for the structure is usually placed on the generation system. Either the generation system builds the structure to produce a coherent text (e.g. [Hovy 1988]), or plausible structures (such as McKeown's schemas [McKeown 1982]) are built ahead of time, and the system selects one that defines the desired structure.

But we have agreed that to be useful, knowledge has to be organized. Obviously, the expert system is making use of that knowledge, so there has to be some organization to it. And, if the experts system desires to explain its knowledge, and the organization is an integral part of that knowledge, then the expert system must explain this organization, too. But a system cannot explain something it doesn't explicitly represent ([Swartout 1983] [Clancey 1983]), so we must somehow explicitly represent the organization of the system.

The traditional expert system paradigm of a production rule knowledge base doesn't represent this organization well, as many researchers have pointed out over the years ([Clancey, Shortliffe, Buchanan 1979], [Swartout 1983], [Clancey 1983], [Kurland *et al* 1992] to name a few. The chief problem is that the independent production rules are the result of a careful organization of the knowledge on the part of the human designers, but these rules don't explicitly represent that organization. All the organization is lost to the system.

Functional hierarchy is a paradigm that explicitly organizes the knowledge to reflect the structure of the knowledge used by actual experts. Since this organization is explicitly represented, it is readily available for explanation purposes. This knowledge structure is itself something that differentiates experts from novices. Therefore, rather than being a nicety, such organization should be explicitly represented and available in *any* expert system as a fundamental part of the knowledge base.

Finally, a few words should be said about our use of RST in *George*. As was discussed above, most generation systems impose little or no organizational requirements on the facts they take as input, and instead build the text structure or select the text structure from a pool of schemas. Those that build their structures, and use RST as the representation of that resulting structure, have extended RST from its original concept to include the mechanisms for the construction of text structure as well as the representation of text structure (see e.g. [Hovy 1988]). *George* uses only the representation originally introduced in [Mann & Thompson 1987], without any of the mechanisms used for building structure that are commonly thought of as part of RST.

3. High-Level Text Organization

An examination of the desired text may yield some hints about the underlying structure. As we stated above, the purpose of the this text is to describe student actions in the course of a troubleshooting session, and how these actions relate to the organization in the FH trees. The only relationships between these actions are how they combine in order to contribute to or detract from an orderly attempt to isolate a problem in the HAWK radar. In themselves, they are completely independent. Testing one component has nothing to do with testing another outside the context of troubleshooting the entire radar. The way RST organizes events, such as student actions, that have no relationship between them other than they occurred in a specific order is with the SEQUENCE relation.

Because we're assuming our audience is conversant with troubleshooting HAWK radars, we don't have to explain the function of components or how the basic actions are performed. Instead, we must tell the student the system's judgment on each specific action, and how that judgment was made. When an action is deemed good, we must explain why it was good; when it is deemed a mistake, we must explain why it was a mistake. In MACH-III, an action is determined to be good or bad based on how it maneuvers the student through the FH tree. Therefore the navigational milestones generated by MACH-III can serve to help explain the critique of each action. Since the text isn't interactive and we don't have to verify and modify our reader model, we can assume that once we explain the validity of a branch in the FH tree or why a problem violates the FH tree organization, the student understands the concept, and it doesn't have to be explained again.

This leaves the problem of the higher organization of the text, that is, where to break paragraphs in the RST structure. It was argued in [Granville 1990] that the structural organization of a text is as important to the message to be conveyed as is its factual content. Therefore any artificial metric for paragraphs, such as limiting them to a specific number of sentences, must be unsatisfactory.

However, the problem is not so daunting when we consider the purpose of the paragraph, which is to describe one idea or topic. One obvious topic category for our text consists of problems and navigational milestones that must be explained because they are being encountered for the first time. The problem or milestone being explained is obviously the topic of the explanation, and therefore deserves its own paragraph. Since problems and milestones that have already been explained do not get full explanations with subsequent encounters, these further references don't merit paragraph treatment.

This does not completely solve the problem, however. Depending on how the FH tree is set up and previous student actions, we may be presented with a set of actions without problems or milestones to explain, and therefore apparently no natural points to break paragraphs. Nevertheless, the set of actions may be too large to describe in a single paragraph.

Once again, a natural solution appears when we consider the purpose of our text, which is to describe the student's actions and how they relate to properly navigating through FH trees, ultimately to help teach the student the organization of these trees. Since teaching this organization is a high-level goal, structuring the text to reflect this organization whenever possible would be beneficial. By placing actions that are directly related in the FH tree in the same paragraph and putting paragraph breaks between actions that aren't related because they're in different branches, our text will reflect the FH tree organization. However, this criterion may be overridden if the paragraph would consist of only

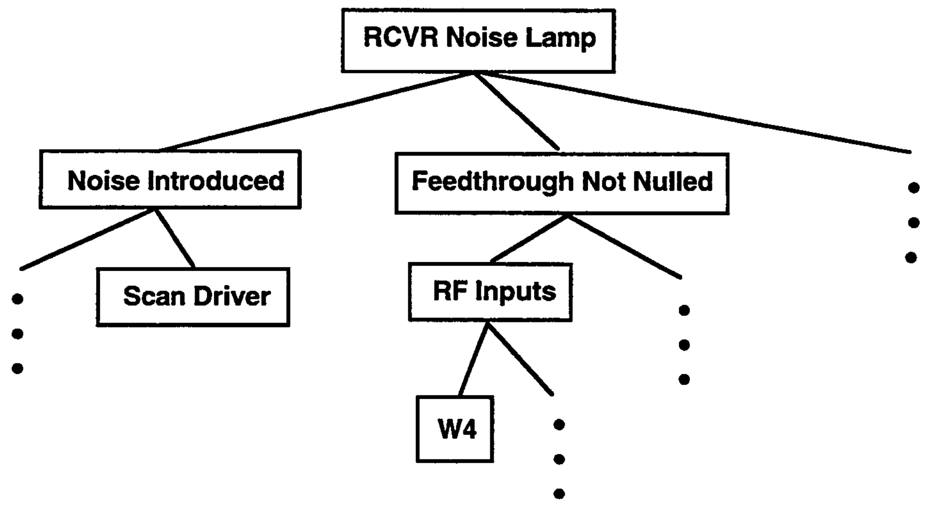

Figure 1. FH Tree for Receviver Noise

one action item that doesn't require elaboration. Otherwise, we could generate a paragraph consisting of one simple sentence.

Finally, there may be circumstances where we don't want a paragraph break, even though we have the conditions we've described. When an FH node has to be explained, we've observed that this requires a central topic and thus its own paragraph. However, we can have two consecutive actions, each requiring explanations on topics that are closely related. For instance, if a student skips to a new branch when neither the skipping problem nor the new branch has yet been explained, we'd have to explain both the milestone of entering the new branch and the skipping problem, and since they both relate to the same action, we want them in the same paragraph. Another example is an action requiring explanation because of a milestone followed immediately by a redundant occurrence of the same action. In both these cases, the central topic is the action, and the two explanations should be in the same paragraph.

From these observations, building a multiparagraph RST structure is straightforward. Let's look at a concrete example. The following is a portion of the output of an actual MACH-III session (run by the author). The test indications (which are not part of the critique output) had indicated a fault with Receiver Noise. A possible text description of these actions is the following. (Phrases are numbered for discussion purposes to correspond with later diagrams.)

(1) The next action you took was to check the W4, (2) one of the RF input cables. (3) If these cables are faulty, feedthrough nulling won't occur properly, (4) resulting in noise in the Receiver, so checking the W4 is certainly a valid step. (5) However, an organized approach to troubleshooting the Receiver is strongly recommended, and checking the W4 at this point left your investigation of whether noise is being introduced unfinished. (6) (You hadn't yet checked the Scan Driver Assembly, which can also introduce noise if not working properly.)

(7) You followed the W4 check by replacing the Scan Driver, (8) further skipping around. (9) By executing a BITE Test, you demonstrated that the fault wasn't with the Scan Driver, (10) and the problem wasn't noise being introduced.

Figure 1 gives the relevant portion of the FH tree for Receiver Noise to assist in following the example. (The complete tree for Receiver Noise, as well as all the other FH trees for HAWK receiver problems, can be found in [Kurland *et al* 1989].)

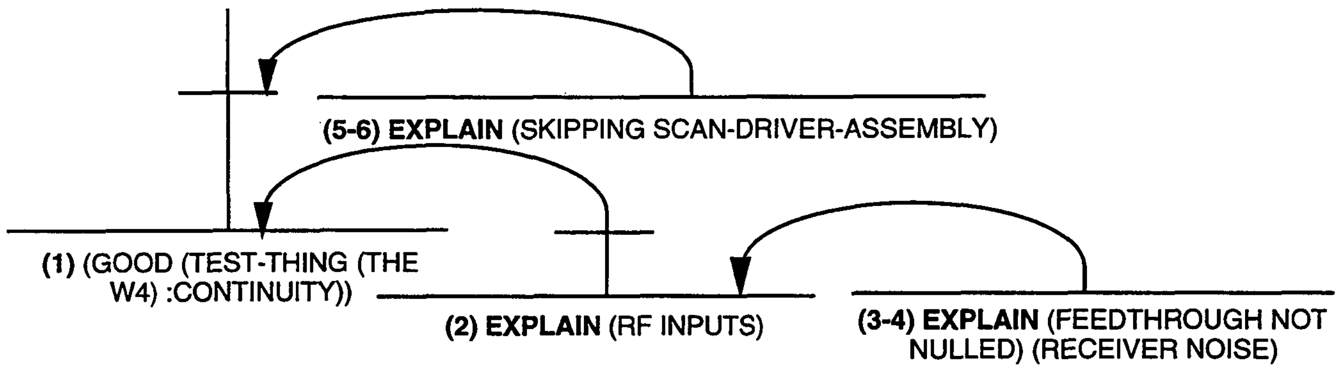
(5-6) EXPLAIN (SKIPPING SCAN-DRIVER-ASSEMBLY)
(1) (GOOD (TEST-THING (THE W4) :CONTINUITY))
(2) EXPLAIN (RF INPUTS)
(3-4) EXPLAIN (FEEDTHROUGH NOT NULLED) (RECEIVER NOISE)

Figure 2. Generated RST Structure (Part 1)

The actual MACH-III output is

```
(START #<NODE FEEDTHROUGH
 NOT NULLED>)
(START #<NODE RF INPUTS>)
(SKIPPING (TEST-THING (THE W4)
 :CONTINUITY) (#<NODE NOISE
  INTRODUCED>))
(RETURNING #<NODE NOISE
 INTRODUCED>)
(SKIPPING-RETURN (REPLACE-
 THING (THE SCAN-DRIVER-
  ASSEMBLY))
(#<NODE RF INPUTS>
 #<NODE FEEDTHROUGH NOT
  NULLED>))
(GOOD (PUSH BITE-TEST-AFTER-
 REPLACE #<DEVICE SCAN-
  DRIVER-ASSEMBLY>))
(FINISHED #<NODE NOISE
 INTRODUCED>)
```

Recall that MACH-III critiques are basically concerned with student actions. Therefore navigational milestones (outputs that begin with START, FINISHED, or RETURNING) are put aside until we have an action with which they can be associated. In our example, the first student action is the testing of the W4, which left the investigation of whether noise is being introduced unfinished. We start a new paragraph with a node for the W4 (the node labeled 1 in Figure 2 below, corresponding to clause 1 in the above sample text). Now we can associate the two START milestones by making RST satellite nodes of the W4 node nucleus (nodes 2 and 3-4 in the diagram). Since neither the concept of feedthrough not being nulled nor that of RF inputs has been described yet, we mark these nodes as needing explanation. However, we also have a SKIPPING problem, and this concept hasn't been described yet, either. Since the problem is with the action we've just explained, we want it in the same paragraph, so we add a contrasting satellite to the complex W4 node to explain skipping (node 5-6).

The next item on the input list is a milestone not related to the W4 test. This is followed by an action item, stating that the Scan Driver Assembly was replaced. With this action, the student has returned to investigating whether noise is being introduced, but leaves the feedthrough not being nulled investigation incomplete. A new paragraph is started since the previous action had been marked for explanation, and this action is not topically related. This paragraph has a node for the Scan Driver Assembly (node 7 of Figure 3) and one for the latest occurrence of a skipping mistake (node 8). Since the skipping problem has been explained (by node 5-6), we don't have to fully explain the problem again, and node 8 is marked as merely requiring mention. Also, we can presume that the concept of noise being introduced has already been explained when we first entered that branch of the FH tree. (We know it had to have been entered because the student left it unfinished when the W4 was tested.) Therefore we don't need to discuss the introduction of noise here. If we assume for the sake of our example that BITE Tests have already been explained (because this is a prevalent action that should be taken after each component replacement), we can simply add a node for this item in SEQUENCE to the current paragraph without full explanation as node 9. Finally we encounter a milestone item showing that the branch for noise

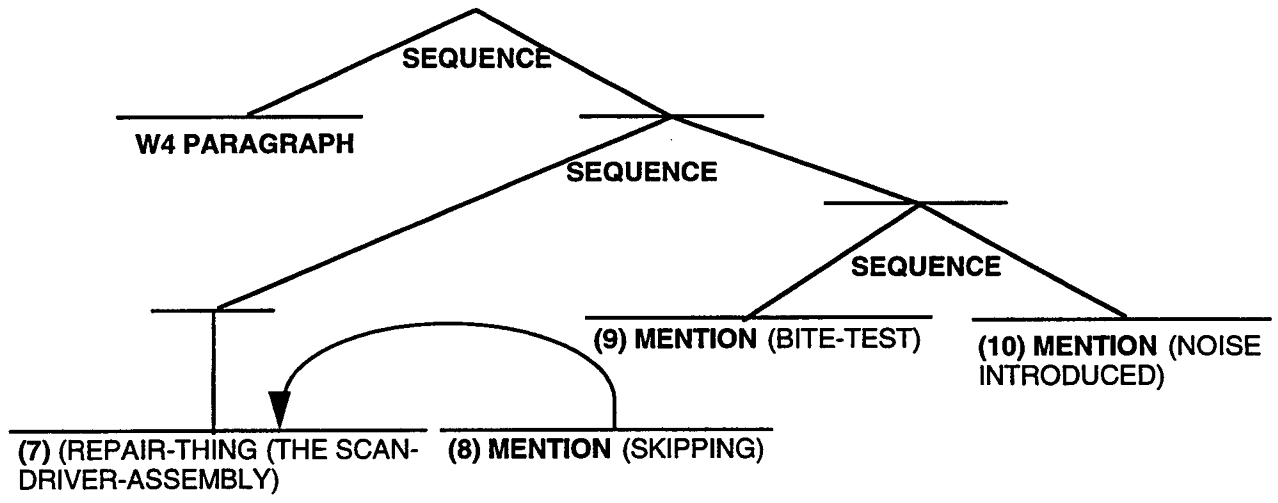

Figure 3. Generated RST Structure (Part 2)

being introduced has been exhausted. Since the concept has been fully described, it doesn't need further explanation here, and node 10 is marked as merely requiring mention.

3. Low-Level Text Organization

We now have a general outline of the paragraph structure in an RST representation, but our work isn't finished yet. We've liberally created a fair number of nodes marked as either MENTION or EXPLAIN, without any discussion about how to act on these marked nodes, or whether they'll have an impact on the final RST structure. It's time to turn our attention to these intraparagraph concerns.

We'll first look at the easiest problem, how to handle nodes marked for MENTION. Recall that these are nodes for problems or milestones that have already been discussed in the generated text, and we're assuming they don't require further elaboration. Each such node can be realized as a simple sentence, or even a phrase. Therefore, they have no impact on the RST structure because the single nodes that represent them are sufficient.

Let's look at an example. Figure 4 is an RST structure that might be generated.

At this point of the RST construction, mixer crystal pairs and the Receiver Assembly have been discussed. A plausible surface realization for this RST might be, "You tested the Reference Mixer Pair, one of the Mixer Crystal Pairs of the Receiver Assembly."

While the realization of this RST pattern is simple enough, the pattern itself deserves a closer look. All the milestone nodes are arranged in the general structure shown in Figure 5. We call this structure the *expository chain*. It represents a statement, A, with a second statement, B, in elaboration of A. Then statement B is elaborated on by statement C. This chain is continued as long as is necessary.

The expository chain is a standard method of presenting arguments, since it represents building a case step by step. As an experiment, the first section of [Bateman 1992] was analyzed. Of the twelve paragraphs in it, all but three are straight expository chains., and those proved to be conjunctions or sequences of expository chains.

4. Conclusion

We have presented an algorithm for constructing multiparagraph structures, both the interparagraph structure defining how paragraphs relate to one another, and the intraparagraph structure defining how each paragraph is organized within itself. This algorithm is dependent on the expert system's knowledge base being organized into a functional hierarchy, and we have argued that rather than being an undue limitation of the generality of the algorithm, such an organization should be an integral part of every expert system's knowledge base.

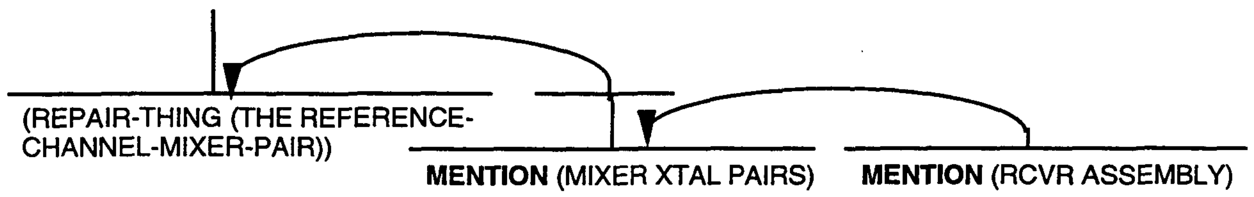
(REPAIR-THING (THE REFERENCE-CHANNEL-MIXER-PAIR))
MENTION (MIXER XTAL PAIRS)
MENTION (RCVR ASSEMBLY)

Figure 4. Sample RST Structure for Reference Channel Mixer Pairs

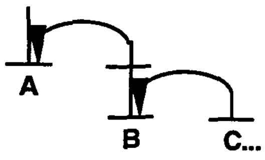

Figure 5. Expository Chain

5. References

Bateman 1992: Bateman, John A., "Towards Meaning-Based Machine Translation: Using Abstractions from Text Generation for Preserving Meaning," in *Machine Translation*, Vol. 7, 1992

Clancey 1983: Clancey, William J., "The Epistemology of a Rule-Based Expert System — a Framework for Explanation," in *Artificial Intelligence*, No. 20, 1983

Clancey, Shortliffe, Buchanan 1979: Clancey, William J., Edward H. Shortliffe, and Bruce G. Buchanan, "Intelligent Computer-Aided Instruction for Medical Diagnosis," in *Proceedings of the Third Annual Symposium on Computer Applications in Medical Computing*, 1979

Granville 1990: Granville, Robert A., "The Role of Underlying Structure in Text Generation," in *Proceedings of the Fifth International Workshop on Natural Language Generation*, 1990

Granville 1993: Granville, Robert, "An Algorithm for High-Level Organization of Multi-Paragraph Texts," in *Intentionality and Structure in Discourse Relations, Proceedings of a Workshop Sponsored by the Special Interest Group on Generation of the Association for Computational Linguistics*, 1993

Hovy 1988: Hovy, Eduard H., "Planning Coherent Multisentential Text," in *Proceedings of the 26th ACL Conference*, Buffalo, New York, 1988

Kurland *et al* 1989: Kurland, Laura C., Robert Granville, Dawn MacLaughlin, *HAWK MACH-III Explanations of the Receiver Troubleshooting Tree*, Technical Report, BBN Systems and Technologies, Cambridge, Massachusetts, 1989

Kurland *et al* 1992: Kurland, Laura C., Robert Alan Granville, and Dawn MacLaughlin, "Design, Development and Implementation of an Intelligent Tutoring System (ITS) for Training Radar Mechanics to Troubleshoot," in *Intelligent Instruction by Computer*, edited by Marshall J. Farr and Joseph Psotka, Taylor & Francis, Washington, DC, 1992

Liskov & Guttag 1986: Liskov, Barbara, and John Guttag, *Abstraction and Specification in Program Development*, MIT Press, Cambridge, Massachusetts, 1986

Mann & Thompson 1987: Mann, William C., and Sandra A. Thompson, *Rhetorical Structure Theory: A Theory of Text Organization*, ISI/RS-87-190, Information Sciences Institute, Marina del Rey, California, 1987

McKeown 1982: McKeown, Kathleen R., *Generating Natural Language Responses to Questions about Database Structure*, PhD thesis, University of Pennsylvania, Philadelphia, Pennsylvania, 1982

Moore 1989: Moore, Johanna Doris, *A Reactive Approach to Explanation in Expert and Advice-Giving Systems*, PhD
thesis, University of California, Los Angeles, California, 1989

Swartout 1983: Swartout, William R., "XPLAIN: A System for Creating and Explaining Expert Consulting Programs," in *Artificial Intelligence*, No. 21, 1983