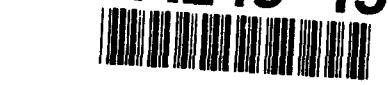
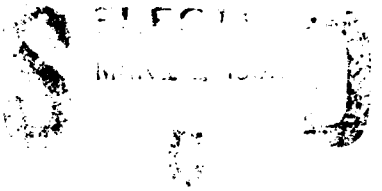




Generating Effective Tutorial Descriptions that use Examples

Vibhu O. Mittal and Cécile L. Paris

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292
U.S.A.

Fax: +1 (310) 823-6714
Phone: +1 (310) 822-1511
e-mail: {mittal,paris}@ISI.EDU

Abstract

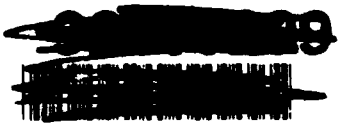
Descriptions of complex concepts often use examples for illustrating various points. This paper discusses the issues that arise in generating complex descriptions in tutorial contexts. Although some tutorial systems have used examples in explanations, they have rarely been considered as an integral part of the complete explanation -- they have been considered as supportive devices -- and are usually inserted in the explanations without any representation in the system of how the examples relate to, and complement the textual explanations that accompany the examples. This can lead to presentations that are at best, weakly coherent, and at worst, confusing and mis-leading for the learner.

Many studies have shown that user comprehension is enhanced significantly if examples are presented in a way that takes into account various features of the accompanying explanation. There is also implicit information present in certain aspects of the presentation, such as the order in which the examples are presented -- the system should understand such criteria and structure its presentation appropriately. In this paper, we consider the generation of examples as an integral part of the overall process of generation, resulting in examples and text that are smoothly integrated and complement each other. We address the requirements of a system capable of this, and present a framework in which it is possible to generate examples as an integral part of a description. We then show how techniques developed in Natural Language Generation can be used to build such a framework.

Keywords: Complex Descriptions, Examples, Natural Language Generation.

Submitted to the Second International Conference on Intelligent Tutoring Systems, Montreal, June 10-12 1992

The first author is also affiliated with the Department of Computer Science, University of Southern California, Los Angeles, CA 90089. We gratefully acknowledge the support of NASA-Ames grant NCC 2-520 and DARPA contract DABT63-91-C-0025.



1 Introduction

There is little doubt that people find examples very beneficial in descriptions of new or complex objects, relations, or processes. Various studies have shown that the inclusion of examples in instructional material significantly increases user comprehension (for e.g., [10,15,24,26,27,33]). Users like examples because examples tend to put abstract, theoretical information into concrete terms they can understand. Few tutoring systems have attempted to make significant use of examples, however. In particular, most systems have not integrated examples in the textual descriptions they use for instructional or explanatory purposes. Examples have been used mostly on their own, independently of the explanation that may also have been provided at that point. However, *examples cannot be generated in isolation, but must form an integral part of the description, supporting the text they help to illustrate*. In this paper, we address some issues in generating descriptions and examples in a coordinated, coherent fashion, such that they complement and support each other.

Section 2 enumerates the issues that must be considered for coherent, effective generation of descriptions. Section 3 discusses some of these issues in more detail. We describe our framework and implementation in Section 4, and conclude in Section 5.

2 Previous Work and Unaddressed Issues

Most previous approaches to the use of examples in tutoring systems focused on the issue of *finding* useful examples. Rissland's CEG system [29], for instance, investigated issues of retrieval versus the construction of examples; Rissland and Ashley's HYPO system [30], on the other hand, investigated more complex and sophisticated techniques for finding examples in a large knowledge base and modifying them along multiple dimensions to fit required specifications; Suther's example generator [32] is also similar to CEG, and investigated efficiency of search techniques in finding examples to modify. Later work by Woolf and her colleagues focused on discourse issues in the context of tutoring systems [39,40]: these studies analyzed the changes in discourse (which included examples), that can occur in an interactive tutoring session (as for instance, the fact that if a hearer did not understand an example, the system should generate an easier example), but did not specifically address issues in generating integrated examples.

Our work builds upon these and other studies (such as the ones reported in [28,31,41]) to study how to provide *appropriate, well-structured and coherent examples in the context of the surrounding text*. It has been shown previously that the difference in effectiveness arising from the use of examples with and without accompanying descriptions is quite significant [3,6--8,11,18,27], and almost doubled in certain cases. The converse -- a presentation of descriptions without the use of examples -- has also been

shown to be not as effective, perhaps because the use of definitions on their own may lead to a learner merely memorizing a string of verbal associations [11]. Thus it is clear that if descriptions are to make use of examples effectively, both the descriptions and the examples must be integrated with each other in a coherent and complementary fashion. Furthermore, there is often a lot of *implicit information* in the sequence of presentation of the examples. The system should be capable of representing this information to structure the sequence correctly.

There are many points that must be considered by any system that attempts to generate effective descriptions of complex concepts:

1. What aspects of information should be exemplified? A system is likely to have detailed knowledge about the concept. It typically needs to select some aspects to present to the user. (This issue has often been raised in natural language generation)
2. When should a description include examples? A few researchers have started to look at this issue [39--41], although much work still remains to be done.
3. How is a suitable example found? Is it retrieved from a pre-defined knowledge base and modified to meet current specifications (as in [31]), or is it constructed (as in [31])?
4. How should the example be positioned with respect to the explanation? Should the example be *within* the text, *before* it, or *after* it?
5. Should the system use one example, or multiple examples? If more than one example is used, how many should be used, and what information should each one convey?
6. If multiple examples are to be presented, how is the order of presentation to be determined?
7. What information should be included in the prompts[1] and how can they be generated?
8. How is lexical cohesion maintained between the example and the text? The text and the example(s) should use the same lexical items to refer to the same concepts.
9. We already know from work in natural language generation that a description is affected by issues such as user-type, text-type, etc. How is the generation of examples (when they should be included, the type of information that they illustrate, and the number of examples) within a text affected by:
   - the prospective audience-type (naive vs. advanced, for instance),
   - the knowledge-type (concepts vs. relations vs. processes),
   - the text-type (tutorial vs. reference vs. report, etc), and

[1] 'Prompts' are attention focusing devices such as arrows, marks, or even additional text associated with examples [5].

```
A list always begins with a left parenthesis.  Then come zero or more pieces of data (called the
elements of a list) and a right parenthesis.  Some examples of lists are:

        (AARDVARK)                                      ;;;     List of a single symbol
        (RED YELLOW GREEN BLUE)                         ;;;     List of symbols
        (2 3 5 11 19)                                   ;;;     List of numbers
        (3 FRENCH FRIES)                                ;;;     List of combined symbols & numbers

A list may contain other lists as elements.  Given the three lists:

        (BLUE SKY)      (GREEN GRASS)       (BROWN EARTH)

we can make a list by combining them all with a parentheses.

        ((BLUE SKY) (GREEN GRASS) (BROWN EARTH))
```

Figure 1: A description of the object LIST using examples (From [36], p.35)

- the dialogue context?

While each of these issues needs to be addressed in a practical system, we shall only discuss some of them here. The issues of positioning the example within a text, the number of examples to be presented, and their order. (Issues #1, #2 and #3 have already been studied to some extent, by other researchers, for e.g., [23, 41, 31, 31, 31]). We now discuss in more detail, the points we are concerned with. We illustrate each point with the description in Figure 1. Although we recognize that the user-type, the text-type and the dialogue context, can all affect the generation of examples, we take as our initial context the generation of a description in a tutorial fashion, for a naive user and as a 'one-shot' response.

3 Integrating Examples in Descriptions

As mentioned previously, a number of studies have shown the need for examples to illustrate descriptions and definitions, as well as a need for explanations to complement the examples presented. Presentation of either on their own is not as useful an approach as one that combines both of them together.

3.1 Positioning the Example and the Description:

Should the example be placed *before*, *within* or *after* the accompanying description? This is an important issue, as it has been reported that there are significant differences that can result from the placement of examples before and after the explanation [6].

Our analyses of different instructional materials reveal that examples usually tend to follow the description of a concept in terms of its critical attributes. Critical attributes are attributes that are *definitional* -- the absence of any of these attributes causes an instance to not be an example of a concept. The examples can then be followed by text which elaborates on features in the examples, unless prompts are included with the examples. If more than one example needs to be presented, the example is usually placed separately from the text (rather than within it). Otherwise, the example is integrated within the text, as in: `An example of a string is "The quick brown fox"`. Following the examples, the description continues with other attributes of the concept, possibly accompanied by further examples.

Sometimes, examples are used as elaborations for certain points which might otherwise have been elaborated upon in the text. For instance, in Figure 1, the LIST could have been described as "A list always begins with a left parenthesis. Then come zero or more pieces of data, *which can be either symbols, numbers, or combinations of symbols and numbers*, followed by a right parenthesis." Instead, the elaboration on the data types is embodied in the information present in the examples. The initial examples in Figure 1 have prompts associated with them, highlighting (number and type information) about the examples. Following these examples, there is some text elaborating on the fact that the elements of a list can also be other lists, and there are examples of lists presented in order to show how these may be combined to form one list.

3.2 Providing the Appropriate Number of Examples

A number of studies have indicated that information transfer is maximized when the learner has to concentrate on as few features as possible [37]. This implies that teaching a concept is most effectively done one feature at a time. This has important implications for example generation: it indicates that examples should try and convey one point at a time, especially if the examples are meant to teach a new concept. Thus, should the concept have a number of different features, a number of examples are likely to be required, one (or a set of) examples for each feature. This is also supported by experiments on differences in learning arising from using different numbers of examples [4, 6, 12, 17].

For example, in Figure 1, each example illustrates one feature of LISTs: that the data can be a single symbol, a number of symbols, numbers, etc. Contrast those examples, with a single example which summarizes most of the features a LIST can have, as given below:

```
(FORMAT T "~ A~ A~ A" 'abcdef 123456 '(abc (123 ("ab"))) ( ))
```

It is important that the system generate an appropriate number of examples, each emphasizing certain selected features.

3.3 Ordering the Examples

Given a number of examples to be presented, the sequencing is also an important matter, because examples often build upon each other. Furthermore, the difference between two adjacent examples is significant. Studies on effects of sequencing can be seen in [6, 10, 13, 14, 17, 34, 35]; they report that proper sequencing can be a very powerful means of focusing the hearer's attention. Consider the sequence of examples on LISTs in Figure 1: the first two examples focus attention on the *number* of elements in a LIST -- they highlight the fact that a LIST can have any number of elements in it; the second and third ones illustrate that symbols are not always required in a LIST -- a LIST can also be made up of numbers; the fourth example contrasts with the third, and illustrates the point that a LIST need not have elements of just one type -- both numbers as well as symbols can be in a LIST at the same time.

It has also been shown that presenting easily understood examples before presenting difficult[2] examples has a significant beneficial effect on learner comprehension [2]. Ordering is important -- it is worth noting that the linguistic notion of the *Maxim of End-Weight* [9, 38], also dictates that difficult and new items should be mentioned after easier and known pieces of information; since there is a direct correlation between the description and the examples, this maxim offers additional motivation for a sequencing of the examples from easy to difficult. Possible orderings may also depend upon factors such as the type of concept being communicated (whether for instance, it is a disjunctive or a conjunctive concept) or whether it is a relation. For example, in Figure 1, the order of examples is determined both by the order in which features are mentioned ("zero or more" and "pieces of data"), and the complexity of examples within each grouping (symbols, followed by numbers followed by combinations of symbols and numbers).

3.4 Generating Prompts for the Examples

Instructional materials that include examples often have tag information associated with each example. This is often referred to as "prompting" information in educational literature [5]. Consider once again, the examples in Figure 1: tags such as "List of Symbols" and "List of combined Symbols and Numbers" represent additional information that is being communicated in the form of prompts. Prompts help focus attention on the feature being illustrated. They often replace long, detailed explanations, and therefore play a role similar to the one of explanation of the examples. However, as they occur in the same sequence as the examples, they capture the change in the examples very efficiently. Consider the example sequence in Figure 1. In this case, the prompts cause the learner to focus on the feature

[2]The terms 'easy' and 'difficult' are easy to misuse, and computationally difficult to specify. These measures of complexity are mostly domain specific -- in the case of LISP, for instance, one measure of difficulty is the number of different grammatical productions that would be required to parse the construct.

that is being highlighted by the sequence; although this information is implicit in the sequence, it may not be noticed by the learner, in the absence of explicit prompts.

3.5 Maintaining Lexical Cohesion between the Text and the Examples

In any given domain, there are likely to be a number of terms available for a particular concept. It is important that the system use consistent terminology throughout the description: both in the definition, as well as in the examples. Consider for instance, the description in Figure 1. Both the examples and the definition use the term 'list', although the terms *list*, *s-expression* and (sometimes) *form* are interchangeable. Difference in terminology can result in confusing messages being communicated to the learner (see for instance [7]). This issue becomes especially important in cases where the examples are retrieved, as the terms used in the example may be different from the terms used in other examples, or the definition. The construction of the textual definition and the examples must thus be done in a coordinated and cooperative manner.

3.6 The Knowledge-Type and its Effect on Descriptions

It has been observed that the type of the knowledge communicated (concept vs. relation vs. a process), has important implications for the manner in which this communication takes place [1,5]. Not only is the information different, but the type of examples and their order of presentation is affected. This is because, if for instance, the system needs to present information on a relation, it must first make sure that the concepts between which the relation holds are understood by the hearer -- this may result in other examples of the concepts being presented before examples of the relation can be presented. The order is usually different too -- there are no negative examples of relations presented in initial teaching sequences (see [5, 1] for detailed arguments on this point).

In this section, we have described in somewhat greater detail, a few of the issues that we identified in Section 2. In the following section, we shall describe a framework for generation that addresses some of the above issues in its presentation strategy. We should mention that our system has only the most rudimentary version of a user-model, and in the description, we shall not discuss aspects such as how dialogue is handled, the effect of the text-type, etc. The description is meant to convey a flavor of how the system processes a goal to describe concepts, and use examples to help achieve its goal.

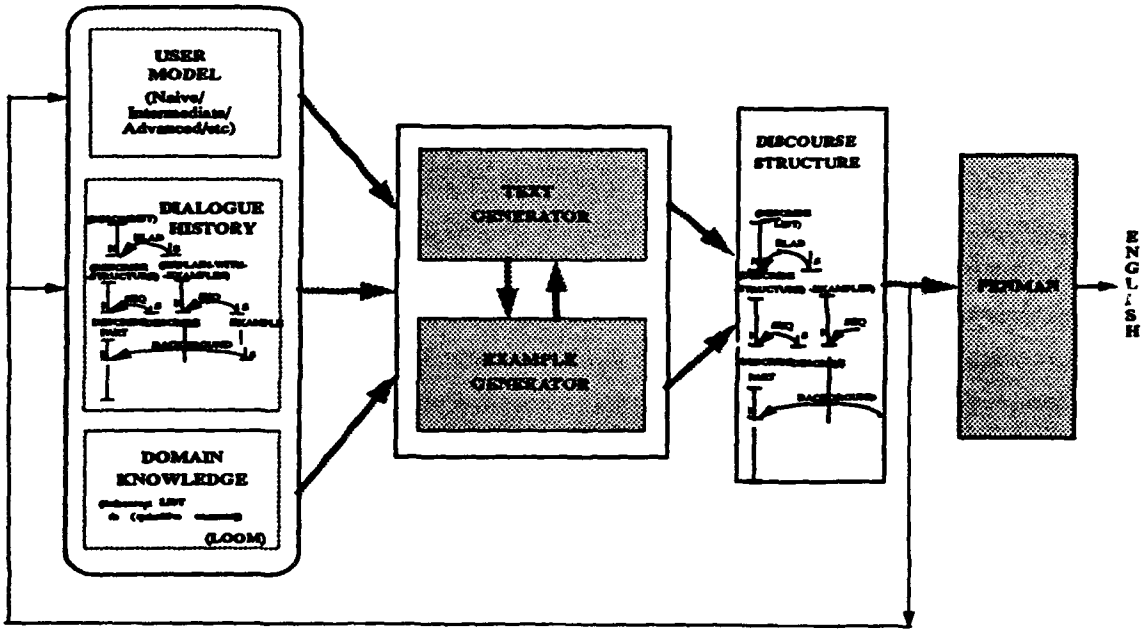


Figure 2: A Block Diagram of the Over-all System.

4 A Framework to Generate Descriptions with Examples

Using techniques developed in natural language generation (NLG), we are working on a framework in which it is possible to integrate examples in a description. This framework will also enable us to investigate and test more carefully the issues raised in Section 2, incorporating and building upon the work of other researchers (e.g., [23, 41, 31, 31, 31]). Our current framework implements the generation of examples within a text-generation system by explicitly posting the goals of providing examples. This text generation system uses a planning mechanism: given a top level communicative goal (such as `(DESCRIBE LIST)`), the system finds plans capable of achieving this goal. Plans typically post further sub-goals to be satisfied, and planning continues until primitive speech acts -- i.e., directly realizable in English -- are achieved. The result of the planning process is a discourse tree, where the nodes represent goals at various levels of abstraction (with the root being the initial goal, such as `(DESCRIBE LIST)` and the leaves representing primitive realization statements, in the form of `(INFORM ...)` statements. In the discourse tree, the discourse goals are related through coherence relations. This tree is then supplied as input to a natural language system (Penman [25]) which converts it into English. A block diagram of the system is shown in Figure 4.

Plan operators can be seen as small schemas (scripts) which describe how to achieve a goal; they are designed by studying natural language texts and transcripts. They include conditions for their applicability. These conditions can refer resources like the system knowledge base (KB), the user

model, or the context (including the dialogue context). A complete description of the generation system is beyond the scope of this paper -- see [19,21,20,22] for more details.

We are adding an example generator to this generation system. Examples are generated by explicitly posting a goal, within the text planning system: i.e., some of the plan operators used in the system include the generation of examples as one of their steps, when applicable. This ensures that the examples embody specific information that either illustrates or complements the information in the accompanying textual description. It is clear that there are additional constraints (for e.g., the user model, text type, dialogue context, etc) that will be needed in any comprehensive implementation of example generation, but we shall investigate those issues in future work. These additional sources of knowledge can be currently added to the system by incorporating additional constraints in the plan operators which reference these resources. Thus, experimenting with different sources in an effort to study their effects is not very difficult.

The number of examples that the system needs to present is determined by an analysis of the features that need to be illustrated. These features depend at least on both the user model and the representation of the concept in the knowledge base. Not all the features illustrated in the examples may be actually mentioned in the text. This is because (as in Figure 1), the description may actually leave the elaboration up to the examples rather than doing it in the text. In Figure 1, for instance, the different data types (numbers, symbols or combinations of both) that may form the elements of a list are not mentioned in the text, but are illustrated through examples. The user model influences the choice of features to be presented. The number of examples is directly proportional to the number of features -- in case of the naive user, there is usually one example per feature. In our framework, the features to be presented are determined based on the domain model and a primitive categorization of the user (naive vs. advanced).

The order of presentation of examples is dependent mainly upon the order of the features being mentioned in the text. In case the text does not explicitly mention the features (as in Figure 1, where the different data types are not mentioned in the text), the system orders them in increasing complexity. Since the ordering in the text is in an increasing order of complexity too (the maxim of end-weight), the least complex examples are always presented first. We have devised domain specific measures of complexity. In the case of LISP for instance, the complexity of a structure is measured in terms of the number of different productions that would need to be invoked to parse the example.

The system maintains lexical cohesion by replacing all occurrences of equivalent terms with one uniform term. This is done as the last step in the discourse tree, before it is used as input to the language generator. There are clearly more issues to be studied to obtain lexical cohesion, but this indicates our framework's ability to at least ensure a consistent use of vocabulary. Our framework is thus centered around a text-planner that generates text and posts explicit goals to generate examples

that will be included in the description. Plans also indicate how and when to generate the prompt information. By appropriately modifying the constraints on each plan-operator, we can investigate the effects of different resources in the framework. In the following section, we shall illustrate the working of the system by generating a description similar to the one in Figure 1.

4.1 A Trace of the System

The system initially begins with the top-level goal being given as (DESCRIBE LIST). The text planner searches for applicable plan operators in its plan-library, and currently finds three that match; one to describe a concept, one to describe a relation, and finally, one to describe a process. Since 'LIST' in our case happens to be a concept rather than a relation (which is the Lisp function 'list'), the first plan-operator is selected. The main features of the concept will be retrieved, and two subgoals will be posted: one to describe the critical features (the left parenthesis, the data elements and the right parenthesis), and another to elaborate upon the difficult features, i.e., the data elements being lists themselves. (The User Model (UM) restricts the choice of the features in this case (naive user) to syntactic ones).

At this point, the discourse tree has only two nodes underneath the initial node of (DESCRIBE LIST) -- namely DESCRIBE-MAIN-FTRS and DESCRIBE-DIFF-FTRS, linked by a rhetorical relation, ELABORATE.[3]

The text-planner now has these two goals to expand:

```
DESCRIBE-MAIN-FEATURES          ;;;  Symbols and numbers
DESCRIBE-DIFFICULT-FEATURES     ;;;  Sub-lists
```

The planner now searches again for appropriate operators to satisfy these goals. The plan operator to describe a list of features indicates that the features should be mentioned in a sequence and elaborated upon by generating an example for each feature. Since the second feature, the data type (number of elements is the first), has three possible types to elaborate upon, the features that the example generator is invoked with, are number of elements, and lists of symbols, numbers and symbols and numbers. The example generator is invoked at this point (our example generator uses techniques similar to the ones developed in [32]). The discourse tree after instantiating this text plan operator (DESCRIBE-MAIN-FEATURES) is shown in Figure 3.

The system now attempts to find a plan capable of generating examples (to satisfy the unexpanded goal in Figure 3. The selected plan operator indicates that the examples for each feature must be

[3]We use rhetorical relations from Rhetorical Structure Theory (RST) [16] to ensure the generation of coherent text -- ELABORATE is one of the relations defined in RST that can connect parts of a text.

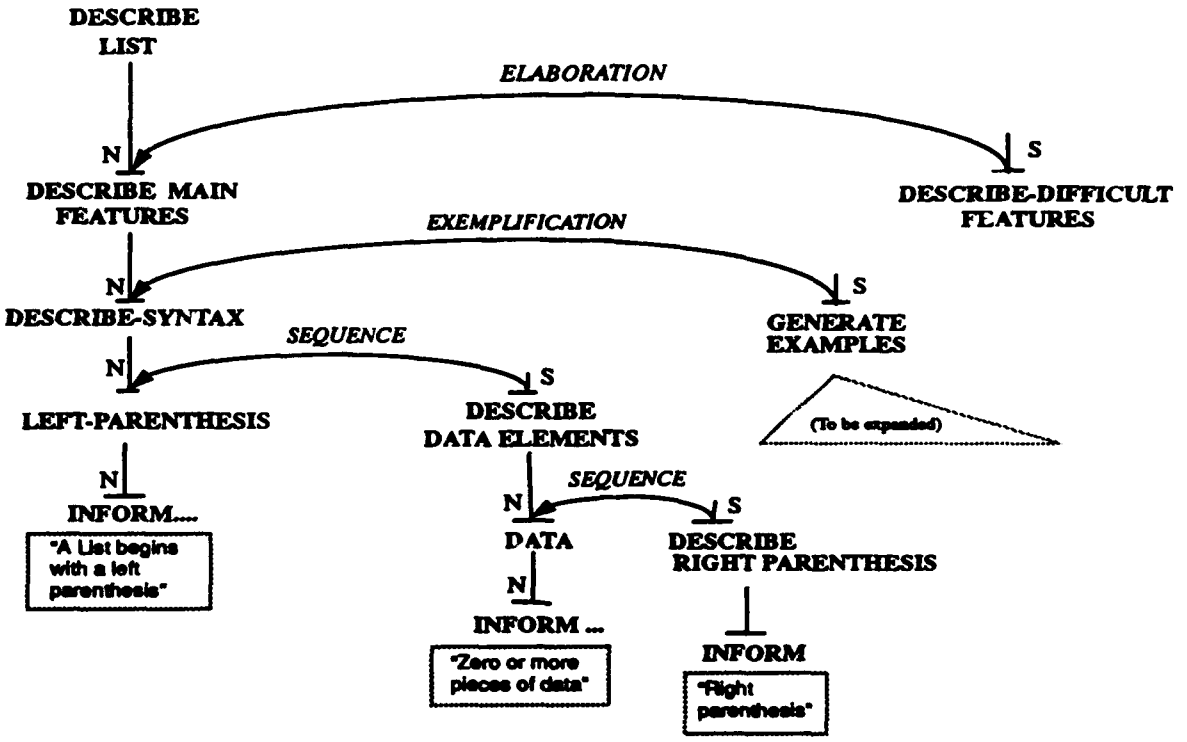


Figure 3: Partial Text Plan for describing the main features of a LIST.

generated in a sequence. Thus, goals to generate examples for each feature are now posted. Each of these goals, in turn is selected and expanded. The first goal, to generate examples for the 'number of elements' feature posts two goals, to produce contrasting examples (one with a single element and another with many elements, the only feature changing between them being the number of elements). These two goals in turn, each expand into goals to generate single examples, as well as the associated prompts. The expanded discourse tree after this feature has been processed, is shown in Figure 4.

The other goals are expanded in similar fashion. The resulting discourse structure is then processed to make final decisions, such as the choice of lexical items. Finally, the completed discourse tree is passed to a a system that converts the INFORM goals into an intermediate form that is accessible to Penman, which generates the desired English output.

5 Conclusions and Future Work

This paper has largely focused on the issues that need to be addressed for an effective presentation of information in the form of a description with accompanying examples. We have identified and

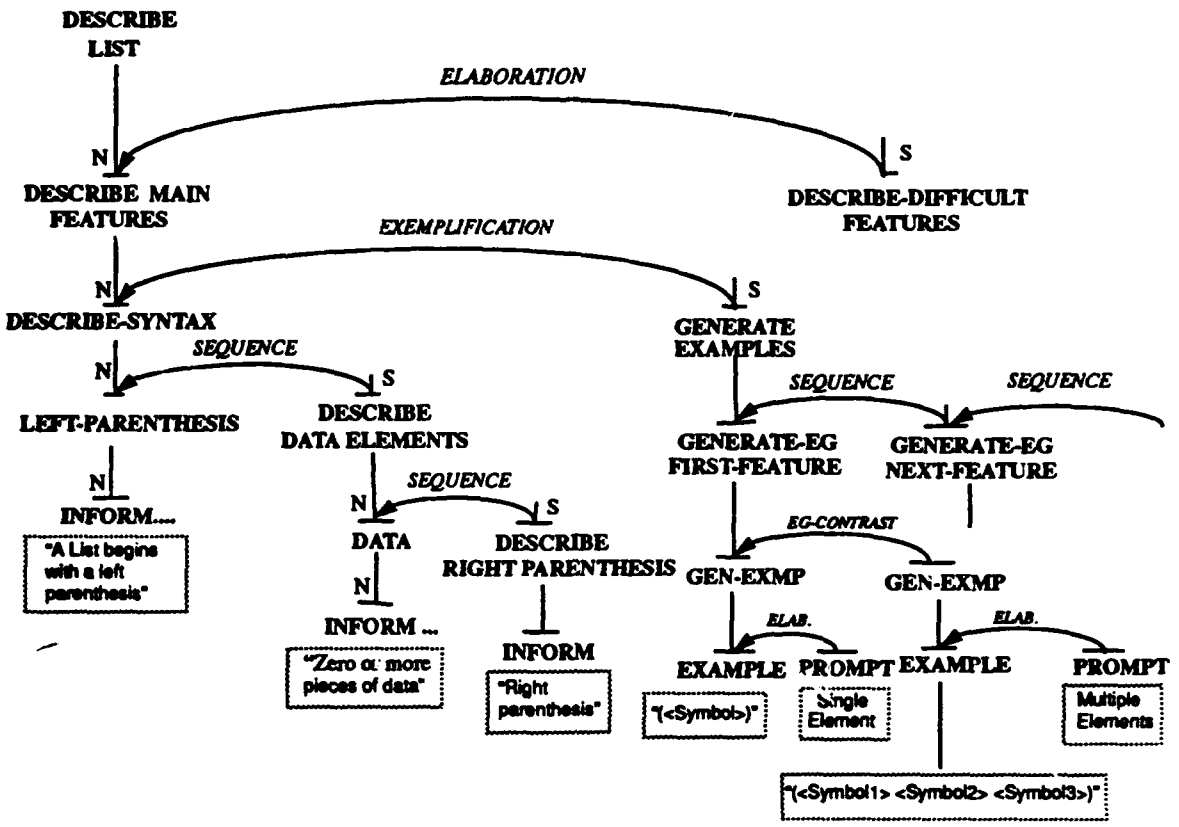


Figure 4: A Fragment of the Discourse Tree.

outlined the various questions that need to be considered, and shown through the use of examples in the domain of LISP, how some of these may be computationally implemented. We have integrated a text generator with an example generator, and have shown how this framework can be used in the generation of coherent and effective descriptions. Our work is based on an analysis of actual instructional materials and books. It illustrates the possibility of planning and generating examples to illustrate definitions and explanations, by considering both of them together during the planning operation. This method, also takes into account various linguistic theories on the order of presentation of facts in the descriptions, and extends them to the presentation of accompanying examples. Our system recognizes the importance of information that is usually implicit in the sequence order, and maintains information in the discourse tree that would allow it to generate prompting information. We have illustrated our framework with a brief trace of an actual description that uses examples. Our work expands on previous work that has been limited to the inclusion of examples with text -- without explicit regard for many of these factors such as sequence, content of each example and prompts.

In future work, we shall investigate questions on issues such as *when* an example should be

generated, and how a previous one may be easily re-used after appropriate modification.

Acknowledgments

We wish to express our gratitude and appreciation to Professors Edwina Rissland and Beverly Woolf for providing us with many references and pointers to related work.

References

[1] Jerome S. Bruner. *Toward a Theory of Instruction*. Oxford University Press, London, U.K., 1966.

[2] Douglas W. Carnine. Two Letter Discrimination Sequences: High-Confusion-Alternatives first versus Low-Confusion-Alternatives first. *Journal of Reading Behaviour*, XII(1):41--47, Spring 1980.

[3] Davida H. Charney, Lynne M. Reder, and Gail W. Wells. Studies of Elaboration in Instructional Texts. In Stephen Doheny-Farina, editor, *Effective Documentation: What we have learned from Research*, chapter 3, pages 48--72. The MIT Press, Cambridge, MA., 1988.

[4] D. C. Clark. Teaching Concepts in the Classroom: A Set of Prescriptions derived from Experimental Research. *Journal of Educational Psychology Monograph*, 62:253--278, 1971.

[5] Siegfried Engelmann and Douglas Carnine. *Theory of Instruction: Principles and Applications*. Irvington Publishers, Inc., New York, 1982.

[6] Katherine Voerwerk Feldman. The effects of the number of positive and negative instances, concept definitions, and emphasis of relevant attributes on the attainment of mathematical concepts. In *Proceedings of the Annual Meeting of the American Educational Research Association*, Chicago, Illinois, 1972.

[7] Katherine Voerwerk Feldman and Herbert J. Klausmeier. The Effects of two kinds of Definitions on the Concept Attainment of fourth- and eighth-grade students. *Journal of Educational Research*, 67(5):219--223, January 1974.

[8] Mark G. Gillingham. Text in Computer-Based Instruction: What the Research Says. *Journal of Computer-Based Instruction*, 15(1):1--6, Winter 1988.

[9] Rachel Giora. On the informativeness requirement. *Journal of Pragmatics*, 12:547--565, 1988.

[10] John C. Houtz, J. William Moore, and J. Kent Davis. Effects of Different Types of Positive and Negative Examples in Learning "non-dimensioned" Concepts. *Journal of Educational Psychology*, 64(2):206--211, 1973.

[11] Herbert J. Klausmeier. Instructional Design and the Teaching of Concepts. In J. R. Levin and V. L. Allen, editors, *Cognitive Learning in Children*. Academic Press, New York, 1976.

[12] Herbert J. Klausmeier and Katherine Voerwerk Feldman. Effects of a Definition and a Varying Number of Examples and Non-Examples on Concept Attainment. *Journal of Educational Psychology*, 67(2):174--178, 1975.

[13] Herbert J. Klausmeier, E. S. Ghatala, and D. A. Frayer. *Conceptual Learning and Development, a Cognitive View*. Academic Press, New York, 1974.

[14] Brenda C. Litchfield, Marcy P. Driscoll, and John V. Dempsey. Presentation Sequence and Example Difficulty: Their Effect on Concept and Rule Learning in Computer-Based Instruction. *Journal of Computer-Based Instruction*, 17(1):35--40, Winter 1990.

[15] James MacLachlan. Psychologically Based Techniques for Improving Learning within Computerized Tutorials. *Journal of Computer-Based Instruction*, 13(3):65--70, Summer 1986.

[16] William Mann and Sandra Thompson. Rhetorical structure theory: a theory of text organization. In Livia Polanyi, editor, *The Structure of Discourse*. Ablex Publishing Corporation, Norwood, New Jersey, 1987. Also available as USC/Information Sciences Institute Technical Report Number RS-87-190.

[17] S. M. Markle and P. W. Tiemann. *Really Understanding Concepts*. Stipes Press, Urbana, Illinois, 1969.

[18] M. David Merrill and Robert D. Tennyson. Concept Classification and Classification Errors as a function of Relationships between Examples and Non-Examples. *Improving Human Performance Quarterly*, 7(4):351--364, Winter 1978.

[19] Johanna D. Moore and Cecile L. Paris. Planning text for advisory dialogues. In *Proceedings of the Twenty-Seventh Annual Meeting of the Association for Computational Linguistics*, pages 203 -- 211, Vancouver, British Columbia, June 1989.

[20] Johanna D. Moore and Cecile L. Paris. Discourse Structure for Explanatory Dialogues. Presented at the Fall AAAI Symposium on Discourse Structure in Natural Language Understanding and Generation, November 1991.

[21] Johanna D. Moore and Cecile L. Paris. User models and dialogue: An integrated approach to producing effective explanations. To appear in the 'User Model and User Adapted Interaction Journal', 1992.

[22] Cecile L. Paris. Generation and Explanation: Building an explanation facility for the Explainable Expert Systems framework. In C. L. Paris, W. R. Swartout, and W. Mann, editors, *Selected papers from the 4th International Workshop on text generation*. Kluwer Academic Publishers, 1990.

[23] Cecile L. Paris. *The Use of Explicit User Models in Text Generation*. PhD thesis, Columbia University, October 1987.

[24] Peter Pirolli. Effects of Examples and Their Explanations in a Lesson on Recursion: A Production System Analysis. *Cognition and Instruction*, 8(3):207--259, 1991.

[25] The Penman Project. *Penman documentation: the Primer, the User Guide, the Reference Manual, and the Nigel Manual*. University of Southern California/Information Sciences Institute, Los Angeles, California, 1988.

[26] Lynn M. Reder, Davida H. Charney, and K. I. Morgan. The Role of Elaborations in Learning a skill from an instructional text. *Memory & Cognition*, 14:64--78, 1986.

[27] Lynne M. Reder, Davida H. Charney, and Kim I. Morgan. The Role of Elaborations in learning a skill from an Instructional Text. *Memory and Cognition*, 14(1):64--78, 1986.

[28] Brian J. Reiser, John R. Anderson, and Robert G. Farrell. Dynamic Student Modelling in an Intelligent Tutor for Lisp Programming. In *Proceedings of the Ninth International Conference on Artificial Intelligence*, pages 8--14. IJCAI-85 (Los Angeles), 1985.

[29] Edwina L. Rissland. Constrained Example Generation. COINS Technical Report 81-24, Department of Computer and Information Science, University of Massachusetts, Amherst, MA., 1981.

[30] Edwina L. Rissland and Kevin D. Ashley. Hypotheticals as Heuristic Device. In *Proceedings of the National Conference on Artificial Intelligence*, pages 289--297. AAAI, 1986.

[31] Edwina L. Rissland, Eduardo M. Valcarce, and Kevin D. Ashley. Explaining and Arguing with Examples. In *Proceedings of the National Conference on Artificial Intelligence*, pages 288--294. AAAI, August 1984.

[32] Daniel D. Suthers and Edwina L. Rissland. Constraint Manipulation for Example Generation. COINS Technical Report 88-71, Computer and Information Science, University of Massachusetts, Amherst, MA., 1988.

[33] Robert D. Tennyson and Ok-Choon Park. The Teaching of Concepts: A Review of Instructional Design Research Literature. *Review of Educational Research*, 50(1):55--70, Spring 1980.

[34] Robert D. Tennyson, M. Steve, and R. Boutwell. Instance Sequence and Analysis of Instance Attribute Representation in Concept Acquisition. *Journal of Educational Psychology*, 67:821--827, 1975.

[35] Robert D. Tennyson and C. L. Tennyson. Rule Acquisition Design Strategy Variables: Degree of Instance Divergence, Sequence and Instance Analysis. *Journal of Educational Psychology*, 67:852--859, 1975.

[36] David S. Touretzky. *LISP: A Gentle Introduction to Symbolic Computation*. Harper & Row Publishers, New York, 1984.

[37] Mark Ward and John Sweller. Structuring Effective Worked Examples. *Cognition and Instruction*, 7(1):1 -- 39, 1990.

[38] Paul Werth. *Focus, Coherence and Emphasis*. Croom Helm, London, England, 1984.

[39] Beverly Woolf and David D. McDonald. Context-Dependent Transitions in Tutoring Discourse. In *Proceedings of the Third National Conference on Artificial Intelligence*, pages 355--361. AAAI, 1984.

[40] Beverly Woolf and Tom Murray. A Framework for Representing Tutorial Discourse. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 189--192. IJCAI, 1987.

[41] Beverly Park Woolf, Daniel Suthers, and Tom Murray. Discourse Control for Tutoring: Case Studies in Example Generation. COINS Technical Report 88-49, Computer and Information Science, University of Massachusetts, 1988.