Generating Object Descriptions: Integrating Examples with Text

Vibhu O. Mittal and Cecile Paris
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, California 90292

January 1993
ISI/RR-93-327

REPORT DOCUMENTATION PAGE

FORM APPROVED
OMB NO. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
January 1993

3. REPORT TYPE AND DATES COVERED
Research Report

4. TITLE AND SUBTITLE
Generating Object Descriptions: Integrating Examples with Text

5. FUNDING NUMBERS
NCC2-250
DABT63-91-C-0025

6. AUTHOR(S)
Vibhu O. Mittal and Cecile Paris

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
USC INFORMATION SCIENCES INSTITUTE
4676 ADMIRALTY WAY
MARINA DEL REY, CA 90292-6695

8. PERFORMING ORGANIZATON REPORT NUMBER
RR-327

9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)

NASA AMES
Moffett Field, CA
94035

ARPA
3701 Fairfax Drive
Arlington, VA 22203

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
T

12A. DISTRIBUTION/AVAILABILITY STATEMENT
UNCLASSIFIED/UNLIMITED

12B. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

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 sued examples in explanation, they have rarely been considered as an integral part of the complete explanation - they have usually been merely supportive devices - and 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.

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

14. SUBJECT TERMS
examples, natural language generation, descriptions

15. NUMBER OF PAGES
8

16. PRICE CODE

17. SECURITY CLASSIFICTION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNLIMITED

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reoprts. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to stay within the lines to meet optical scanning requirements.

Block 1. Agency Use Only (Leave blank).

Block 2. Report Date. Full publication date including day, month,a nd year, if available (e.g. 1 jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element numbers(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	- Contract	PR	- Project
G	- Grant	TA	- Task
PE	- Program Element	WU	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the repor.

Block 9. Sponsoring/Monitoring Agency Names(s) and Address(es). Self-explanatory

Block 10. Sponsoring/Monitoring Agency Report Number. (If known)

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of ...; To be published in... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

DOD - See DoDD 5230.24, "Distribution Statements on Technical Documents."
DOE - See authorities.
NASA - See Handbook NHB 2200.2.
NTIS - Leave blank.

Block 12b. Distribution Code.

DOD - Leave blank.
DOE - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
NASA - Leave blank.
NTIS - Leave blank.

Block 13. Abstract. Include a brief (Maximum 200 words) factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code (NTIS only).

Blocks 17.-19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contins classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

Generating Object Descriptions: Integrating Examples with Text

Vibhu O. Mittal[†*] and **Cécile L. Paris**[*]

[*]USC/Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292
U.S.A

[†]Department of Computer Science
University of Southern California
Los Angeles, CA 90089
U.S.A

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 usually been merely supportive devices – and 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.

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.

1 Introduction

It has long been known that examples are very useful in communication – especially in explanations and instruction. New ideas, concepts or terms are conveyed with greater ease and clarity, if the descriptions are accompanied by appropriate examples (e.g., [Houtz *et al.*, 1973; MacLachlan, 1986; Pirolli, 1991; Reder *et al.*, 1986; Tennyson and Park, 1980]). People like examples because examples tend to put abstract, theoretical information into concrete terms they can understand.

Explanation capabilities are becoming increasingly important nowadays, especially as domain models become larger and more specialized. One requirement in such systems is the ability to explain complex objects, relations or processes that are represented in the system. Advances in natural language generation and research in user modelling have resulted in impressive descriptions being produced by such systems. However, these systems have not, for the most part, concentrated on the issue of generating examples as a part of the overall description. While examples can, in some cases, be *retrieved* from a pre-defined 'example-base' and added to the description, using examples effectively, as an important and a complementary part of the overall description, requires the system to reason with the constraints introduced by both the textual explanation, as well as the examples, in making decisions during the generation process.

There are many issues that must be considered in selecting and presenting examples – in this paper, we shall briefly highlight some of these issues. We view example generation as an integral part of generating descriptions, because examples affect not only other examples that follow, but also the surrounding text. We describe how a text generation system that *plans* text in terms of both intentional and rhetorical goals, can be structured to plan utterances that can include examples in an integrated and coherent fashion. Some of the issues addressed in this regard are equally important in the planning and presentation of *other* explanatory devices – such as diagrams, pictures and analogies.

2 Previous Work and Unaddressed Issues

Most previous approaches to the use of examples in generating descriptions and explanations focused on the issue of *finding* useful examples. Rissland's (1981) CEG system , for instance, investigated issues of retrieval versus construction of examples; Rissland and Ashley's (1986) HYPO system retrieved examples and investigated techniques for modifying them along multiple dimensions to fit required specifications; Suther's example generator [Suthers and Rissland, 1988] 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 design issues of tutoring systems, including determination of when examples are necessary [Woolf and McDonald, 1984;

We gratefully acknowledge the support of NASA-Ames grant NCC 2-520 and DARPA contract DABT63-91-C-0025. The authors may be contacted through electronic mail at: {MITTAL,PARIS}@ISI.EDU

Woolf and Murray, 1987]. However, it did not address issues of discourse generation and the integration of text with examples.

Our work builds upon these and other studies (e.g., [Reiser *et al.*, 1985; Rissland *et al.*, 1984; Woolf *et al.*, 1988]) to study how to provide *appropriate, well-structured and coherent examples in the context of the surrounding text*. It has been shown previously that presentation of descriptions without the use of examples is not effective, perhaps because the use of definitions on their own may lead to a learner merely memorizing a string of verbal associations [Klausmeier, 1976]. The converse – presentation of examples alone – has also been shown previously to be less effective than the use of both examples and descriptions; the use of both almost doubled user comprehension in certain cases (e.g., [Charney *et al.*, 1988; Feldman, 1972; Feldman and Klausmeier, 1974; Gillingham, 1988; Klausmeier, 1976; Merrill and Tennyson, 1978; Reder *et al.*, 1986]). However, text and examples that are not well integrated can cause greater confusion than either one alone (e.g., [Ward and Sweller, 1990]). 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 [Woolf and McDonald, 1984; Woolf and Murray, 1987; Woolf *et al.*, 1988], 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 [Rissland *et al.*, 1984]), or is it constructed (as in [Rissland, 1981])?
4. How should the example be positioned with respect to the surrounding text? 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 probably 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
   - the dialogue context?

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)                      ;;; an atom
(RED YELLOW GREEN BLUE)         ;;; many atoms
(2 3 5 11 19)                   ;;; numbers
(3 FRENCH FRIES)                ;;; atoms & 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 [Touretzky, 1984], p.35)

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., [Paris, 1987; Woolf *et al.*, 1988; Rissland *et al.*, 1984; Rissland and Ashley, 1986; Rissland, 1983]). We now discuss in more detail, the points we are concerned with. We illustrate each point with the description in Figure 1. We are not yet (for this paper) addressing the issues of user-type, the text-type and the dialogue context, though they 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

[1] 'Prompts' are attention focusing devices such as arrows, marks, or even additional text associated with examples [Engelmann and Carnine, 1982].

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 text? 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 [Feldman, 1972].

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. In the lisp domain, for example, a LIST has parentheses as its critical attributes; the elements of the list itself are not. 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 first set of examples in Figure 1 have prompts associated with them, highlighting features (number and type information) about the examples. Following these examples, some text elaborates on the fact that the elements of a list can also be lists. Further examples of lists are used to show how these can be combined to form another list.

3.2 Providing the Appropriate Number of Examples

Studies have indicated that information transfer is maximized when the learner has to concentrate on as few features as possible [Ward and Sweller, 1990]. 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 [Clark, 1971; Feldman, 1972; Klausmeier and Feldman, 1975; Markle and Tiemann, 1969].

This is illustrated in Figure 1, in which each example highlights 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 present, the sequencing is also an important matter, because examples often build upon each other. Furthermore, the difference between two adjacent examples is significant, as proper sequencing can be a very powerful means of focusing the hearer's attention (e.g., [Feldman, 1972; Houtz *et al.*, 1973; Klausmeier *et al.*, 1974; Litchfield *et al.*, 1990; Markle and Tiemann, 1969; Tennyson *et al.*, 1975; Tennyson and Tennyson, 1975]). 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 [Carnine, 1980]. Ordering is thus important – it is worth noting that the linguistic notion of the *maxim of end-weight* [Giora, 1988; Werth, 1984], 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 (e.g., [Engelmann and Carnine, 1982]). 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

[2] The terms 'easy' and 'difficult' are difficult to specify, and are usually highly 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.

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 (tags such as "List of Symbols" and "List of combined Symbols and Numbers") cause the learner to focus on the feature that is being highlighted by the examples.

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 (e.g., [Feldman and Klausmeier, 1974]). 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 (e.g., [Bruner, 1966; Engelmann and Carnine, 1982]). 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 and there are no negative[3] examples of relations presented in initial teaching sequences (e.g., [Engelmann and Carnine, 1982; Bruner, 1966]).

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 describe a framework for generation that addresses some of the above issues. We should mention that our system has only a simple user-model, and in the description, we shall not discuss other aspects of the system such as how dialogue is handled [Moore, 1989a], 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 uses examples to help achieve its goal.

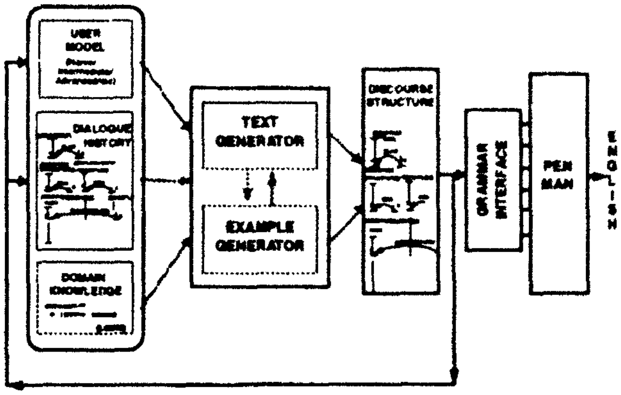

Figure 2: A block diagram of the overall system.

4 A Framework to Generate Descriptions with Examples

Using techniques developed in natural language generation, we are working on a framework within 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., [Paris, 1991b; Woolf *et al.*, 1988; Rissland *et al.*, 1984; Rissland *et al.*, 1984; Rissland *et al.*, 1984]). Our current framework implements the generation of examples within a text-generation system by explicitly posting the goals of providing examples. Our 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, and the leaves representing primitive realization statements, such as (INFORM ...) statements. In the discourse tree, the discourse goals are related through coherence relations. This tree is then passed to a grammar interface which converts it into a set of inputs suitable for input to a natural language generation system (Penman [Mann, 1983]). A block diagram of the system is shown in Figure 2.

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 [Moore and Paris, 1992; Moore, 1989b; Moore and Paris, 1991; Paris, 1991a; Moore and Paris, 1989] for more details.

We are adding an example generator to this generation system. Examples are generated by explicitly posting a goal

[3] 'Positive' examples are instances of the concept they illustrate; 'negative' examples are those which are *not* instances of the concept being described.

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 on the representation of the concept in the knowledge base and the user model. Not all the features illustrated in the examples may be actually mentioned in the text. This is because 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 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 it picks one based on the applicable constraints such as the user model (introductory), the knowledge type (concept), the text type (scientific), etc. The user model restricts the choice of the features in this case (naive user) to syntactic ones. The main features of LIST are retrieved, and two subgoals are posted: one to list the critical features (the left parenthesis, the data elements and the right parenthesis), and another to elaborate upon them.

At this point, the discourse tree has only two nodes: the initial node of (DESCRIBE LIST) – namely LIST-MAIN-FEATURES and DESCRIBE-FEATURES, linked by a rhetorical relation, ELABORATE.[4]

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

```
LIST-MAIN-FEATURES
DESCRIBE-FEATURES
```

The planner searches 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. Three goals are appropriately posted at this point. These goals result in the planner generating a plan for the first sentence in Figure 1. The other sub-goal of DESCRIBE-LIST also causes three goals to be posted for describing each of the critical features. Since two of these are for elaborating upon the parentheses, they are not expanded because no further information is available. A skeleton of the resulting text plan is shown in Figure 3.

The system now attempts to satisfy the goal DESCRIBE-DATA-ELEMENTS by finding an appropriate plan. Data elements can be of three types: numbers, symbols, or lists. The system can either communicate this information by realizing an appropriate sentence, or through examples (or both). The text type and user model constraints cause the system to pick examples. It generates two goals for the two dimensions in which the data elements can vary in: the number and the type. The goal to illustrate the number feature of data elements causes two goals to be generated:

```
GENERATE-EXAMPLE-SINGLE-ELEMENT
GENERATE-EXAMPLE-MULTIPLE-ELEMENTS
```

so as to highlight the difference in number of elements between the two examples. (Each of these goals posts further goals to actually retrieve the example and generate an appropriate prompt, etc.) The example generation algorithm ensures that the examples selected for related sub-goals (such as the two above) differ in *only* the dimension being highlighted.

The goal to illustrate the *type* dimension using examples expands into four goals: to illustrate symbols, numbers, symbols and numbers, and sub-lists as possible types of data elements. (The other combinations possible – numbers and sub-lists, symbols and sub-lists, and all three together – are also possible

[4] We use rhetorical relations from Rhetorical Structure Theory (RST) [Mann and Thompson, 1987] to ensure the generation of coherent text – ELABORATE is one of the relations defined in RST that can connect parts of a text.

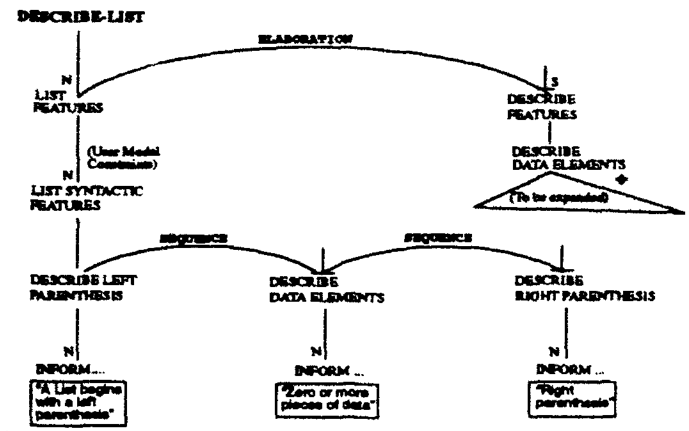

Figure 3: Plan skeleton for listing the main features of a LIST.

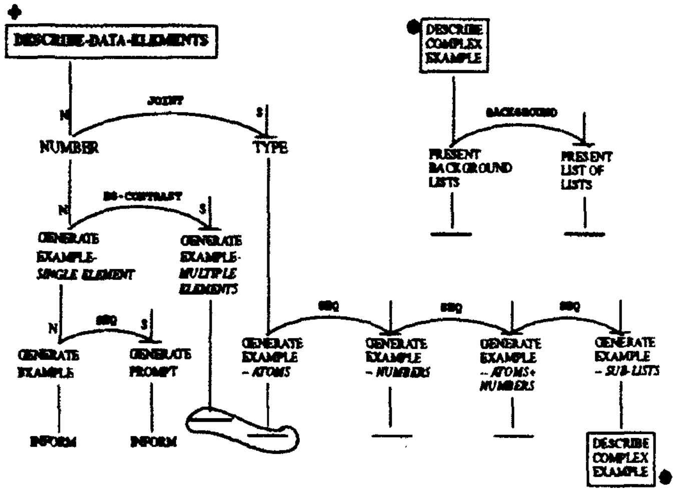

Figure 4: Partial text plan for generating the LIST examples.

data element types, but are not illustrated because of a global constraint that the number of examples generated should not exceed four ([Clark, 1971]) and the system attempts to present small amounts of information at a time (because of the user model). Each of the first three goals posts appropriate goals to retrieve examples and generate prompts. The first goal of generating a LIST of atoms can be collapsed with the previously satisfied goal of generating an example of a LIST with multiple elements in our case.

In the fourth case, the user-model prevents the system from simply generating an example of a LIST which has other LISTs as its data elements. The system therefore posts two goals, one to provide background information (which presents three simple lists), and the other to build a list from these three lists. Heuristics in the system cause the system to generate text for this fourth case, rather than just a prompt. A skeleton of the second half of the complete text plan is shown in Figure 4.

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 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 and other studies on examples. 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

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

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

[Charney *et al.*, 1988] 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. This shows the so called 48combined effect, etc. Check this out.

[Clark, 1971] 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.

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

[Feldman and Klausmeier, 1974] 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.

[Feldman, 1972] 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.

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

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

[Houtz *et al.*, 1973] 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.

[Klausmeier and Feldman, 1975] 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.

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

[Klausmeier, 1976] 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.

[Litchfield *et al.*, 1990] 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.

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

[Mann and Thompson, 1987] 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.

[Mann, 1983] William Mann. An Overview of the Penman Text Generation System. Technical Report ISI/RR-83-114, USC/Information Sciences Institute, 1983.

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

[Merrill and Tennyson, 1978] 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.

[Moore and Paris, 1989] Johanna D. Moore and Cécile 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.

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

[Moore and Paris, 1992] Johanna D. Moore and Cécile L. Paris. Exploiting user feedback to compensate for the unreliability of user models. To appear in the 'User Model and User Adapted Interaction Journal', 1992.

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

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

[Paris, 1991a] Cécile L. Paris. Generation and Explanation: Building an explanation facility for the Explainable Expert Systems framework. In C. Paris, W. Swartout, and W. Mann, editors, *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, pages 49 – 81. Kluwer Academic Publishers, Boston/Dordrecht/London, 1991.

[Paris, 1991b] Cécile L. Paris. The role of the user's domain knowledge in generation. *Computational Intelligence*, 7 (2):71 – 93, May 1991.

[Pirolli, 1991] 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.

[Reder *et al.*, 1986] 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.

[Reiser *et al.*, 1985] 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.

[Rissland and Ashley, 1986] 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.

[Rissland *et al.*, 1984] 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.

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

[Rissland, 1983] Edwina L. Rissland. Examples in Legal Reasoning: Legal Hypotheticals. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 90–93, Karlsruhe, Germany, 1983. IJCAI.

[Suthers and Rissland, 1988] 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.

[Tennyson and Park, 1980] 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.

[Tennyson and Tennyson, 1975] 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.

[Tennyson *et al.*, 1975] 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.

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

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

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

[Woolf and McDonald, 1984] 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.

[Woolf and Murray, 1987] 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.

[Woolf *et al.*, 1988] 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.