Interactive Multimedia Explanation for Equipment Maintenance and Repair

Steven Feiner
Kathleen R. McKeown

Department of Computer Science
Columbia University
New York, New York 10027

Technical Report CUCS-449-89

June 1989

Abstract

COMET (COordinated Multimedia Explanation Testbed) is a research system that we are developing to explore the coordinated generation of multimedia explanations of equipment maintenance and repair procedures. The form and content of all material presented is generated interactively, with an emphasis on coordinating multiple media to allow cross-references between media and to make possible display layout that reflects the fine-grain relationships among the material presented.

COMET's architecture includes multiple static and dynamic knowledge sources, a content planner, a media coordinator, media generators (currently text and graphics), and a media layout manager. Examples are given of the kinds of material processed and produced by each of the components.

1 Introduction

Organizations that make use of complex equipment currently maintain large amounts of hard-copy documentation for use in maintenance, repair, and training. Unfortunately, hard-copy documentation is ill-suited for large-scale applications: it generally does not accommodate users with different skill levels, is clumsy to use and bulky, and is capable of presenting only static graphics and text. For these reasons, a number of organizations have been investigating the use of interactive, computer-based documentation delivery, including hypermedia [13] and expert systems [18]. In both of these areas, however, the material that is ultimately delivered to the user is authored in advance by human authors, while the system determines only which material to present. As a result, these systems require an immense investment in human authoring time to develop text and pictures. In addition, because the information being provided is authored in advance, it cannot cater to the specific user and situation, and at best groups them into overly broad equivalence classes.

Our approach addresses these problems by generating interactively all the material that is presented, taking into account knowledge about the user and the situation. This

means that the user will be presented with exactly the information they need in a form that is tailored to their specific abilities. In order to explore these ideas we are developing COMET (COordinated Multimedia Explanation Testbed), an experimental environment for research in multimedia generation. COMET dynamically determines both the content and form of an explanation. By *form* we refer to the choice of text or graphics for information to be communicated, the coordination of text and graphics within the explanation, the choice of words and sentence structure for text, and the choice of objects, style, viewing, and lighting for graphics. Because it composes the explanation when it is requested, taking the purpose for the request into account, COMET can provide exactly the information needed in a concise, usable form, unlike approaches that rely on preauthored material.

In the following sections, we describe COMET's system organization and application domain, the knowledge sources (both static and learned) that are used for an explanation, the production of explanation content, and the coordinated generation of text and pictures from content.

2 System Organization and Domain

COMET currently consists of the six major components illustrated in Fig. 1. On receiving a request for an explanation[1], the *content planner* uses text plans, or *schemas*, to determine which information should be included from the underlying *knowledge sources* in the explanation. COMET is designed to use four different knowledge sources: a static representation of the domain encoded in LOOM [22], a dynamic representation of the world as influenced by plan execution [2], a rule-base learned over time [8], and a detailed geometric knowledge base necessary for the generation of graphics [32]. The content planner produces the full content for the explanation, represented as a list of logical forms (LFs) [1], which are passed to the *media coordinator*. The media coordinator refines the LFs by adding directives indicating which portions are to be produced by each of a set of media-specific generation systems. COMET currently includes text and graphics generators. The *text generator* and *picture generator* each process the same LFs, producing fragments of text and graphics that are keyed to the LFs they instantiate. This output will be combined by the *layout manager*, which formats the final presentation on the display.

We have selected the US Army AN/PRC-119 radio receiver-transmitter as the domain in which to develop COMET. COMET is currently able to produce coordinated text and graphics explanations of how to carry out a repair/maintenance procedure, textual descriptions of radio components, and graphical displays of the radio that emphasize the physical properties, location, and state of its components. We will use the troubleshooting guide for loss of memory [9] shown in Fig. 2 throughout this paper to illustrate how COMET works. Given this scenario, our dynamic knowledge sources would be used to determine if loss of memory is occurring, which components are suspect, and which tests would be most useful in identifying the cause. The generation components create multimedia explanations of how to perform the steps in Fig. 2, such as installing the holding battery or loading the frequency. Follow-up questions could be answered in text (e.g., "What is the holding battery?") or in graphics (e.g., "Where is the holding battery?").

---

[1]Currently, this request is received in an internal notation. Up to this point we have focused on the generation of explanations and not on the interpretation of requests. We will soon include an interface to COMET consisting of a combination of menu, pointing, and natural language input devices, based on automated interface design work [4].

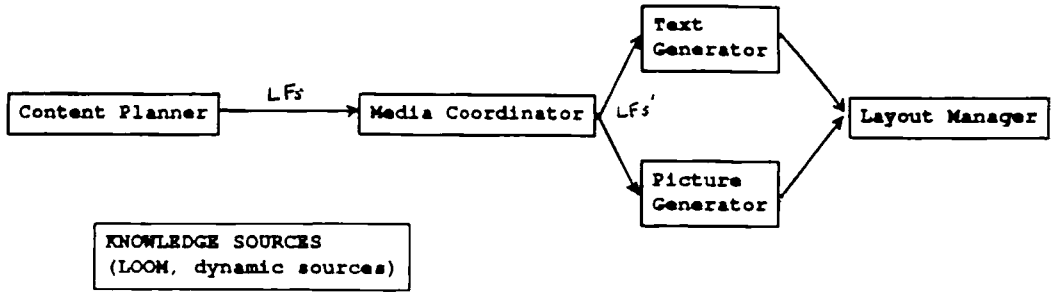


Figure 1: System architecture.

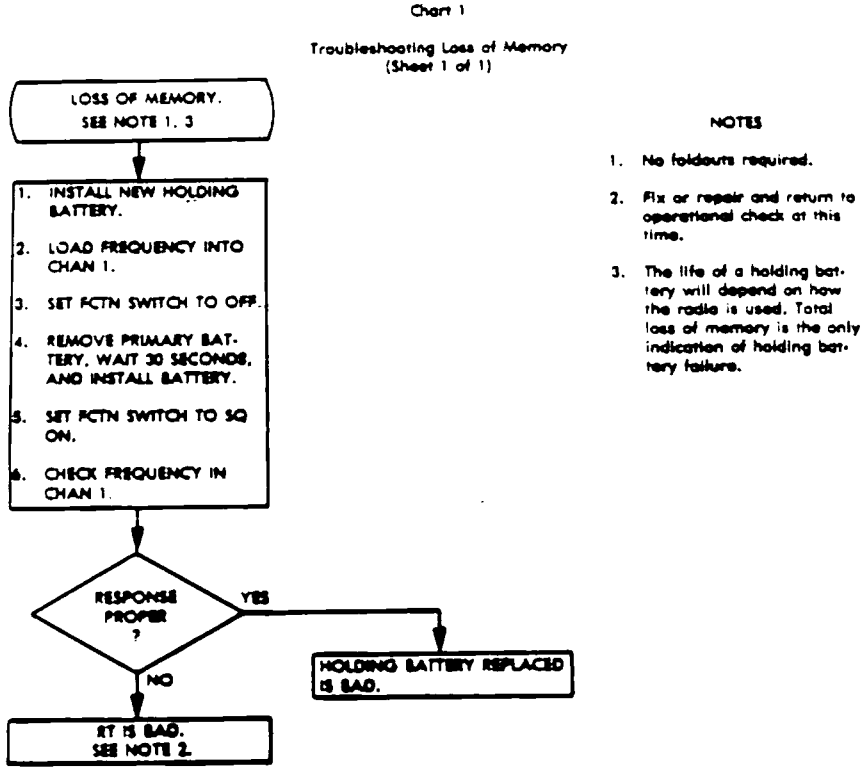


Figure 2: Troubleshooting Loss of memory.

3 Knowledge Sources: Static and Dynamic

We represent static knowledge about the communications radio in LOOM using two hierarchies. The object hierarchy includes information about radio components and their attributes. The action hierarchy contains information about the procedures required for maintenance and repair, and references the object hierarchy. COMET's explanations are currently generated from the static domain knowledge source and geometric knowledge source. For example, to explain how to install the holding battery, COMET would access the LOOM plan for installing the holding battery (shown in Fig. 3) along with concepts for substeps of the plan and parameters to the plan.

```
(defconcept install-hb
            :is (:and process :p
                      (:the substep1 put-rt-on-top)
                      (:the substep2 remove-hb-cover)
                      (:the substep3 remove-hb)
                      (:the substep4 clean-hb-compartment)
                      (:the substep5 set-new-hb-in-place)
                      (:the substep6 replace-hb-cover)))
```

Figure 3: The LOOM plan for installing the holding battery.

COMET contains two dynamic knowledge sources that are currently under development. The first is the *plan execution* [2] component which carries out LOOM plans when needed, probabilistically representing the world as it changes. The plan executor is represented using a Bayesian net, also encoded in LOOM. It is capable of probabilistic reasoning about the state of the radio using dependency rules encoded in the net. It will be used to determine when a particular troubleshooting method (e.g., that shown in Fig. 2) should be used, to determine the part most likely to be bad, and to determine the best diagnostic test to use next. A method for learning probabilities from data on past failures has been developed [3].

As an example of how the plan executor works, consider troubleshooting loss of memory. A portion of the net used for reasoning about the causes of memory loss is shown in Fig. 4. To determine whether the holding battery is responsible, one would load the net with probabilities representing the current state of the radio and values would be propagated to determine the probabilities at the node labeled "Holding Battery good/bad?". Figure 5 shows how the values from an incoming arc are used to compute the probability of attributes at the node in question. This figure is simpler than the normal case since there is only one attribute influencing the probabilistic result at the holding battery node. Also not shown is the novel representation for plan execution in the net developed by Baker [2].

Our second dynamic knowledge source is a standard expert system rule base for diagnosis of equipment failures. Unlike other expert systems, however, our system includes a learning component called GEMINI [8] that can learn missing rules in the system based on past equipment failures. GEMINI integrates similarity- and explanation-based learning techniques to detect and fill in gaps in a rule chain. The system has been fully developed and tested on other domains and is now being ported to the radio domain. Explanation generation using dynamic knowledge sources will begin in fall 1989.

4 Content Planner

The content of an explanation is dynamically planned at the time it is requested. This means that, ultimately, the response content can be influenced by the information in the knowledge base at that time, by the context in which the question was asked, and by information about the requestor. Content is constructed by using schemas [25, 30] that dictate the kind of information to include and its sequencing for given discourse purposes (e.g., explanation, description, or directions). COMET currently uses two schemas, the *process* schema [29] and the *constituency* schema [25], to generate directions for troubleshooting and object descriptions, respectively. Adapting the schemas for a new domain and discourse purpose has led to the development of a hierarchical library of schemas, in which the lower levels contain variants of a schema for a different purpose (or domain) and the higher levels contain generalizations of the schema that hold across domains. This results in smaller schemas that can be more flexibly combined to produce

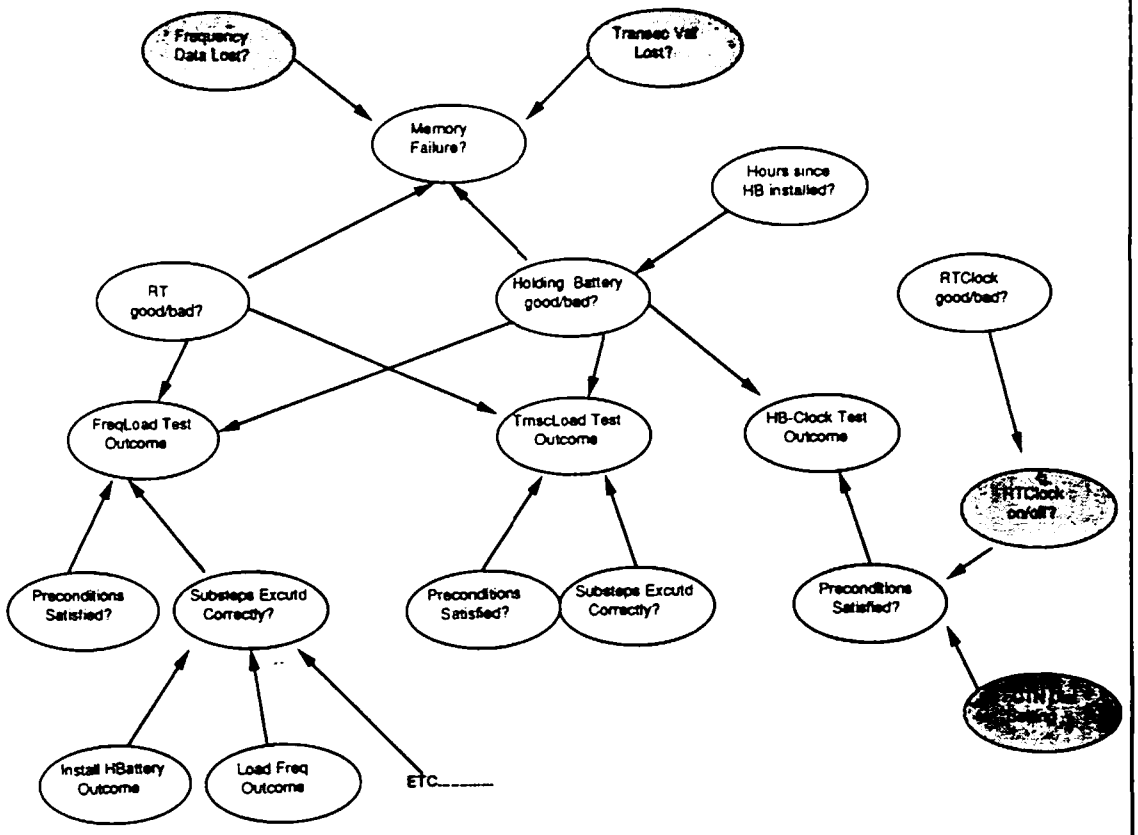


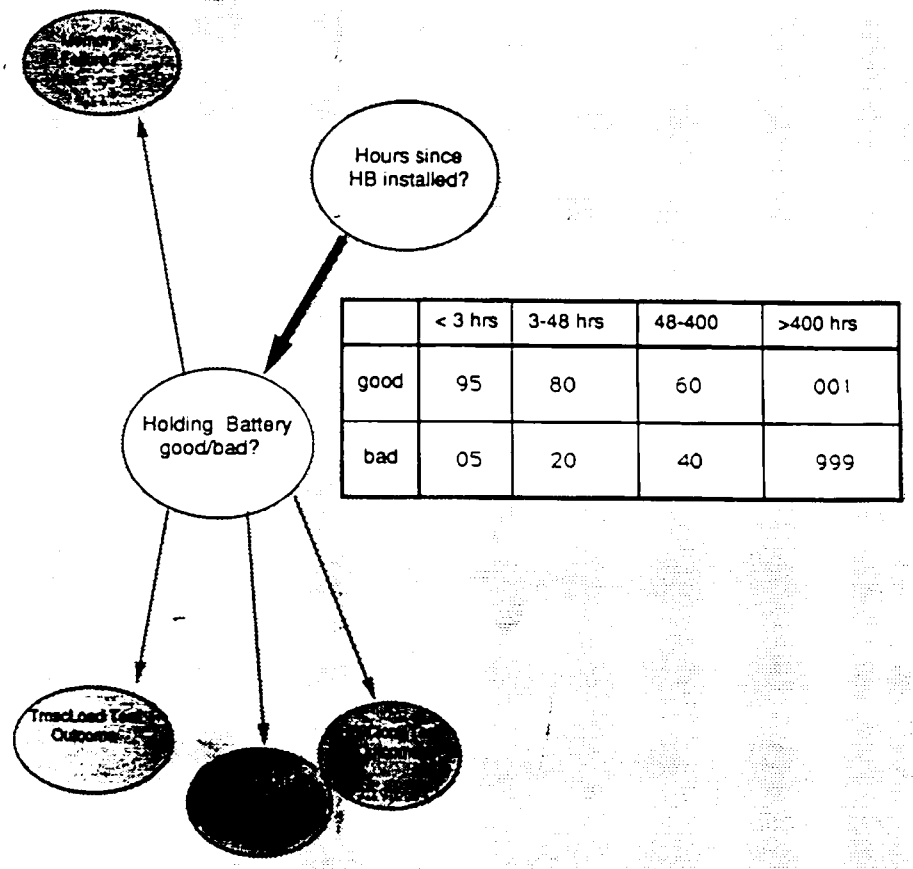
Figure 4: A portion of the net for memory loss.

a greater variety of text.

The content planner produces a list of logical forms (LFs) representing content. These can be realized with a combination of text and graphics. This is done by traversing the schema, which is represented as a graph, and selecting information from the knowledge base for each arc traversed. Tests and semantics on the arcs dictate whether an arc can be traversed. COMET's process schema and the first of six LFs it produces for an explanation of how to install the holding battery are shown in Figs. 6 and 7. We are currently incorporating tests on the user model [20] so that schemas can be combined depending on user background and are developing schemas that will allow us to describe the purpose of an object, action, or test. Generation of a response about purpose is very much dependent upon current context and the task the user is currently performing and thus, it also will require tests on the user model. We are also actively pursuing development of schemas for discussing plans and goals, constraints for selecting and traversing schemas based on user plans and goals, and rules which modify user output based on information about current context and user plans and goals [38, 37, 36]. This is currently implemented in a UNIX consultant domain and is being ported to the radio domain.

5 Media Coordinator

The media coordinator receives as input the list of LFs produced by the content planner and determines which information should be realized in text and which in graphics. Our media coordinator does a fine-grained analysis, unlike other multiple media



	< 3 hrs	3-48 hrs	48-400	>400 hrs
good	95	80	60	001
bad	05	20	40	999

Figure 5: Propagating probabilities.

generators (e.g., [31]), and can decide whether a portion of an LF should be realized in either or both media. We distinguish between six different types of information that can appear in an LF. Based on informal experiments[2] plus relevant literature, we have categorized each type of information as to whether it is more appropriately presented in text or graphics as shown below in Fig. 8. Our experiments involved hand-coding displays of text/graphics explanations for situations taken from the radio repair manual. We used a number of different combinations of media, including text only, graphics only, both text and graphics for everything, and several slight variations on the results shown in Fig. 8. Among the surprising results, we found that subjects preferred that certain information appear in one mode only and not redundantly in both (e.g., location information in graphics only, and conditionals in text only). Furthermore, we found that there was a strong preference for tight coordination between text and graphics. For example, readers strongly preferred sentence breaks that coincided with picture breaks. We would like to further test our hypotheses with actual military users of the repair manual.

The media coordinator is implemented using our functional unification formalism (see Section 6 below). The coordinator has a grammar that does the mapping from

[2]These experiments were conducted by Christine Lombardi as part of her forthcoming masters thesis.

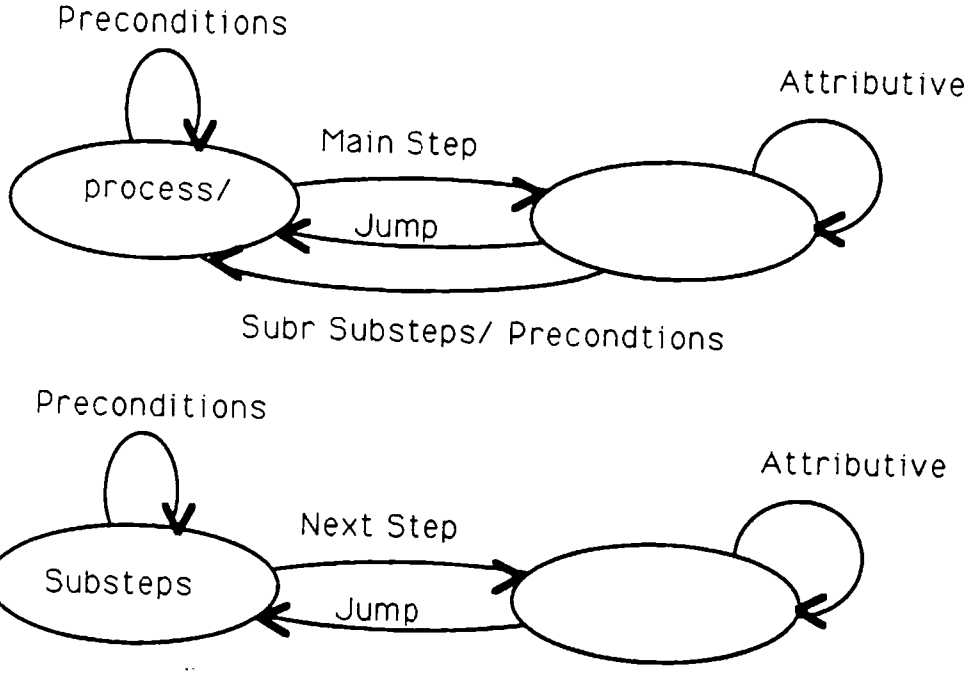


Figure 6: The process schema for directions.

```
(
  ((cat lf)
   (directive-act substeps)
   (substeps
    ((distinct
      ~(((process-type action)
         (process-concept c-put)
         (mood non-finite)
         (tense present)
         (aspect ((perfective no) (progressive no)))
         (speech-act directive)
         (roles
          ((medium
            ((object-concept c-rt)
             (quantification
              ((definite yes)
               (countable yes)
               (ref-obj 1)
               (ref-set 1)))
             (ref-mode description)))
           (on-loc
            ((object-concept c-top)
             (ref-mode name)))))))))))
  )
```

Figure 7: Content planner output (LF 1): Stand the radio on its top-side.

information types to specification of modes. This grammar is unified with the input LFs and results in portions of the LF being tagged with the attribute value pairs `(media-text yes)`, `(media-graphics yes)` (or with values of `no` when the information is not to be presented in a given media). The media coordinator also annotates the LFs with indications of the type of information (e.g., simple action versus

location information	graphics only
physical attributes	graphics only
simple actions	text and graphics
compound actions	text and graphics
conditionals	text for connectives, text and graphics for actions
abstract actions	text only

Figure 8: Division of information.

compound action) as this information is useful to the graphics component in determining the style of the generated pictures. The resulting expanded output for the first LF is shown below in Fig. 9, with the annotations that have been added for the media generators emboldened.

6 Text Generator

The text generator produces English text from the tagged LFs passed to it by the media coordinator. The main problems for the text generator are the selection of words for semantic primitives in the LFs and the construction of syntactic structure and linearization of that structure to produce the sentence. Our focus in this portion of the project has been divided into three major areas: the development and efficient implementation of a functional unification formalism (FUF) and processor for generation, the selection of connectives—one class of word choice, and the identification and representation of collocations that can be used as part of word choice in language generation.

Our motivation for using the functional unification formalism stems from the observation that one main problem in language generation is interaction between constraints. Constraints on selection of words and syntactic structure come from a variety of different knowledge sources (e.g., semantics, syntax, and other lexical selections) and interaction between constraints is bidirectional. In one case, syntactic constraints may force the selection of certain words. For example, if the object of a verb must be an adjective, word choice for the object is limited to adjectives. Conversely, the selection of a word may automatically select syntactic structure. For example, if a verb is selected that cannot be passivized, the sentence must be produced in active form.

While other researchers have claimed that no general statement of constraints is possible due to this complex interaction [7], we have found that FUF does allow for concise statement and control of general constraints [26]. First, because it is a *functional* formalism, it gives equal status to representation of semantic, pragmatic, and syntactic constraints and this means that we can represent constraints from different knowledge sources in a single formalism. Second, it allows for bidirectional interaction between constraints through unification. These advantages, however, have corresponding drawbacks. Since unification is a non-deterministic algorithm, efficiency has been a problem in past systems [25]. We have significantly improved the efficiency of the unification algorithm so that our generator is able to produce a sentence in one or two seconds [10]. Furthermore, we have extended the formalism to allow for the representation of certain complex constraints, such as taxonomic relations and set intersection, through the addition of types [11]. This is also used to increase the efficiency of the unification algorithm.

These extensions allow us to represent constraints on a number of deeper level generation tasks in addition to the construction of surface form. Thus we have been able to use FUF in our system for the media coordination task and for the selection of words. This latter task has been handled in previous generation systems by some form of phrasal

```
((cat lf)
 (directive-act substeps)
 (substeps
   ((distinct
      ((car
         ((process-type action)
           (process-concept c-put)
          (mood non-finite)
          (tense present)
          (aspect
            ((perfective no)
             (progressive no)))
          (speech-act directive)
          (roles
            ((medium
               ((object-concept c-rt)
                (quantification
                  ((definite yes)
                   (countable yes)
                   (ref-obj 1)
                   (ref-set 1)))
                (ref-mode description)
                (cat object-role)
                (object-type none))
               'done')
              (on-loc
                ((object-concept c-top)
                 (ref-mode name)))
              (cat p-role)
              (in-loc
                ((cat object-role)
                 (object-concept none)
                 (object-type none))
                'done'))
             'done')
          (cat lf)
          (substeps none)
          (media-text yes)
          (media-graphics yes))
         'done')
       (cdr none)))
    (cat list)
    (common ((cat lf)))
    (element (substeps distinct car))
    (rest
      ((cat list)
       (common (substeps common))
       (distinct none))
      'done')
    (cset (element rest)))
  'done')
(preconditions none)
(goal none)
(effects none)
(simple-action yes))
```

Figure 9: Tagged LF.

dictionary (e.g., [19, 21]) or discrimination net (e.g., [17, 24]), both of which usually account for only one type of constraint on word choice. We are currently extending FUF further to handle local constraints on content planning and organization so that we can determine the content of the next turn in an interactive session based on constraints from the immediately preceding discourse. Finally, we have also implemented a traditional syntactic grammar that allows us to generate a variety of sentence forms. The result is a cascaded series of FUF "grammars," each handling a separate task but with complete

interaction through unification between the different types of constraints. Thus, for example, decisions made regarding syntax can propagate back to the lexical chooser to influence further choice and the same holds true for interaction between the text generator and the media coordinator.

COMET currently uses the FUF lexical chooser and syntactic generator to produce the final text for the request "How do I install the holding battery?" as shown in Fig. 10. This will later be combined with the corresponding pictures and laid out appropriately on the screen (see Section 8).

```
Stand the radio on its top-side.
Remove the holding battery cover plate:
  Loosen the captive screws and
  pull the plate off of the radio.
Remove the old holding battery.
If the compartment is dirty, then clean it.
Put the new holding battery in the compartment:
  Check the polarity.
Replace the holding battery cover plate:
  Align the holding battery cover plate with the holding battery
  compartment, align the captive screws with the screw holes and
  tighten the screws.
```

Figure 10: Text produced by COMET.

In addition to the development of the formalism for lexical choice, we have also developed a model for connective choice [12]. Each connective (e.g., "but," "although," "because," "since," "and") is defined as a set of constraints between features of the propositions it connects. Using these features, we can account for connective usage where the relation between connected propositions is implicit and for cue uses of connectives, in addition to more standard uses. This model is fully implemented as part of our syntactic FUF grammar and allows COMET to generate complex sentences that often convey information implicitly and thus more concisely.

Finally, we are also investigating the role of collocations in lexical choice. Collocations are word pairs that often appear together, but for which there is no apparent syntactic or semantic basis. For example, "strong tea" and "powerful car" are acceptable noun phrases, while "powerful tea" and "strong car" are not. This holds despite the fact that "powerful" and "strong" are synonymous adjectives. We have developed a system, EXTRACT [34, 33] that can retrieve collocations from large text corpora. We have developed a representation for these collocations in FUF, so that once the generator selects one word in a collocation, the other is automatically selected. EXTRACT has been tested on the Jerusalem Post and Usenet texts and approximately 1000 collocations have been retrieved. We are planning on applying it to the radio domain providing we can locate a large corpus of online maintenance manuals.

Further work for the immediate future will involve including constraints from graphics on choice of words (e.g., if the graphics component has chosen to highlight a dial, text may decide to generate "the highlighted dial" as a reference) and including constraints from the user model on word choice.

7 Graphics Generator

The graphics generator produces pictures from the tagged LFs passed to it by the media coordinator. Our work in COMET has concentrated on full generation of pictures, not the selection or modification of previously generated graphics. In contrast to other researchers [23, 31, 28], we address the generation of 3D, rather than 2D, graphics.

Therefore, the system must choose the viewing specification, lighting specification, object list, and graphical style information that define a picture, guided by the LFs to be depicted. COMET's graphics generator is IBIS (Intent-Based Illustration System) [32]. IBIS uses a rule-based control component, implemented with the CLIPS production system language [6], to build and evaluate a representation of the illustration using a generate-and-test approach.

Each IBIS illustration is created by an *illustrator*. An illustrator must design the illustration so that it fulfills a set of communicative goals that are derived from the information specified for graphics in the LF. The simplest way it can do this is by choosing the viewing specification, lighting specification, objects, and object properties to be used in the illustration. These choices are guided by an *illustration style* that is embodied in the set of rules that IBIS uses.

We refer to the objects that are included in IBIS's pictures as *illustrator objects*. Each is created for a specific illustration. A set of *object relations* indicate the relationships between the illustrator objects and the physical objects in the world being depicted. It is common for an illustrator object to represent a single physical object. There are many cases, however, in which there is not a 1:1 correspondence. For example, several illustrator objects may represent a single physical object at different points in time or as seen from different viewpoints. In contrast, a single illustrator object may correspond to no physical object at all, but may serve as a *metaobject* (e.g., an arrow) that is used to show that an action is being performed.

Each of the physical objects has a set of properties that include *visual properties* that directly affect an object's appearance and *nonvisual properties* that do not. For example, an object's material or size are visual properties, while its importance or cost are not. While an illustrator object may be created with the same visual properties as a physical object that it represents, it may instead be given a visual property that corresponds to a nonvisual property of the physical object. For example, an illustrator object may be emphasized to show its importance by modifying its color from that of the actual physical object. In addition to visual properties, an illustrator object includes additional information about the style in which it will be rendered. One kind of communicative goal that may be associated with an object specifies that it is visible in the illustration. This involves determining whether or not an object is blocked from the viewpoint by other objects. If it is blocked a number of options are available: the viewpoint may be changed; the intervening objects may be deleted from the illustration if unimportant, rendered transparent, or shown using a cutaway view; or the obstructing objects may be left in place if it is determined that they do not block the view sufficiently. This processing will be performed with a solid modeling component currently being incorporated, based on [35, 5].

Unlike our previous work on 3D picture generation [14], IBIS can create composite illustrations, corresponding to compound sentences. A composite picture contains subpictures that may form a series of pictures or a picture with one or more insets. A composite picture may be created when an illustrator determines that a single simple illustration will not realize its communicative goals adequately. For example, a picture may need to show recognizable representations of two objects that differ greatly in size. If both are drawn to size with the larger one as big as possible, the smaller one may be too small to be legible. Creating an inset picture for the smaller one would allow it to be shown at a proper size relative to the larger object in the main picture, and at large enough size to be legible in the inset.

An illustrator creates a composite illustration by spawning one or more child illustrators, each of which inherits from its parent a set of communicative goals to fulfill. As well, the parent determines the child's size and position. If necessary, a child may in turn recursively spawn additional illustrators. As an illustration is built, its illustrator

evaluates it to determine whether it satisfies its communicative goals. Figure 11 shows the illustrator hierarchy for a composite illustration. The root of the tree shows the finished illustration and the communicative goals that it fulfills, while the branches show the illustrators that are spawned, the goals they inherit, and the component pictures they produce.

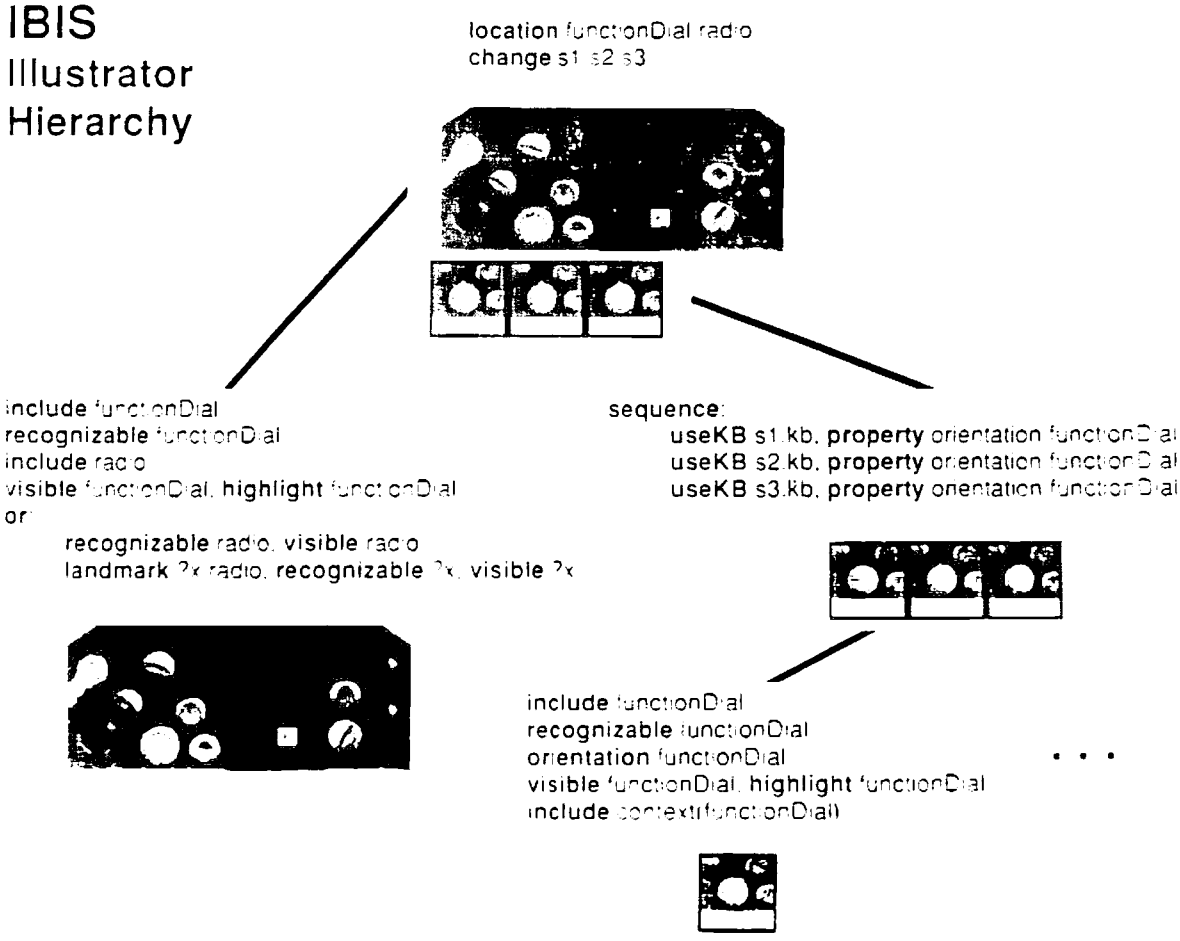


Figure 11: A composite illustration and its illustrator hierarchy

8 Media Layout Manager

COMET's media layout manager, which is currently being designed, receives the generated text and graphics, along with the LFs that they represent. Its task is to determine the precise size and position of each piece of text or graphics. The approach to display layout being used is based on the concept of design grids [27], which are sets of horizontal and vertical lines, associated with rules for their use, that are developed to constrain the positions of graphics and text in publications. The media layout manager will incorporate a rule-based component that creates and uses a design grid based on information about the current user, display hardware, and material to be laid out [15].

Display layout decisions are based in part on the relationships between the objects being laid out. For example, a series of pictures or sentences corresponding to a sequence of steps in a repair procedure have a temporal relationship that may be expressed by presenting them in left-to-right or top-to-bottom spatial order. As well, a

piece of text and a picture that represent all or part of the same LF are related because they express the same or complementary information and may be positioned near one another to express this relationship. These relationships can be determined easily because each media generator correlates the parts of its generated surface structure with the corresponding parts of the input LFs and all media generators use the same LFs.

9 Conclusions and Future Work

COMET dynamically constructs an explanation of how to carry out a maintenance and repair procedure at the time a request is received. In its current state, this means that information available in the knowledge base will influence the selection of explanation content, information will be presented in either text or graphics as appropriate, words and sentence structure will be selected depending on the linguistic context, and picture style will be selected according to information content. We are currently extending the system so that the selection of content and form can be influenced by the user's background and current task. This will involve entering information about potential users into our user model package and identifying and encoding constraints from the user model within each explanation module.

A second current direction is the incorporation of interacting constraints between graphics and text. We plan to modify the text generator so that it can refer to accompanying pictures, the graphics generator so it can incorporate generated textual callouts, and both the text and graphics generators so that sentence structure and picture structure can reflect each other. For example, if three pictures were generated to correspond to the three actions of the last compound sentence in Fig. 10, the text generator may produce three sentences instead. As well, we are interested in developing strategies for those situations in which the media assignments made by the media coordinator are determined to be unsatisfactory by the media generators, which must provide feedback to the media coordinator that can be used to adjust its plan.

As we add to COMET the ability to respond to user requests, it will be necessary to choose appropriate user interface techniques with which to allow users to request information (as part of an interface technique generator) and lay them out on the display in the media layout manager [16]. We will accomplish this by incorporating our SCOPE interface generation system [4].

Acknowledgements

This work is supported in part by the Defense Advanced Research Projects Agency under Contract N00039-84-C-0165, the Hewlett-Packard Company under its AI University Grants Program, the Office of Naval Research under Contract N00014-82-K-0256, the National Science Foundation under Grant IRT-84-51438, and the New York State Center for Advanced Technology under Contract NYSSTF-CAT(88)-5.

The development of COMET is a group effort and has benefited from the contributions of Michelle Baker (plan execution component), Andrea Danyluk (learned rule base), Michael Elhadad (FUF), Laura Gabbe (static knowledge base and content planner), David Fox (text formatting component for media layout), Jong Lim (static knowledge base and content planner), Jacques Robin (lexical chooser), Doree Seligmann (graphics generation), Matt Kamerman (user model), and Christine Lombardi and Yumiko Fukumoto (media coordinator).

References

[1] Allen, J.
Series in Computer Science: Natural Language Understanding.
Benjamin Cummings Publishing Company, Inc., Menlo Park, Ca., 1987.

[2] Baker, M.
Probablistic Analogical Inference in Human and Expert Systems.
Technical Report, Columbia University, February, 1989.

[3] Baker, M. and Roberts, K.
Learning the Probability Distribution for Probabilistic Expert Systems.
Technical Report, Columbia University, December, 1988.

[4] Beshers, C. and Feiner, S.
Scope: Automated Generation of Graphical Interfaces.
In *Proc. ACM Symposium on User Interface Software and Technology.*
Williamsburg, VA, November 13-15, 1989.

[5] Chin, N. and Feiner, S.
Near Real-Time Shadow Generation Using BSP Trees.
Computer Graphics (Proc. SIGGRAPH 89) 23:3, July, 1989.

[6] Culbert, C.
CLIPS Reference Manual
NASA/Johnson Space Center, TX, 1988.

[7] Danlos, L.
The Linguistic Basis of Text Generation.
Cambridge University Press, Cambridge, England, 1987.

[8] Danyluk, A.
Finding New Rules for Incomplete Theories: Explicit biases for induction with contextual information.
In *Proceedings of the Sixth Internaional Workshop on Machine Learning.* Ithaca, N.Y., June, 1989.

[9] Department of the Army.
TM 11-5820-890-20-1 Technical Manual: Unit Maintenance for Radio Sets AN/PRC-119, . . .
Headquarters, Department of the Army, June, 1986.

[10] Elhadad, M.
The FUF Functional Unifier: User's manual.
Technical Report, Columbia University, June, 1988.

[11] Elhadad, M.
Extended Functional Unification ProGrammars.
Technical Report, Columbia University, New York, NY, 1989.

[12] Elhadad, M. and McKeown, K.R.
A Procedure for the Selection of Connectives in Text Generation.
Technical Report, Columbia University, New York, NY, 1989.

[13] Feiner, S., Nagy, S., and van Dam, A.
An Experimental System for Creating and Presenting Interactive Graphical Documents.
ACM Transactions on Graphics 1:1:59-77, January, 1982.

[14] Feiner, S.
APEX: An Experiment in the Automated Creation of Pictorial Explanations.
IEEE Computer Graphics and Applications 5:11:29-38, November, 1985.

[15] Feiner, S.
A Grid-Based Approach to Automating Display Layout.
In *Proc. Graphics Interface '88*, pages 192-197. Edmonton, June, 1988.
(Palo Alto: Morgan Kaufmann, 1988).

[16] Feiner, S.
An Architecture for Knowledge-Based Graphical Interfaces.
In *Proc. ACM SIGCHI Workshop on Architectures for Intelligent Interfaces*, pages 129-140. Monterey, April, 1988.

[17] Goldman, N.M.
Conceptual generation.
Conceptual Information Processing.
North Holland, Amsterdam, 1975.

[18] Hayes-Roth, F., Waterman, D., and Lenat, D. (eds.).
Building Expert Systems.
Addison-Wesley, Reading, MA, 1983.

[19] Jacobs, P. S.
A knowledge-based approach to language production.
PhD thesis, Univ. of California, Berkeley, 1985.

[20] Kamerman, M.
Radio Project User Model.
Technical Report, Columbia University, New York, N.Y., April, 1989.

[21] Kukich, K.
Design of a Knowldege Based Text Generator.
In *Proceedings of the 21st Acl Conference.* ACL, 1983.

[22] Mac Gregor, Robert and David Brill.
LOOM Reference Manual.
Technical Report, USC-ISI, Marina del Rey, CA, 1989.

[23] Mackinlay, J.
Automating the Design of Graphical Presentations of Relational Information.
ACM Transactions on Graphics 5:2:110-141, April, 1986.

[24] McDonald, D.D. and Pustejovsky, J.D.
Description-directed natural language generation.
In *Proceedings of the 9th IJCAI*, pages 799-805. IJCAI, 1986.

[25] McKeown, K.R.
Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text.
Cambridge University Press, Cambridge, England, 1985.

[26] McKeown, K. and M. Elhadad.
Comparison of Surface Language Generators: A Case Study in Choice of Connectives.
Proceedings of the 4th Workshop on Language Generation.
Forthcoming, 1988.
Forthcoming.

[27] Muller-Brockmann, J.
Grid Systems in Graphic Design.
Verlag Arthur Niggli, Niederteufen, Switzerland, 1981.
Trans. by D. Stephenson.

[28] Neal, J. and Shapiro, S.
Intelligent Multi-Media Interface Technology.
In *Proc. ACM SIGCHI Workshop on Architectures for Intelligent Interfaces*, pages 69-91. Monterey, April, 1988.

[29] Paris, C.L.
The Use of Explicit User models in Text Generation: Tailoring to a User's Level of Expertise.
PhD thesis, Columbia University, 1987.

[30] Paris, C. L. and McKeown, K. R.
Discourse Strategies for Describing Complex Physical Objects.
In G. Kempen (editor), *Natural Language Generation: Recent Advances in Artificial Intelligence, Psychology, and Linguistics*. Kluwer Academic Publishers, Boston/Dordrecht, 1987.
Paper presented at the Third International Workshop on Natural Language Generation, August 1986, Nijmegen, The Netherlands.

[31] Roth, S., Mattis, J., and Mesnard, X.
Graphics and Natural Language as Components of Automatic Explanation.
In *Proc. ACM SIGCHI Workshop on Architectures for Intelligent Interfaces*, pages 109-128. Monterey, April, 1988.

[32] Seligmann, D.D., and Feiner, S.
Specifying Composite Illustrations with Communicative Goals.
In *Proc. ACM Symposium on User Interface Software and Technology.* Williamsburg, VA, November 13-15, 1989.

[33] Smadja, F.
Microcoding the Lexicon with Co-Occurrence Knowledge.
In *Proc. of the First International Workshop of Lexical Acquisition.* Detroit, MI, August, 1989.

[34] Smadja, F.
Lexical Co-occurrence: The Missing Link.
Journal of the Association for Literary and Linguistic Computing , To appear, 1989.

[35] Thibault, W. and Naylor, B.
Set Operations on Polyhedra Using Binary Space Partitioning Trees.
Computer Graphics 21:4:153-162, July, 1987.

[36] Wolz, U.
Finding a better way: Choosing and explaining alternative plans.
Technical Report CUCS-409-88, Department of Computer Science, Columbia University, New York, NY, 1988.

[37] Wolz, U.
Tutoring that responds to users' questions and provides enrichment.
Technical Report CUCS-410-88, Department of Computer Science, Columbia University, New York, NY, 1988.
Also appeared in the conference proceedings of the 4th International Conference on Artificial Intelligence and Education, May 1989, Amsterdam, The Netherlands.

[38] Wolz, U., McKeown, K. R., Kaiser, G.
Automated Tutoring in Interactive Environments: A Task Centered Approach.
Journal of Machine Mediated Learning , accepted with minor revisions, Jan. 1989.