

# Simple Natural Language Generation and Intelligent Tutoring Systems

**Barbara Di Eugenio, Michael Glass, Michael J. Trolio**

Electrical Engineering and Computer Science Department  
University of Illinois at Chicago  
1120 SEO (M/C 154), 851 S. Morgan St  
Chicago, IL, 60607 USA  
{bdieugen,mglass,mtrolio}@eecs.uic.edu

**Susan Haller**

Computer Science Department  
University of Wisconsin Parkside  
Kenosha, WI 53141  
haller@cs.uwp.edu

## Abstract

In this paper, we report on our approach to adding Natural Language Generation (NLG) capabilities to ITSs. Our choice has been to apply simple NLG techniques to improve the feedback provided by an existing ITS, specifically, one built within the DIAG framework (Towne 1997a). We evaluated the original version of the system and the enhanced one with a between subjects experiment. On the whole, the enhanced system is better than the original one, other than in helping subjects remember the actions they took. Current work includes exploiting more sophisticated NLG techniques but still without delving into full fledged text planning. We are also conducting a constrained data collection, in which students and tutors interact via the ITS. The system presents the human tutor with the facts the ITS deems relevant to answer the student's question, and the tutor uses them as appropriate in the answer.

## Introduction

Today, many projects aim at providing ITSs with a full-fledged dialogue interface, e.g. see the work at the CIRCLE<sup>1</sup> center (<http://www.pitt.edu/~circle/>), or (Hume *et al.* 1996; Moore, Lemaire, & Rosenbloom 1996; Rosé, Di Eugenio, & Moore 1999; Freedman 1999; Graesser *et al.* 2000). On the contrary, our approach to adding NLG capabilities to an Intelligent Tutoring System falls on the weak side of the divide: we are concentrating on simple sentence planning with no or minimal amounts of text planning. Our choice is partly a development strategy, because we set out to rapidly improve the language feedback provided by an existing ITS shell, partly a desire to evaluate how effective the system can be with a relatively small effort. The results so far are that simple NLG can help, but the gains are

---

<sup>1</sup>Center for Interdisciplinary Research on Constructive Learning Environments.

small enough to suggest that moving to somewhat more sophisticated techniques should be beneficial, even if we still don't intend to develop a full fledged NLG interface.

We take this approach for two reasons. First, we want to understand what can be accomplished by interfacing an NL generator to an ITS taken as a blackbox: can the ITS tutoring strategy be left as is, or is there a point in which the dialogue strategies and the original tutoring strategy are at odds with each other? Second, we are interested in finding out what is the "added value" of an NL interface to an ITS. One way to do so is to compare a system that does not use NL techniques to a version of the same system that uses NL. We are aware of only one other experiment in this direction, described in (Trafton *et al.* 1997). However, the system they describe (navigation in a geographical map) does not seem to qualify as a real ITS.

A similar approach — using simple generation techniques for surface realization in tutoring dialogues — is taken in YAG (McRoy, Channarukul, & Ali 2000). According to McRoy *et al.*, using templates, as opposed to e.g. unification with a grammar, provides a fast mechanism for surface generation. However, we believe their system has not been evaluated yet. In general, the evaluation of NL interfaces to ITSs is an area that needs investigation. ITSs are often evaluated in terms of pre/post-test score, however other measures such as some of the metrics we will discuss later (such as task performance measures) may be appropriate as well. To our knowledge, the only ITSs with an NL interface which has been formally evaluated is CIRCSIM (Evens *et al.* 1993; Kim, Glass, & Evens 2000), but the results of the evaluation are not available yet.

We will first discuss DIAG, the ITS authoring shell we are using. We will then discuss the work we have completed; this comprises the aggregation rules we implemented within EXEMPLARS and the formal evaluation we conducted. We will then discuss some current

work on generating more coherent feedback by exploiting more sophisticated NLG techniques, and the data collection we have started, to study how tutors verbalize the information that the ITS wants to communicate.

## DIAG

DIAG (Towne 1997a; 1997b) is a shell to build ITSs that teach students to troubleshoot complex artifacts and systems, such as home heating and circuitry. DIAG in turn builds on the VIVIDS authoring environment (Munro 1994). VIVIDS based tutors deliver instruction and practice in the context of graphical simulations. Authors build interactive graphical models of complex systems, and build lessons based on these graphical models.

A typical session with a DIAG application presents the student with a series of troubleshooting problems of increasing difficulty. DIAG's tutoring strategy steers the student towards performing the tests that have the greatest potential for reducing uncertainty (Towne 1997a). Most of the times, a test consists of the visual observation of an *indicator*. DIAG keeps track of the tests the student performs, and the inferences that could be made from the symptoms shown. The student interacts with the application by testing indicators and trying to infer which faulty part (RU) may cause the detected abnormal states. RU stands for *replaceable unit*, because the only course of action open to the student to fix the problem is to replace faulty components in the graphical simulation. Figure 1 shows one of the graphical views in a DIAG application that teaches how to troubleshoot a home heating system. The subsystem being displayed is the furnace system. Some of its components are indicators (e.g., the gauges labeled Burner Motor RPM and Water Temperature). Others are either replaceable units, or other complex modules that contain indicators and replaceable units, e.g. the Oil Burner. Complex components are in turn zoomable, i.e., if the user clicks on them a new view that reveals their inner components is shown.

At any point, the student can consult the built-in tutor in one of several ways. For example, if the student suspects an RU to be faulty, s/he can ask the tutor to specify the likelihood that this part is the cause of the fault. The tutor will also indicate the state of any indicators that the student has explored and try to imply a correlation, positive or negative, between the states of the indicators to the RU in question. By utilizing the tutor's feedback, the student can deduce relationships among the system parts and continually refine his/her solution.

### Language Generation in DIAG

After deciding which content to communicate, the original DIAG system (*DIAG-orig*) uses very simple templates to assemble the text to present to the student.

The result is that the feedback that DIAG provides is repetitive, both as a sequence of replies to requests for feedback, and within each verbal feedback. In many cases, the feedback presents a single long list of many parts. This problem is compounded by the fact that most DIAG applications involve complex systems with many parts. Although there are different levels of description in the system model, and hierarchies of objects, the verbal feedback is almost always in terms of individual indicators or units. The top part of Figure 2 shows the reply originally provided by DIAG to a request of information regarding the indicator named "Visual Combustion Check".

We set out to improve on DIAG's feedback mechanism by applying aggregation rules. For example, a long list of parts can be broken down by classifying each of these parts in to one of several smaller lists and then presenting the student with this set of lists.

The bottom part of Figure 2 shows our aggregation rules at work. The revised output groups the parts under discussion by the system modules that contain them (Oil Burner and Furnace System), and by the likelihood that a certain RU causes the observed symptoms. Notice how the *Ignitor Assembly* is singled out in the revised answer. Among all mentioned units, it is the only one that cannot cause the symptom. This fact is just lost in the original answer.

As our sentence planner, we chose EXEMPLARS (White & Caldwell 1998) over better known systems such as FUF (Elhadad 1993) and Penman (Bateman 1994) because of the complexity and learning curve of the latter two. Efficiency and rapid prototyping are among the reasons we chose EXEMPLARS.

EXEMPLARS is an object-oriented, rule based generator. The rules (called *exemplars*) are similar to schema-like text planning rules because they are meant to capture an exemplary way of achieving a communicative goal in a given communicative context, as determined by the system designer. EXEMPLARS is a hybrid system that mixes template-style and more sophisticated types of text planning. The text planner selects rules by traversing the exemplar specialization hierarchy. The applicability conditions associated with each exemplar are successively evaluated in order to find the most specific exemplar for the current context.

In the enhanced version of the system (*DIAG-NLP*), DIAG passes the information to be communicated to EXEMPLARS (the two systems communicate via a text file). Based on the message sent by DIAG, EXEMPLARS performs essentially three tasks:

1. it determines the specific exemplars needed;
2. it adds the chosen exemplars to the sentence planner as a goal;
3. it linearizes and lexicalizes the feedback in its final form, writing it to an external file which is passed

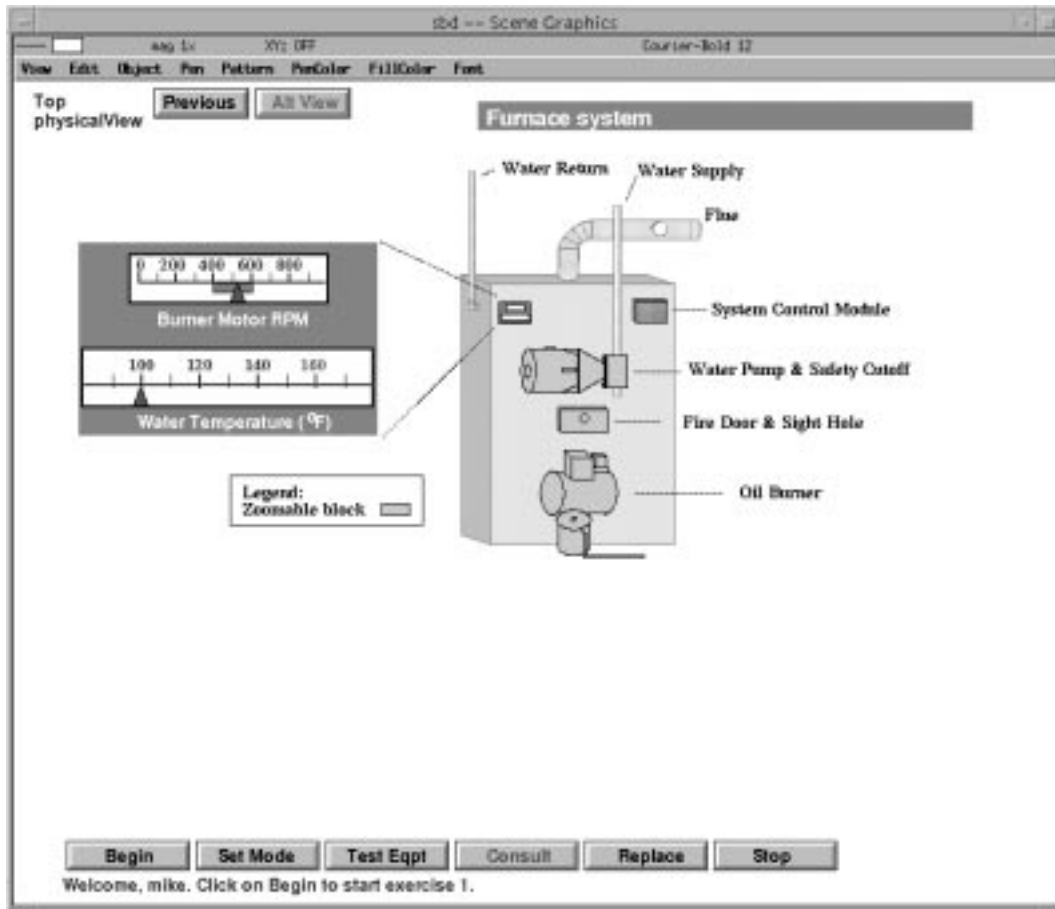


Figure 1: A screen from a DIAG application on home heating

back to DIAG for display in the appropriate window.

In *DIAG-NLP*, we concentrated on rules for aggregation, some of which also affect format and layout. Our choices were suggested by the need to relate the language feedback to the hierarchical structure of the physical system. We have two main kinds of rules, description rules and aggregation rules.

Description rules are used when the full description of a part is required, such as whether the part is in a normal state, its current reading, and, if abnormal, what the normal state should be (see the first sentence in the bottom part of Figure 2).

The aggregation rules are used to group large lists of parts into smaller lists. They allow composite aggregation, so that nested lists are created. Among our aggregation exemplars are:

- *AggByContainer*: each part within this DIAG application is contained within a larger block, called a system module. The *AggByContainer* rule accepts a list of parts, classifies each part by its containing module, and then creates a set of lists by module;

- *AggByFuser*: it groups replaceable units according to the likelihood of being at fault for a specific symptom;
- *AggByState*: it groups indicators by their normal / abnormal state.

A final exemplar, invoked by the other aggregation rules, deals with formatting, namely, creating vertical lists, spacing, etc.

The most frequent application of the aggregation rules is to group parts according to the system module they belong to, and within each module, to group replaceable units by how likely it is they may cause the observed symptom.

The bottom part of Figure 2 shows our rules at work. The revised output groups the parts under discussion by the system modules that contain them (Oil Burner and Furnace System), and by the likelihood that a certain RU causes the observed symptoms. Note how the *Ignitor Assembly* is singled out in the revised answer. Among all mentioned units, it is the only one that cannot cause the symptom. This fact is just lost in the orig-

## turnace

The Visual combustion check is igniting which is abnormal in this startup mode (normal is combusting).

Oil Nozzle always produces this abnormality when it fails.

Oil Supply Valve always produces this abnormality when it fails.

Oil pump always produces this abnormality when it fails.

Oil Filter always produces this abnormality when it fails.

System Control Module sometimes produces this abnormality when it fails.

Ignitor assembly never produces this abnormality when it fails.

Burner Motor always produces this abnormality when it fails.

and, maybe others affect this test.

OK

## turnace

The Visual combustion check indicator is igniting which is abnormal in startup mode. Normal in this mode is combusting.

### Within the Oil Burner

These replaceable units always produce this abnormal indication when they fail:

- Oil Nozzle;
- Oil Supply Valve;
- Oil pump;
- Oil Filter;
- Burner Motor.

The Ignitor assembly replaceable unit never produces this abnormal indication when it fails.

### Within the Furnace System

The System Control Module replaceable unit sometimes produces this abnormal indication when it fails.

Also, other parts may effect this indicator.

OK

Figure 2: Original (top) and revised (bottom) answers provided by DIAG to the same *Consult Indicator* query

inal answer.

In this version of *DIAG-NLP*, morphology, lexical realization and referring expression generation were all treated ad hoc, i.e., they were directly encoded in the appropriate exemplars.

## Experiments

Intuitively, the contrast between the feedback produced by *DIAG-orig* and by *DIAG-NLP* (top and bottom in Figure 2) suggests that even simple aggregation rules dramatically improve the language feedback. To provide a real assessment of this claim, we conducted an empirical evaluation designed as a between-subject study. Both groups interact with the same *DIAG* application that teaches them to troubleshoot a home-heating system. One group interacts with *DIAG-orig* and the other with *DIAG-NLP*.

Seventeen subjects were tested in each group. Our subject pool comprised 13 undergraduates, 18 graduates, and 3 research staff, all affiliated with our university. Participation in the experiment was restricted to science or engineering majors. Each subject first reads some short material about home heating that we developed. Afterwards, each subject goes through the first problem as a trial run. Each subject then continues through the curriculum on his/her own. The curriculum consists of three problems of increasing difficulty. Subjects are encouraged to interact with *DIAG* as much as possible. At the end of the experiment, each subject is administered a questionnaire.

**Metrics.** A detailed log is collected while the subject solves problems, and includes how many problems the subject solved, and, for each problem: total time, and time spent reading feedback; how many and which indicators and RUs the subject consults *DIAG* about; how many, and which RUs the subject replaced.

**Questionnaire.** The questionnaire is divided into three parts. The first part tests the subject’s understanding of the domain. Because the questions asked are fairly open ended, this part was scored as if grading an essay.

The second part concerns the subjects’ recollection of their actions, specifically, of the indicators they consulted the system on and of the RUs they replaced. By taking the log of the subject’s actions as the target, we can compute the usual measures of precision and recall applied to the answers that the subject gives with respect to the target. For each subject, we compute precision as the percentage of correct answers out of the total number of answers the subject gave; whereas recall is the percentage of correct answers they gave with respect to the log of their actions. We also compute the

	DIAG-orig	DIAG-NLP
Time	29.8'	28.0'
Feedback Time	6.9'	5.4'
Consultations	30.4	24.2
Indicator consultations	11.4	5.9
RU consultations	19.2	18.1
Parts replaced	3.85	3.33
Essay score	81/100	83/100

Table 1: Performance measures

	DIAG-orig	DIAG-NLP
Indicator Precision	.33	.17
Indicator Recall	.33	.27
Indicator F-measure	.44	.29
RU Precision	.74	.65
RU Recall	.73	.63
RU F-measure	.72	.63

Table 2: Precision / recall

F-measure,  $\frac{(\beta^2+1)PR}{\beta^2P+R}$ , that smooths precision and recall off, with  $\beta = 1$ .

The third part of the questionnaire asks the subject to rate the system’s feedback along four dimensions on a scale from 1 to 5 (see Table 3).

**Results.** Every student solved all the problems, but differences emerge with respect to other measures. Tables 1, 2, 3 show the results for the cumulative measures across the three problems (measures on individual problems show the same trends).

On the whole, Tables 1 and 3 show a cumulative effect in favor of *DIAG-NLP*, whereas Table 2 does not. Focusing first on Tables 1 and 3, differences on individual measures are not statistically significant; the measure that individually comes closest to statistical significance is *indicator consultations*, which exhibits a non-significant trend in the predicted direction (Mann-Whitney test,  $U=98$ ,  $p=0.11$ ). However, we can compute the cumulative effect that shows *DIAG-NLP* performs better than *DIAG-orig* with the following approach.

We consider only the independent measures (for example, the total number of consultations in Table 1 is

	DIAG-orig	DIAG-NLP
Usefulness	4.35	4.47
Helped stay on right track	4.35	4.35
Not misleading	4.00	4.12
Conciseness	3.47	3.76
Average score	4.04	4.18

Table 3: Subjective rating of *DIAG*’s feedback

	DIAG-orig	DIAG-NLP
Total Time		✓
Indicator consultations		✓
RU consultations		✓
Parts replaced		✓
Essay score		✓
Usefulness		✓
Helped stay on right track	✓	
Not misleading		✓
Conciseness		✓

Table 4: Successes for *DIAG-orig* and *DIAG-NLP*

clearly not independent from indicator and RU’s consultations, given it is the sum of these two measures). For each measure, we decide whether its value indicates a “success” for *DIAG-NLP*. We are not looking at the magnitude of the difference between the two values of the measure, but simply at the fact that the values differ. For example, we consider the lower number of indicator consultations in *DIAG-NLP* as evidence in favor of the effectiveness of the aggregated feedback: because the feedback is better and highlights what is important (such as that the Ignitor Assembly can never cause the Visual Combustion check to ignite, see Figure 2), the subjects can focus their troubleshooting without asking as many questions of the system. Every measure in Table 1 is in *DIAG-NLP* favor, and so is every measure apart from *helped stay on right track* in Table 3 (we consider a tie as a success for *DIAG-orig*). We then ask the question: what is the probability that the  $m$  successes for *DIAG-NLP* out of the  $n$  independent measures are simply due to chance? We can answer this question via  $B(m - 1, n, 0.5)$ , the binomial cumulative distribution function through  $m - 1$  for sample size  $n$  and probability of success  $p=0.5$ : it gives us the probability that of  $n$  random trials, the number of successes will fall between 0 and  $m - 1$ , inclusive. Thus,  $1 - B(m - 1, n, 0.5)$  gives us the probability that  $m$  or more successes out of  $n$  are due to chance.

As an example, consider Table 4, in which we combine the independent measures from Tables 1 and 3 and note whether they represent a success for *DIAG-orig* or *DIAG-NLP*. The probability of 8 successes out of 9 measures is  $p = 0.020$  ( $1 - B(7, 9, 0.5)$ ). If we leave *Total Time* out because it may not be an independent measure,<sup>2</sup> the probability of 7 successes out of 8 is  $p = 0.035$  ( $1 - B(6, 8, 0.5)$ ). Finally, if instead of using the four subjective measures we use their average (which constitutes a success for *DIAG-NLP*), we obtain  $p = 0.016$ , and if we eliminate time in this last case, we obtain  $p = 0.031$ . To conclude, in whatever way we combine these measures, we obtain evidence that the

<sup>2</sup>One could argue the time went down because of the smaller number of consultations.

better scores *DIAG-NLP* obtains, albeit individually not statistically significant, cumulatively show that *DIAG-NLP* outperforms *DIAG-orig*.

However, we have not discussed Table 2 yet. This table shows that subjects in *DIAG-orig* remember what they did better than those in *DIAG-NLP*. The results are very poor for both groups, especially for indicator consultations; precision and recall are much higher for RU replacement, as would be expected because there are many more indicator consultations than RU replacements. The measures concerning indicators achieve or show trends towards statistical significance: indicator precision and indicator F-measure are significant (t-test, respectively 2.19,  $p = 0.04$  and 2.51,  $p = 0.02$ ), and indicator recall is marginally significant (Mann-Whitney,  $U = 93.5$ ,  $p = 0.08$ ). All in all, this is a puzzling result, especially because subjects in *DIAG-orig* consult the system on indicators almost twice as many times as the subjects in *DIAG-NLP*, thus we would expect them to have more problems remembering what they did. Perhaps this result can be related to some results from (Kintsch 1998), that show that high-quality text does not necessarily lead to better performance. Kintsch describes experiments (p. 314-318) in which high-knowledge subjects performed better on problem solving questions when presented with a low-coherence, rather than with a high-coherence, text; however, their recall of the text was better for the high-coherence texts. Kintsch’s explanation is, informally, that the high-coherence texts didn’t prod those subjects to “work enough” on the text to integrate it properly with their previous knowledge to construct an appropriate model, whereas the low-coherence text did. The situation here is somewhat different, in that in fact our subjects’ performance was better with the better texts, and it is their recall of their actions (not of the text) that is affected.

Finally, the reader may wonder what happens to the cumulative effect that shows *DIAG-NLP* better than *DIAG-orig* if we take into account the measures in Table 2 as well. By adding to Table 4 two successes for *DIAG-orig*,<sup>3</sup> we compute the probability of obtaining 8 successes out of 11 measures by chance. We obtain  $p = 0.113$ , which shows a non significant trend in the predicted direction. However, recall that we are being conservative: for example, we counted *help stay on right track* in favor of *DIAG-orig* even if it is a tie; if we count it in favor of *DIAG-NLP*,  $p$  goes down to 0.033.

We refer the reader to (Di Eugenio & Trollo 2000) for further statistical analysis of individual measures.

<sup>3</sup>Given their definitions, precision and recall cannot be considered as really independent measures, and certainly the F-measure that combines them is not independent from either of them. So we synthesize Table 2 as two successes for *DIAG-orig*, one for indicators, one for RU’s.

## Current and future work

The results of the study we just discussed make us confident that it is not necessary to add a full fledged NL generator to an existing ITS to obtain reasonable results. This is not to say that the tutoring strategy the ITS uses to start with cannot be improved, however, this shows that better language can be added at a relatively low cost (the implementation took one graduate student six months), and that it can be effective.

As a consequence, we are now pursuing two lines of research. The first is to add some more sophisticated NL techniques without plunging into full text planning, because we want to see how far the weak approach can go. Second, we are conducting a constrained data collection to help us discover some empirical foundations on which to base the realization of the facts the ITS intends to communicate.

We now discuss both efforts more in detail.

### Focusing and rhetorical relations

In the work done so far, we imposed coherence on the tutor turn by means of aggregation rules. However, the turn could be made more coherent by introducing appropriate referential expressions (generated ad hoc so far), and the minimum amount of text planning needed to express a few domain or rhetorical relations among the facts to be expressed. For example, the fact that the ignitor assembly never causes the abnormal indication mentioned in Figure 2 as opposed to the fact that the other parts within the oil burner always cause it, could be given more prominence if the relevant propositions were linked by a *contrast* relation<sup>4</sup> rendered via an appropriate cue phrase, such as *but* ((A) and (B) are used later to refer to the appropriate part of the explanation):

- (1) *The visual combustion check indicator is igniting which is abnormal in startup mode. Normal in this mode is combusting.*
  - (A) *Within the oil burner, the oil nozzle, oil supply valve, oil pump, oil filter and burner motor always produce this abnormal indication when they fail.*
  - (B) *But the ignitor assembly never does.*

We have started some work in this direction, in which we have coupled EXEMPLARS to a knowledge base built via the SNePS representation system (Shapiro & Rapaport 1992). SNePS is a semantic network formalism where each node represents a proposition. In general, it is very difficult to access the knowledge about the physical structure of the system and causal relationships in VIVIDS-based tutors. These types of knowledge are often only indirectly present: they are reflected in how changes to graphical objects affect other objects, but this is not sufficient to generate language.

<sup>4</sup>We are using relations from Rhetorical Structure Theory (Mann & Thompson 1988).

When they are present, they are expressed in a very non-symbolic way. So, in a sense we need to extract some of this knowledge from the existing tutor and represent it in a usable form for the NL generator — this was done in *DIAG-NLP* by representing the required knowledge via Java classes (EXEMPLARS is written in Java).

Using SNePS has the advantage that it makes it very easy to represent and reason about entire propositions, not just about objects. For example, it is straightforward to represent the various individual propositions that underlie Ex. 1 above, and the causal relations between the failure of the individual parts and the abnormal state of the visual combustion check. Moreover, it is also easy to represent the contrast relation between the two complex propositions (A) and (B). Finally, because propositions are full fledged entities in the representation, they can become part of the discourse model, and be referred to with appropriate referential expressions. In this version of the generator, we implemented the GNOME algorithm to generate referential expressions (Kibble & Power 2000), which is a simple algorithm that uses insights from centering (Grosz, Joshi, & Weinstein 1995) and from theories of saliency.

This revised version of the generator renders the same facts underlying Figure 2 as shown in in Figure 3.

In Figure 3, the deictic *This* is generated by the GNOME algorithm and is used to refer to the proposition representing the abnormal state of the visual combustion check indicator; this cuts down on some of the repetitiveness of the feedback generated by *Diag-orig* and also by *DIAG-NLP*, see Figure 2. However, the indefinite articles introduced in Figure 3 are incorrect (the algorithm we implemented does not take into account the visual context, or the fact that there is only one part with that description). The contrastive particle *but* is not included in Figure 2 because we have not yet implemented exemplars to generate cue phrases; however, as soon as we do so, it will be very easy to generate it, as the appropriate rhetorical relation is included in the SNePS representation of the message to be conveyed.

Further, in order to improve our surface realization capabilities, we are planning to use a different version of EXEMPLARS which is integrated with *RealPro* (Lavoie & Rambow 1997), a surface realizer developed at CogenTex like EXEMPLARS.

### First observations of human consulting

The aggregation rules we implemented in EXEMPLARS appear to be plausible, but they have no empirical foundation. To understand how a human tutor may verbalize a collection of facts, we are collecting tutoring dialogues between a student interacting with the same DIAG application we have previously discussed and a human tutor. In this experiment the tutor and the student are in different rooms, sharing images of the same DIAG tutoring screen. When the student exer-

A visual combustion check indicator is igniting in startup mode.  
 The visual combustion check indicator igniting in startup mode is abnormal.  
 Within the furnace system,  
 This is sometimes caused when a system control module replaceable unit is inoperative.  
 Within the oil burner,  
 This is never caused when an ignitor assembly replaceable unit is inoperative.  
 This is sometimes caused when a burner motor, oil filter, oil supply valve, or oil nozzle is inoperative.

Figure 3: Adding a bit more of sophistication to the generator

cises the consult function the tutor sees the information that DIAG would use in generating its advice — exactly the same information that DIAG gives to EXEMPLARS in *DIAG-NLP*. The tutor then types a response that substitutes for DIAG’s response. Although we cannot constrain the tutor to provide feedback that includes all and only the facts that DIAG would have communicated at that specific moment, we can still see the effects of how the tutor uses the information provided by DIAG. As of this writing, we have preliminary observations of two human tutors, consisting of 45 human responses to DIAG consult requests.

The most striking pattern we observe in the human-generated advice is that the humans eschew syntactic aggregation of part lists and instead describe or name functional aggregations of parts. Our tutors write like this:

1. Referring to oil nozzle, supply valve, pump, filter, etc:
  - 1a) “... check the other items on the fuel line” [Tutor 1]
  - 1b) “... follow the path of the oil flow” [Tutor 2]
2. Referring to all the burner parts:
  - 2a) “What are the parts that control the combustion?” [Tutor 1]
  - 2b) “... consider the units that are involved with heating the water” [Tutor 2]
3. Referring to the photocell that senses the presence of flames:
  - 3a) “Check the electronics that indicates that there is combustion” [Tutor 1]
  - 3b) “Consider the sensors that tell the SCM to turn the burner motor off” [Tutor 2]

In these examples, the pairs of utterances show the two tutors independently describing the same assemblages of parts.

It is not clear to us to whether these assemblages of parts are fixed in content, or whether they are constructed according to the evolving discourse context. In general it seems to be a fixed collection. But the tutor sometimes constructs an impromptu subset, as in “the valve is open, so you have to check the point below the

filter” which appears to be an impromptu reference to parts in the fuel line “below” the filter.

The assemblages we see in the human discourse are not necessarily represented in the training documentation or the functional diagrams on the DIAG screen; it appears the tutors are constructing them.

Some of the other phenomena we have observed:

- In contrast to DIAG, neither tutor ever mentions parts that *cannot* be causing the problem (e.g., the ignitor assembly in Figure 2), except when the student consults precisely on those parts.
- One tutor frequently introduces devices for inter-turn coherence. For example, two adjacent turns were introduced by “not a good choice” and “better choice,” respectively. Another turn was introduced by “the question is now,” indicating the reasoning was in some way following from the previous turn.
- The human tutors occasionally justify a statement, frequently by appealing to causal reasoning. For example, one tutor wrote “The oil filter is normally clean. A dirty and clogged oil filter *blocks the flow of oil* and should be replaced” (emphasis added). By contrast, DIAG merely states whether a broken oil filter can cause the problem, without interpolated explanation.

As our experiments with human tutors continue, we should be able to produce a more complete catalog of language and discourse phenomena. Of particular interest, given our emphasis on aggregation, is the parts assemblages the tutors use, especially their content, how they are described, when they are invoked, and how to organize the knowledge the tutor needs in order to imitate the human tutors’ behavior.

**Acknowledgements.** This work is supported by grants N00014-99-1-0930 and N00014-00-1-0640 from the Office of Naval Research, Cognitive, Neural and Biomolecular Science and Technology Division. We are grateful to CoGenTex Inc., in particular to Mike White, for making EXEMPLARS available to us; and to Michael Scott for showing us how to compute the cumulative effect of different success measures.



## References

- Bateman, J. A. 1994. KPML: The KOMET-Penman (Multilingual) Development Environment. Technical report, Institut für Integrierte Publikations- und Informationssysteme (IPSI), GMD, Darmstadt. Release 0.6.
- Di Eugenio, B., and Trolio, M. J. 2000. Can simple sentence planning improve the interaction between learners and an intelligent tutoring system? In *Building Dialogue Systems for Tutorial Applications (AAAI Fall Symposium)*. American Association for Artificial Intelligence.
- Elhadad, M. 1993. FUF: the universal unifier – user manual version 5.2. Technical Report CUCS-038-91, Columbia University.
- Evens, M. W.; Spitzkovsky, J.; Boyle, P.; Michael, J. A.; and Rovick, A. A. 1993. Synthesizing tutorial dialogues. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, 137–140. Hillsdale, New Jersey: Lawrence Erlbaum Associates.
- Freedman, R. K. 1999. Atlas: a plan manager for mixed-initiative, multimodal dialogue. In *AAAI99 Workshop on Mixed-Initiative Intelligence*. Orlando, FL: American Association for Artificial Intelligence.
- Graesser, A. C.; Wiemer-Hastings, K.; Wiemer-Hastings, P.; Kreuz, R.; and the Tutoring Research Group. 2000. Autotutor: A simulation of a human tutor. *Journal of Cognitive Systems Research*. To appear.
- Grosz, B.; Joshi, A.; and Weinstein, S. 1995. Centering: A Framework for Modeling the Local Coherence of Discourse. *Computational Linguistics* 21(2):203–225.
- Hume, G. D.; Michael, J. A.; Rovick, A. A.; and Evens, M. W. 1996. Hinting as a tactic in one-on-one tutoring. *Journal of the Learning Sciences* 5(1):23–47.
- Kibble, R., and Power, R. 2000. Nominal generation in GNOME and ICONOCLAST. Technical report, Information Technology Research Institute, University of Brighton, Brighton, UK.
- Kim, J. H.; Glass, M.; and Evens, M. W. 2000. Learning use of discourse markers in tutorial dialogue for an intelligent tutoring system. In *COGSCI 2000, Proceedings of the 22nd Annual Meeting of the Cognitive Science Society*.
- Kintsch, W. 1998. *Comprehension. A paradigm for cognition*. Cambridge University Press.
- Lavoie, B., and Rambow, O. 1997. A fast and portable realizer for text generation systems. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*.
- Mann, W. C., and Thompson, S. 1988. Rhetorical Structure Theory: toward a Functional Theory of Text Organization. *Text* 8(3):243–281.
- McRoy, S. W.; Channarukul, S.; and Ali, S. 2000. Text realization for dialog. In *Building Dialogue Systems for Tutorial Applications (AAAI Fall Symposium)*. American Association for Artificial Intelligence.
- Moore, J. D.; Lemaire, B.; and Rosenbloom, J. A. 1996. Discourse generation for instructional applications: Identifying and exploiting relevant prior explanations. *Journal of the Learning Sciences* 5(1):49–94.
- Munro, A. 1994. Authoring interactive graphical models. In de Jong, T.; Towne, D. M.; and Spada, H., eds., *The Use of Computer Models for Explication, Analysis and Experiential Learning*. Springer Verlag.
- Rosé, C. P.; Di Eugenio, B.; and Moore, J. D. 1999. A dialogue based tutoring system for basic electricity and electronics. In *AI-ED 99, Proceedings of the 9th International Conference on Artificial Intelligence in Education*.
- Shapiro, S., and Rapaport, W. 1992. The SNePS Family. *Computers and Mathematics with Applications, Special Issue on Semantic Networks in Artificial Intelligence, Part I* 23(2–5).
- Towne, D. M. 1997a. Approximate reasoning techniques for intelligent diagnostic instruction. *International Journal of Artificial Intelligence in Education*.
- Towne, D. M. 1997b. Intelligent diagnostic tutoring using qualitative symptom information. In *AAAI97 Fall Symposium on ITS Authoring Tools*.
- Trafton, J. G.; Wauchope, K.; Raymond, P.; Deubner, B.; Stroup, J.; and Marsch, E. 1997. How natural is natural language for intelligent tutoring systems? In *Proceedings of the Annual Conference of the Cognitive Science Society*.
- White, M., and Caldwell, T. 1998. Exemplars: A practical, extensible framework for dynamic text generation. In *Proceedings of the Ninth International Workshop on Natural Language Generation*, 266–275.