Criticism-Based Knowledge Acquisition for Document Generation

Barry G. Silverman

This chapter describes an AI application called the TRADOC issue management expert (TIME), which was deployed in 1990 for the U.S. Army Training and Doctrine Command (TRADOC). TIME helps generate a document that is part of the system acquisition milestone or decision process. The goal of TIME is to bridge the gap between the headquarters decision makers who know how to write the desired document and the various authors at 17 sites nationwide who know what to put in the document. TIME has several capabilities. In one capacity, it is an expert critiquing and tutoring system that emulates the headquarters decision makers. In this capacity, it communicates good document-authoring rules, heuristics, and practices in real-time (during authoring) mode. In another capacity, TIME is also a knowledge-acquisition system that interviews the author and helps him(her) generate the document. Finally, TIME acts as an intelligent memory that dynamically indexes and collects as many as 600 documents that are produced annually in a corporate memory for later analogical retrieval and illustration purposes. This last capability means that the more TIME is used, the more useful it is.

Significance of the Application to the U.S. Army

The domain for TIME consists of two principal tasks: forecasting and decision problem structuring. The forecasting task is to predict the worst mission a new piece of materiel (for example, tank, helicopter, or gas mask) might have to be taken on. Decision makers can then use this forecast to structure the issues and criteria that must be evaluated to decide whether to purchase or develop this piece of materiel. Accurate forecasts and robust decision issues and criteria are important to the decision to procure the new item and, thus, to the ultimate safety of those who will have to take it into the battlefield. However, this accuracy and robustness are often obscure to the mind of the Army author because of the short schedule for completing the report and all the conflicting demands on the author's time.

Lack of concrete, first-hand information further compounds the prediction-specification tasks. Increasingly, Army experts have less direct experience with current threats (threats are what make a mission difficult to complete) or with what leads to robust performance on the battlefield. Army experts gain much of their insight from interviewing sources and closely studying intelligence reports about the various disciplines (for example, artillery, electronic warfare, or biochemical agents). These experts have little direct experience. That is, except in areas for which they have direct concrete experience, much of the information available for making these forecasts and specifications is abstract data.

In other words, there is a short timetable for completing the overall task as well as delays associated with researching abstract battlefield information. These two features combine in the mind of the expert to create a situation where weak and biased heuristics might replace a more normative form of reasoning. This use of biased heuristics occurs with no loss of confidence on the part of the Army knowledge base preparer.

The system acquisition process in the Army and in the military at large requires hundreds of documents to be generated during the life cycle of each new piece of materiel. The same types of problems occur with other kinds of military documents as occur in the one type of document that TIME helps with. Any success achieved in the TIME project is potentially reusable for solving similar problems in other document preparation efforts.

Significance of the Application to the Field of AI

Helping this domain requires a novel approach. Decision-aiding techniques that might improve the authors' reasoning historically rely on theory-rich but domain knowledge–impoverished approaches. Some examples are decision analysis and multiattribute utility theory. By them-

selves, such techniques can't solve the problem because domain knowledge is one of the critical missing ingredients leading to bias and error.

Expert systems also are not entirely appropriate. The Army problem requires an interactive decision aid that drives the user to a more theoretically rigorous and normative decision style. However, expert systems are theory poor. Further, expert systems are for replace-the-expert purposes. The domain in this case, however, is too vast (that is, the entire Army is the domain) to replace the human. For the same reason, an intelligent tutoring system also cannot work in this domain.

Although they often are found in support-the-expert roles, expert systems increasingly appear to have poor human factors when applied to decision-aiding settings (for example, see Langlotz and Shortliffe [1983]; Klein and Calderwood [1986]; or Roth, Bennett, and Woods [1988]). More of a "joint cognitive system" orientation is warranted than expert system technology can supply alone. Again, the same human factors concerns apply to intelligent tutoring systems.

Another consideration is that TIME's purpose is to help generate reports or documents in the form of knowledge bases. This tool has to adhere to theoretically correct prescriptions for decision aiding and simultaneously provide knowledge-rich support to its users. These needs require a knowledge-acquisition tool that also uses a form of knowledge-based system technology to critique the validity of the users' input.

Expert critiquing systems that help an expert improve his(her) task performance are not new (for example, see Miller [1983] and Langlotz and Shortliffe [1983]). What is new is (1) the use of deep knowledge of human judgment as a theory of bugs and repairs to guide the design of the critics and (2) the merger of expert critiquing with knowledge-acquisition–system technology. The result is criticism-based knowledge acquisition. This technique offers a judgment theory–motivated, knowledge-rich approach to expert decision supporting, expert problem solving, and other types of human expert–conducted tasks.

This section gives a brief introduction to the theory of criticism (that is, bugs and repairs) of expert intuition. The interested reader can see Silverman (1990, 1991, 1992) for more detail. Readers can also directly consult the bug theories in the literature, such as Kahneman, Slovic, and Tversky (1982). In this literature, it is popular to debias human judgment with the aid of a linear model. Expert critics replace the linear model with a decision network of heuristic critics.

Specifically, a machine critic's heuristics include a decision network of alternative criticism strategies that are triggered when earlier strategies fail to remove the error. As suggested by the rules across the top of table 1, it is useful to have influencers warn experts about biases and explain how to avoid them (prevention is quicker than a cure). Biases,

IF: Error Category Is	THEN: Potential Critic Strategies Are		
	Influencers (Positive, before and during task critics)	Debiasers (Negative, after task critics)	Directors (Task definition adjutants)
OBTAINING INFORMATION (from Memory, Environment, or Feedback) o Availability o Base Rate o Data Preservation o Ease of Recall o Hindsight Bias o Memory Capacity Constraint o Recency of Occurrence o Selective Perception	o Hint & Cue to Stimulate Recall o Show Good Defaults, Analogs , and Other Cut-and-Paste Items to Replace What's Missing From Memory o Explain Principles, Examples, and Referencess (tutoring) to Impart New Insight o Use Repitition	o Test, Trap, and Doubt to See If Memory has Retained the Information Offered By The Influencers o Recognize Memory Failure Modes, Explain Cause and Effect, and Suggest Repair Actions	o Cause the User to Notice More Cues in the Environment and in Memory so he Can Correct the Error Himself o Tell the User How to Follow a Proper Path so He Can Reach a Normative Anchor
HANDLING UNCERTAIN INFORMATION o Adjustment o Confirmation o Conservatism o Gambler's Fallacy o Habit o Illusion of Control o Law of Small Numbers o Overconfidence o Regression Effect o Representativeness o Selective Perception o Spurious Cues o Success/Failure Attribution o Wishful Thinking	o Hint & Cue About Laws of Probability o Show Good Defaults, Analogs , and Other Drag-and-Paste Items to Help User Improve o Explain Principles, Examples, and Referencess (tutoring) to Impart New Insight o Tutor with Differential Descriptions, Probability Models, etc. o Use Visualization Aids	o Test, Trap, and Doubt to See If Info Processing Is Succombing to the Biases o Recognize Processing Failure Modes, Explain Cause and Effect, and Suggest Repair Actions	o Suggest How the User Can Better Structure the Problem so He Can See the Error Himself o Tell the User How to Follow a Proper Reasoning Path so He Can Reach a More Optimal Outcome
OUTPUT ERRORS o Errors in Difference Between Intended and Actual Output	o Display Visual Depictions of the Output to Give User the 'Big Picture' o Suggest Standard and Useful Responses the User Might Overlook	o Notice Defective Responses o Explain Causes and Adverse Effects o Suggest Repairs	o Provide Proper Format Guidance o Walk the User Through Response Specifications
Knowledge Errors	o Alert User to News and Updates About Knowledge, Constraints, Viewpoints o Disseminate Information not in the User's Purview	o Evaluate the User's Solution From Different Views o Suggest Incremental Improvements	o Tell the User How to Follow Known Normative Procedure and Use Prescriptive Knowledge

Table 1. Types of Strategies Relevant for Critiquing Various Errors.

like perceptual illusions, are often hard to remove even after they have been pointed out. Thus, it is necessary to have debiasers notice that an erroneous judgment remains and steer the experts toward a more correct reasoning strategy. Finally, a formal reasoning director is needed if experts still retain their biased answer. The director requires the user to gather (abstract) data, conduct the steps of a rigorous analysis, and compute the correct answer.

The precise strategy to use for each influencer, debiaser, or director depends on several variables. As the left side of table 1 shows, it de-

pends on the cognitive process and the type of error involved. The critic designer looks down this list to the relevant rows and selects the strategies from the body of the table. The knowledge-rich side of a decision network of critics is domain specific. Individual critics must be instantiated for each new domain tackled. Those built for the Army problem are explained in detail in Case Study: Issue and Criteria Knowledge Base Acquisition and in Description of the Forecasting Task.

In addition to table 1, this application also adheres to the implementation rules of table 2, guidelines for deployment. Table 2 shows the important where, when, and how considerations of critic design. When the conditions on the right side of table 2 exist, the designer implements the critic so that it has the features on the left.

Finally, the TIME critic system is a working application that successfully passed on-site field test in late 1990 . Alpha testing was conducted at 6 sites from January through April 1991. TIME is believed to be one of the largest critic systems ever built. To date, it has over 1,500 rules, 1,500 objects, 2,000 note cards, 300 analogs, and numerous other data structures. This write-up offers only a small look at TIME.

Case Study: Issue and Criteria Knowledge Base Acquisition

The simplest way to introduce the Army case study is to describe a scene that captures the essentials and to which details can and will be added as needed. In illustrating the use of tables 1 and 2, the issue and criteria structuring task appears before the forecasting task even though the reverse takes place in the real TIME system.

Scene

A potential contractor, John, has a new satellite communications component called SATCOM that he hopes organization XYZ will sponsor for further development and, ultimately, for production. XYZ's new communications systems evaluator, Sally, got her graduate degree several years ago in telecommunications engineering and has stayed abreast by regularly reading technical journals and reports and attending short courses on the subject. Sally knows the telecommunications subject fairly well and is impressed with the technical merit of the ideas that John is proposing to incorporate. To stay up to date in the telecommunications technology development field, however, Sally has had little chance to work with the application or operation side of the XYZ organization. She realizes personnel in operations don't have advanced degrees or analytic backgrounds, which is part of the reason that Sally chose technology development as a career, but she doesn't

		THEN These Choices Are Appropriate	IF: These Conditions Exist
WHEN TO DEPLOY DIFFERENT CRITIC TYPES	Types of Applica-tions	Structured	Well-structured task, and intermediate skilled users.
		Semi-Structured	Semi-structured task with existing electronic environment, situated learning is needed, and user errors are likely.
	Timing	Before Task Critic	Commonly recurring errors can be anticipated/prevented and users won't be info overloaded by the preventive measures.
		During Task Critic	Error can be detected before task is finished, users can be safely interrupted, interruption will be beneficial.
		After Task Critic	An error needs correcting and users will correct after the fact.
	Process	Incremental Critics	Suggestions are best given close to time of error.
		Batch Critics	Suggestions can be bunched in a group at specific intervals.
	Mode	Passive Critic	User is best determiner of when to use the critic and the output of the critic is not always desired.
		Active Critic	User is novice/intermediate, suggestions will improve task performance, and user would welcome the interruption(s).
	Knowledge	Shallow Critic	Superficial task and user goal understanding is permissible.
		Deep Critic	Task understanding is required to be precise or user goals and intentions are neede to be non-intrusive.
	Algorithms	Heuristic Critic	Judgment error has replaced a heuristic procedure or missing knowledge must be obtained via heuristic procedure.
		Simple Equation Critic	User error lies in the mis- or non-use of either a qualitative or quantitive equation.
		Formal Model-Based Critic	User error lies in the mis- or non-use of a model-based reasoning procedure.
HOW	Human Computer Interface	Restricted Natural Language Critic	Textual dialogs are the only way to convey information or in-frequent use prohibits training of other communication modes.
		Command Language Driven Critic	Frequent users wish to bypass delays of other modes.
		Direct Manipulation Driven Critic	A visual metaphor of the critic can be created, the visual icon is intuitively appealing, and little training is needed to use it.
		Influencer Critics	Positive appeals to the user's intelligence will succeed and reasonable preventive measures can be defined.
	Strategies	Debiaser Critics	Negative condemnations of user's sfforts are needed and after task correction will be heeded by the user.
		Director Critics	Step by step procedures must be explained to either cause t he user to see or correct his error.

Table 2. Rules for Design and Deployment of Expert Critics.

see why this deficiency should stop XYZ from keeping technologically apace in the communications technology area.

Sally wrote a three-page report to her manager, Roger, indicating the technical issues, criteria, and thresholds the SATCOM component should be tested against, a set of constraints she had long wanted a component to overcome and that John assured her were achievable.

Within a week, Roger returned Sally's report with a note indicating Sally's assessment needed to be rewritten before he could back it, indicating three basic points: (1) refocus away from technical gadgetry alone and balance the evaluation more squarely on gains in operational performance, (2) stop viewing the component in a stand-alone fashion and instead assess how well it can be integrated with other equipment and operating and maintenance personnel and practices, and (3) summarize the most important issues and criteria only on a level that higher-level management can base decisions on.

Sally stopped by Roger's office later that day to discuss things. Roger said that he was swamped as manager of the Office of Technology Evaluation and that his three written comments to Sally were the same ones he had to write again and again to all the evaluators, no matter what their discipline. Although his evaluators were technically proficient in their respective disciplines, he wished that he had some sort of knowledge-writing aid that would help them to create appropriate evaluation issues, criteria, and thresholds and tests for new technology the first time around.

In the Army domain, as at XYZ, Sally's evaluation report is a knowledge base of the issues, criteria, thresholds, and tests that the new component must pass. That is, the report contains the rules needed to diagnose whether the component should be contracted for (that is, whether it is "healthy"). In this scene, as in the Army, the knowledge-writing aid Roger wants could be an influencer aid, a debiaser, or both. The description of these two classes of critics follows. It addresses each of the task, bias, and strategy levels of a decision network relevant to this domain. It took about two person-years of effort to develop the criticism knowledge bases for the actual U.S. Army version of this case study. This task was achieved in the first six months of 1989.

The Task Level

The case study user community includes individuals skilled in one of about 17 specific disciplines for which the Army buys new combat systems. Sally's telecommunications speciality is analogous to such a discipline. In any given discipline, the responsible individual writes a critical operational issues and criteria (COIC) document, or knowledge base, that defines the conditions under which the Army should proceed (or not) with the purchase-development of the proposed new materiel item. As in Sally's case, COIC is a 5- to 9-page decision paper that is eventually signed off by a headquarters' decision maker. It stipulates what criteria the weapon system must satisfy if the Army is to purchase it. Issues and criteria can be equated to disease hypotheses and symptoms that the

weapon system should avoid to be healthy enough to purchase.

Thus, the tasks of writing COIC are to fill in the issue and criteria knowledge base chunks. COIC is a highly structured knowledge base containing primarily logical statements that need to be tested as true or false (or for degrees of belief). Still, this arrangement is just the structural aspect of the knowledge to acquire for a given COIC. Knowing the structure is relatively trivial, the structure is given in a handbook available to all evaluators, and the structure does not provide the answers that are needed in the subtasks just mentioned.

If the structure were all that mattered, this domain could use a traditional knowledge-acquisition system to interview the human to enter another issue or criteria. To the contrary, the cognitive task analysis shows the machine must collaborate with and criticize the human in each of the subtasks of a well-formed COIC. The knowledge-acquisition system needs many critiquing heuristics, first principles, and work package question sets to help in the construction of proper issues and criteria. These needs, in turn, require a number of additional subtasks (for example, the forecasting task) to collect information about the new system, synthesize these data, and think about what is relevant or irrelevant to the COIC document.

The Bias Level of the Cognitive Task Analysis

As with the subtasks, only a sample of the numerous biases encountered in this domain appear in this section. An example bias situation exists in the subtask for developing the criterion measure for a given criterion for a given COIC issue. Figure 1 shows this subtask along with some of the related biases that I now discuss. The strategies plotted on this diagram are the focus of the ensuing subsection. As indicated earlier, the criteria are the dimensions along which the system will be measured so that a procurement decision can be made. In Sally's case, they might include items such as the number of communications successfully processed in a given time interval by SATCOM.

Figure 1 plots Roger's three complaints along with other biases found during the cognitive task analysis of the Army users. The causes of each of these biases are different, potentially unrelated to each other, and often result for different reasons. The individuals interviewed from four separate disciplines react to different organizational influences and exhibit different causes for similar biases. In effect, not all the causes of the bias are known at this point, and it isn't clear whether sufficient study could ever be done to isolate all the causes.

Like the cause, the effect of the errors in the measures varies widely. Where headquarters' personnel catch the errors and send COIC back

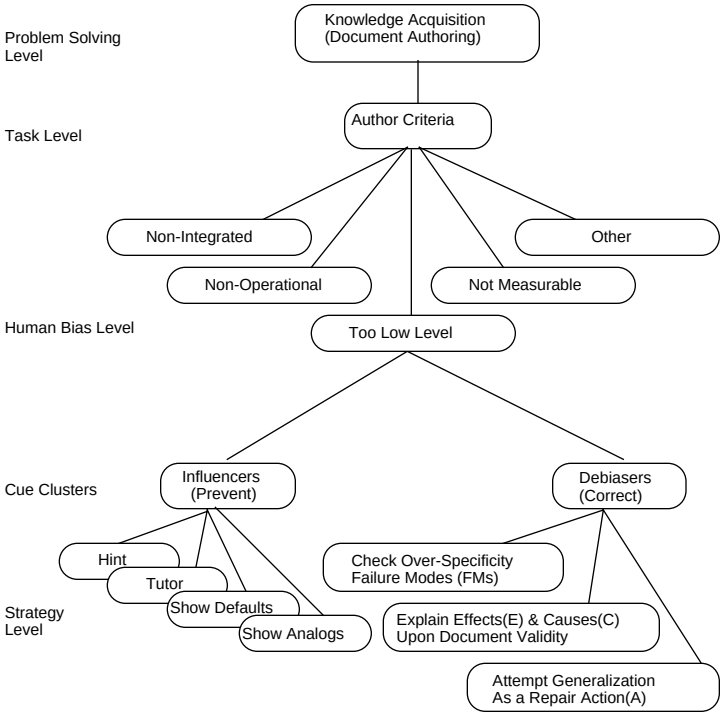

Figure 1. Cognitive Task Analysis Diagram Showing an Actual Set of Biases and Critiquing Strategies for the Criteria-Authoring Task.

for further rewrite, the main effect is a delay in processing the document. If headquarters misses the error, unneeded testing of the item might be done after the document is approved, or insufficient attention might be paid to critical dimensions of the item being procured. The latter effect is, of course, the most serious. In the extreme, it can lead to systems that are ineffective in the battlefield to be fielded, causing needless loss of life.

The process of identifying these biases involved interviewing about a dozen discipline specialists (Sally's) and seven managers (Roger's). Also, in five instances, successive drafts of COICs were chronologically analyzed with the help of one of the headquarters experts who had worked out the bugs on them. Only five could be analyzed because either the files or the relevant experts were no longer available. Thus, the cognitive task analysis covered over 22 cases of biased COIC criteria sets (plus numerous anecdotes from specific interviewees). Each case included anywhere from two to about two dozen specific bias examples.

Implementation-Level Details of the Strategies

What follows is an explanation of one strategy network for one of the biases found when domain experts authored criteria knowledge base chunks, as shown in figure 1. The other biases are subjected to a similar set of strategies; however, space restrictions, as well as readability concerns, dictate that I explain only one illustrative strategy.

The strategies shown in figure 1 implement the two top-level strategies of table 1: (1) influence the user before s/he finalizes his(her) answer (with hints, defaults, analogs, principles, examples, or repetition) and (2) debias and direct the user after s/he gives an answer. Specifically, the domain expert interacts with two separate tools when creating his(her) knowledge base. First, there is a question-asking and critiquing system that includes the influencer and debiaser strategies. There is also a graphic knowledge-level editor that cues the expert to directly edit the knowledge base. For example, s/he sees color cues that indicate incomplete elements of his(her) knowledge base. In the following discussion, I explain the types of interactions found in using tool 1 to generate a first draft of the knowledge base. Tool 2 is then used to refine the knowledge base and further guide the user's visualization of his(her) invention (although it is not described here).

The decision network established for the criteria critics includes many of the strategies in table 1. Specifically, the decision network established for criticizing during the decision problem-structuring task includes the following strategies (figure 2):

The first strategy is *leading question asking*. Many of the influencers or knowledge critics provide leading questions that attempt to place the user in a specific frame of mind to follow the prescribed cues. For example, one leading question is "Would you like to input a criterion for the effectiveness issue?" Other leading questions force the user to study the domain of the weapon system more closely and, thereby, pave the way for collecting a number of terms for the criteria phrase.

The second strategy is *Socratic hinting*. As used here, hinting is a little more direct than leading question asking in that it often states a cue. To continue the example, the question "Would you like to input a criterion for the effectiveness issue?" is accompanied by the following hint: "A good criterion normally factors in the operational and battlefield integration concerns."

The third strategy is *tutoring*. Tutoring makes use of information hiding. It is only invoked when the user fails to understand a hint. When invoked, tutoring exposes the user to a hierarchy of note cards of interlinked principles, good and bad examples, subprinciples, examples of the subprinciples, and references. The user can navigate as deep into

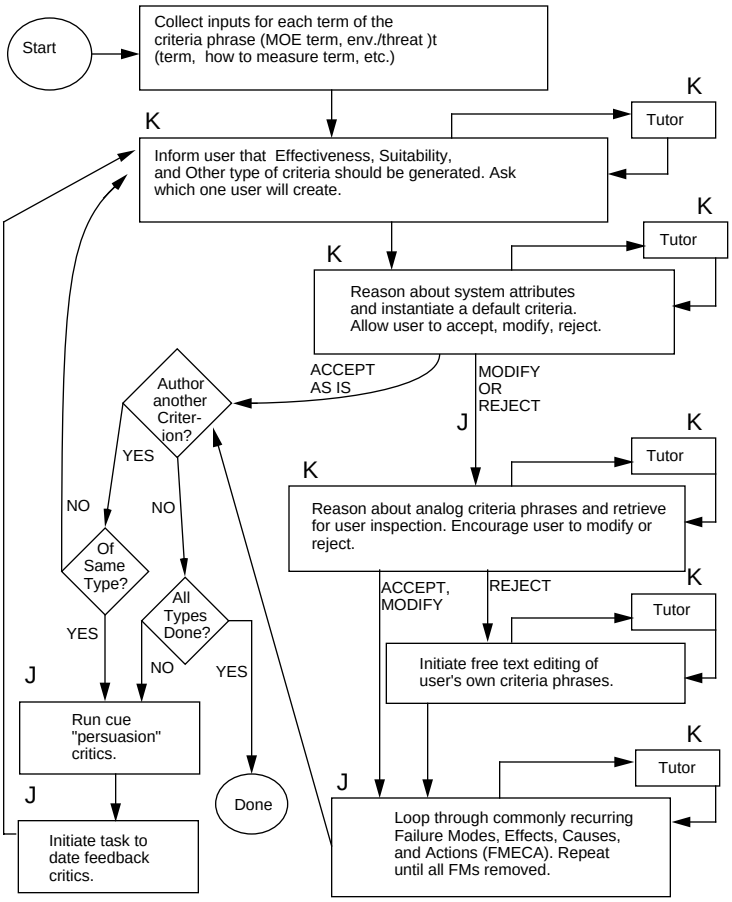

Figure 2. Decision Network of Critics for the Criteria-Authoring Task.

this hierarchy as s/he needs to to understand the associated cue. Tutoring can be set on mandatory mode for novices, which was done (at the first level of note cards) for the Army field test. Part of the first principle note card for our continuing example quotes the handbook on the high-level, operational criteria cue. It gives the decision rules that explain why its important to aggregate and orient away from strictly technical detail. Other principle note cards contain lists of dos and don'ts plus examples and citations.

The fourth strategy is *default reasoning*. When the user selects a type of criteria to input, say, satellite communications, TIME reasons from earlier collected weapon system information and instantiates a good

default phrase for the user that s/he can accept, modify, or reject. An example of a well-formed phrase that the system might suggest appears earlier at the start of this case study. If the user accepts the default as is, the user is asked if s/he wants to input another issue. If s/he wants to put in another issue of the same type, it is a deviation from the prescribed cue, and the user is sent to the persuasion critics (see later in this subsection). If s/he is finished, s/he returns to near the top of the network shown in figure 2.

The fifth strategy is *analog reasoning*. If the user wants to modify or reject the default, s/he is first sent to the analog module. There, TIME shows him(her) successful past phrases from other weapon systems (analogs) relevant to the particulars of this target weapon system. The hope is to offer another concrete modify-and-paste anchor that might appeal to his(her) intelligence before s/he commits a potential error. Analogs are accompanied by leading questions, hinting, and tutoring that provide appropriate cautions to properly adjust the analog on transferring it to his(her) knowledge base.

The sixth strategy is *repetition*. If the user rejects the default as well as all the analogs, s/he is in free-text mode, where some of the cue-use cautions are repeated.

The seventh strategy is *failure, mode, effects, causes, and actions* (FMECA). If the user modifies or rejects the default or the analogs and adds any of his(her) own phrasing to the issue, the debiasers are triggered. That is, a set of judgment-debiasing critics examine what the user authored from a number of the possible cue-underuse perspectives. For example, critics exist that could identify any of the errors associated with a poor criterion. If no error is found, the user is unaware that these critics exist. If an error is found, the FMECA paradigm involves displaying the failure mode and explaining its effect if left uncorrected, a cause, and a suggested repair action. Often, this process involves sending the user on a mandatory tour through some of the tutoring note cards s/he earlier overlooked (figure 3). In the tutoring literature, FMECA is often referred to as a theory of bugs (for example, see Wenger [1987]).

The eighth strategy is *persuasion*. If the user appears to violate a normative cue, s/he is sent to a cluster of critics that attempt to find out why s/he is taking these actions. If certain legitimate reasons are missing, the critics attempt to persuade him(her) to use the relevant cue. Persuasion usually takes the form of explaining cause and effect if the cue is ignored.

The ninth strategy is *feedback*. If all the other strategies fail to cause the user to use a given cue, the last resort strategy is to show him(her) what s/he has done and ask him(her) if s/he can bring it more in line

with the prescribed cues. For example, if the user creates four communications criteria, s/he is shown the list and asked if s/he can merge and combine any of these to make a smaller set that is more in line with the cue's objectives. The critics assist him(her) in repeatedly looping through the set of criteria until the user feels s/he is done.

Description of the Forecasting Task

The previous task description explains how one can analyze a domain to decide which critic strategies mitigate the biases and commonly recurring expert errors. Without going into the details, it was similarly determined that the following biases prevail in the forecasting subtask: In the information-acquisition stage, the biases are availability biases in the acquisition of distributional information and base-rate biases that promote the attractiveness of the concrete information the forecasters possess in their personal experience base. In the information-processing stage, the biases are habit, nonregressive effects, and adjustment from deficient anchors. In the feedback stage, the bias is the ease of recall.

The purpose of this section is to focus the reader's attention on several of the critic design parameters introduced in tables 1 and 2 but not yet discussed. First, I discuss the human factors of the human-machine interface. Figure 3 contains a listing of the dialogue but omits the actual screens that the user interacts with. Second, the decision network in this section includes a director. This case illustrates how to heuristically handle what analysts often view as a more traditional decision-analytic and quantitative subtask.

The screens, interfaces, and behaviors of the decision network of critics built for the Army problem are shown in figures 4, 5, and 6. These figures follow the strategies for critic rules in table 1 and adhere to the table 2 guidelines for deployment. In each figure, the darkened area is the principal work area. The user types answers into this space after the > sign. The answers are in bold print for emphasis. The white regions hold information the user must read or functions the user can invoke. The functions also appear in the lower-right box of each screen. These functions are primarily of three types: (1) navigational aids that allow the user to leap between different screen layers (help, principle, examples, citation), (2) aids to edit answers or replay earlier sequences (rewind and fast forward, used either to refresh the users memory or undo and update), and (3) commands to save and quit.

The influencers built for the task shown in figure 4 show up just before the task begins. That is, the user has been working in the environment for some time. S/he now reaches the point where s/he must venture a "worst mission" forecast. Five steps are shown in figure 4:

The idea of Critical Operational Issues and Criteria (COIC) is to summarize for decision makers at headquarters. The measures you've selected thus far are:

o Using standard I/O devices, SATCOM must demonstrate a 90 percent probability that it can successfully handle three User Control Interface Devices in use simultaneously operating at data rates up to 4 kbs.

o SATCOM will communicate through the atmospheric conditions produced by a high altitude nuclear burst.

o Receptions successfully completed.

o Successful transmissions sent in a dirty battlefield.

The following are inappropriate:

o Receptions successfully completed.
o Successful transmissions sent in a dirty battlefield.

for the following reason.

These MOEs are often too specific for a COI. A higher level composite MOE which incorporates these MOEs is preferable.

A more appropriate version is:

o Communications successfully completed.

Do you:

I. Accept as is
J. Edit further
K. Reject

> 2

> Communications successfully completed in a dirty battlefield.

What 2 to 3 word name would you like to save this under on the dendritic?

> Receive/Transmit

Figure 3. Example of Epistemological Assessment to Debias Expert Intuition after an Error.

First, the system asks for the expert's forecast of the worst mission. It already knows that the materiel of interest is a helicopter. It also already elicited a list of all missions that the helicopter might be sent on.

Second, before the user is allowed to type his(her) answer, an active influencer attempts to alert him(her) to the potential for biased reason-

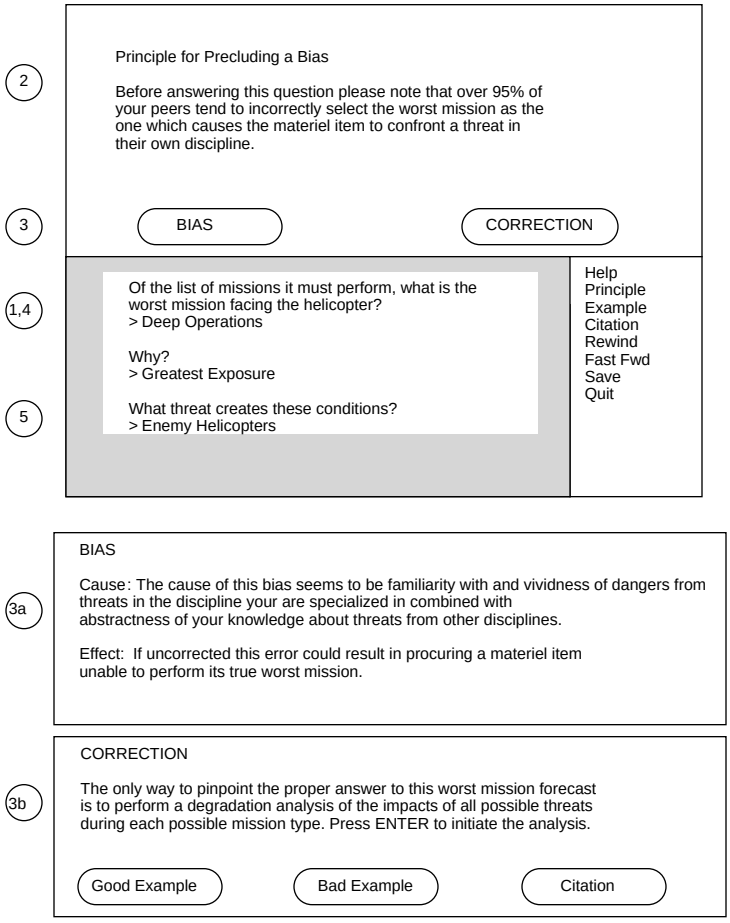

Figure 4. Active and Passive Influencers for a Forecasting Task.

ing. Here, it warns the user that concrete experience in one's own discipline tends to obscure other information. The message is kept short.

Third, two buttons can be pushed to provide deeper explanations of the bias (figure 4, part 3a) and corrective procedure (figure 4, part 3b). These buttons produce passive critics for the user who wants to interrupt his(her) own agenda, having an alternative approach available. The correction note card contains three additional buttons that place the user in an optional tutoring mode. This note card also contains an instruction to select ENTER. By pressing ENTER, the user initiates a director. The director walks the user through a formal reasoning model

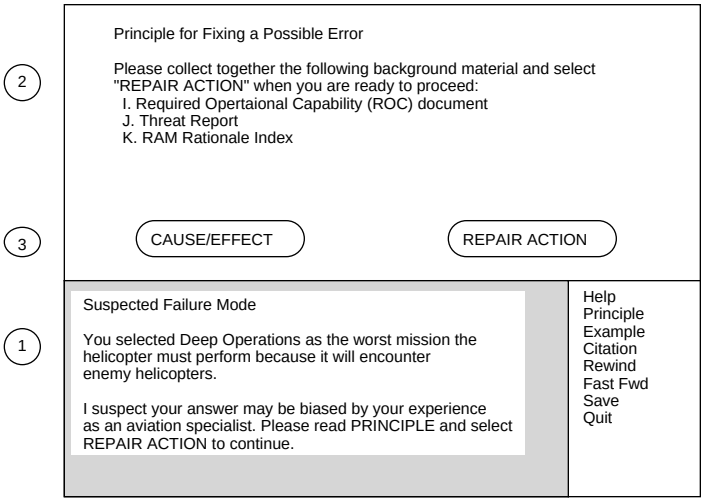

Figure 5. Active and Passive Debiasers for a Forecasting Task.

of the forecasting task. In all the screens, the users know that ESC will return them to the dark-shaded work area.

Fourth, the user presses ESC and chooses to proceed on his(her) own. S/he opted out of the further tutoring and formal reasoning assistance. S/he answers the original question with "deep operations." This phrase is a designator for missions that require penetration deep behind the enemy line.

Fifth, the system follows with two questions that set the stage for identifying that a cognitive bias leads to a known failure mode. In an earlier dialogue, the system learned that the user is an expert at the Aviation Center. The answer to the last question, "enemy helicopters," is a trigger to the critic that the user relied on concrete information to get his(her) answer. That is, the critic knows the user has experience with aviation threats. Also, its dictionary equates "enemy helicopters" with the aviation domain. Thus, the last question is a trap that the user falls into.

At this point, a debiaser concludes what bias occurred. It also attempts corrective action. Several biases can occur in any given task. Recall that six distinct biases can arise in the Army forecasting task. Each bias might require a different trap and debiaser. For this reason, the critic avoids fixed-dialogue sequences. Instead, a reasoning engine infers the correct set of debiasers to invoke. The engine bases this inference on the facts in working memory about the user and the traps s/he fell into. In the same vein, the reasoning engine branches through a

tree of possible traps that might catch the user. Thus, the why question in step 5 of figure 4 yields no trap for the user. His(her) answer is valid; so, the engine branches to a second trap that is successful.

In figure 5, the debiaser adheres to the FMECA method. Here, "failure mode" is a trap checker, or trigger, to verify the bias. EC (effect and cause) explains the error and its associated cause and effect. The action in this case suggests the solution and invokes three director steps:

First, the critic tells the user s/he fell into a common failure mode. It also tells the user its placing him(her) in a temporary machine-directed sequence. The influencer (figure 4, step 2) had only superficial and generic knowledge about types of user errors. This critic (figure 5, step 1) believes it has deeper insight into the user's cognitive judgment processes. For this reason, it is more assertive. The critic uses words such as "suspected failure mode" because it is still conceivable that the user put in the correct worst mission answer.

Second, the critic tells the user to get ready for a formal reasoning procedure (director). There is no simple way to test if the user gave the right answer without doing this analysis. The preparation includes collecting the important sources of abstract information the user previously overlooked.

Third, the two buttons contain active and passive debiasers. The cause-effect button is passive. It serves as motivational information, should the user want it. Selecting it leads to a display of the same bias note card shown in part 3a of figure 4. The repair-action button is no longer passive. It represents the user's only path for proceeding toward the completion of his(her) document (other than selecting "Quit"). Again, it displays the same note card as the earlier correction card shown in part 3b of figure 4. This time the critic deactivates the ESC key.

These three steps cause the user to go through about one hour of computation and literature searches once s/he obtains the proper documents. These computations provide estimates of the degree of degradation experienced by each mission the materiel might be taken on. The director guides the user through a knowledge-rich version of a formal reasoning process. This process consists primarily of following a checklist. The critic runs simple scoring or figure-of-merit equations to compute the type of end results that are depicted in figure 6a. (Purposely falsified data have been used here to avoid any breach of security. However, neither this practice nor the precise meaning of each of the threats and mission list entries is important to understanding how the director works.) In brief, the director gets the user to display the distribution of threats the helicopter will encounter on each mission type. The user then estimates three sets of numbers. These numbers represent the percent of time spent on each mission, the percent of

(A): Completed Degradation Analysis Showing Distributional Information

MISSION LIST	Percent of Time on that Mission	THREAT (Exposure % / Degradation) Tank	DTR	Air	Enemy	Personnel	INDEX
Close Operation	30	50/H	40/m	0	10/l	0	72
Deep Operation	25	50/H	30/H	0	20/M	0	70
Rear Operations	15	0	10/H	5/L	85/M	0	0
Air Combat	25	50/H	0	40/M	5/H	5/H	98
Fire Fighting	5	0	0	0	0	0	0

(B): The Critic's Suggestion for a Correct Answer is Usually Accepted

1

Suggested Action

You have earlier given an incorrect answer of

"Deep Operations"

I would like to replace your ealier answer with

"Air Combat"

2

What would you like to do with my suggestion?

1. Accept as is
2. Edit further
3. Reject

> 2
> Air Combat plus Operations

Help
Principle
Example
Citation
Rewind
Fast Fwd
Save
Quit

Figure 6. Results of Using Critics in the Forecasting Task. (a) A completed degradation analysis showing distributional information. (b) The critic's suggestions for a more correct answer, which is usually accepted.

time exposed to each threat during a given mission, and the amount of likely degradation (high, medium, low, or none for 3, 2, 1, or 0 points) the helicopter will incur from meeting this threat.

Although this analysis is crude, it forces the expert to confront the range of distributional data. It makes him(her) rethink his(her) original answer. In each system run, the Army users experienced an "aha" factor as they realized their own error even before the machine rolled up the numbers into an index and computed the correct answer on its

own. For the sake of completeness and for the eventuality that some users might need it, the machine computes the index in the right-hand column of the table shown in figure 6a. A score of 1.0 is a mission of maximum danger—high losses 100 percent of the time. When done, the director returns control to the debiaser. The debiaser performs the following simple operations, which are also listed in figure 6b:

First, the debiaser ranks the list of missions according to the index. It compares the user's original answer with the worst mission from the table. From this information, it concludes a suggested action. It displays the suggestion on the screen.

Second, the user is given three options. Depending on which option is selected, the reasoning engine might need to search for more traps and debiasers. In this case, the user adopts a decision rule that every mission with a high index is bad enough to be the worst. S/he creates a merged name for the worst mission. Because the name includes the machine's opinion of the worst mission in the string, no further reinforcing or debiasing occurs.

How the Application Is Programmed

The TIME application is programmed in a generic language and environment called COPE. COPE is a criticism-based problem-solving and knowledge-acquisition toolbox available on 286- and 386-chip machines under DOS or 68020-based platforms under UNIX. The author and his colleagues created COPE to facilitate research into human-computer collaboration and to act as a criticism-based problem-solving test bed in which numerous applications and experiments could be attempted (IntelliTek 1989b, 1990; Silverman et al. 1987). The COPE architecture supports the life cycle of the development, use, and maintenance of TIME or other applications. This section discusses the development and use steps. Maintenance and Administration of TIME's Knowledge Bases covers the maintenance step.

The developer of a COPE application prepares the necessary objects and rule trees using tool 2, which was mentioned earlier. If s/he is an advanced C programmer, s/he can also decide to extend tool 1's function library in any of a number of directions depending on the analytic, symbolic reasoning, or screen and user interface needs of the application.

The COPE user, in turn, runs tool 1 to construct a case in a given domain (for example, a critical operational issue and criteria case for TRADOC). Tool 1 runs the rule trees that interview the user to collect a new knowledge base or that detect, criticize, and repair problems caused by the user's input. In general, tool 1 reacts to the user's an-

swers to the previous knowledge elicitation questions and fires the proper rule tree objects plus library functions.

The critiquing involves some additional objects of a given rule tree that invoke checking functions: a differential analyzer and a dialog generator. The differential analyzer examines the user's answer to a just-asked question and compares this answer with the target answer stored in an expertise module. This module is often another rule tree or object slot of answers that exemplify what an expert would offer when performing the same task. Differences beyond an acceptance threshold are passed to a file of errors. This file collects biases, opportunities not taken, and so on.

The dialog generator receives the file of errors from the differential analyzer, parses them into user-presentable forms, and displays them on the screen. Often, critics fire canned textual note cards to converse with the users, although these note cards can be organized in a hierarchy, as shown in figures 4, 5, and 6. Critics also include a language instantiation capability that allows them to improve the dialog with the user by binding text variables to context-sensitive strings that are prestored in a database of dialogue utterances. Also, some critics alter what is shown to the user based on insights into the user's skill level collected through direct inquiry.

Using the tool 1 inference engine and a rule tree representation scheme facilitates the creation and experimentation with the critics. Changing an influencer to a debiaser, a debiaser to a director, and so on, is often just a matter of repositioning where the critic is fired in the rule tree sequence (and probably also editing its textual note cards). In this fashion, the decision network of critics is molded into the knowledge-acquisition sequence in the least intrusive fashion.

As the dialogue proceeds, the domain case, or new knowledge base, resulting from the interview is written, piece by piece, into a case database on the computer disk. A transformer module converts this case database into a case tree that can be read by tool 2. Tool 2, the graphic knowledge base editor, allows the user to inspect the new knowledge base constructed by the guided and critiqued interview of tool 1. This process allows the user to visualize what s/he wrote and make any modifications using a direct manipulation interface.

After any editing, the final case can be converted to a hard-copy textual report using WRITER. Alternatively, this case can serve as a new knowledge base for an expert system shell. This new knowledge base can also be passed to an administrator module for assimilation by an analogical reasoning agent that helps COPE learn more about the domain as it is used more (see Maintenance and Administration of TIME's Knowledge Bases).

This overview covered the input, output, and modules and algorithms of the COPE language underlying the TIME application. To program TIME, it was useful to create about 60 rule trees, averaging about 25 objects each, for a total of approximately 1,500 objects. The objects of each tree form on average about 2 dozen rules, hence the estimate of approximately 1,500 rules for TIME as well. Finally, each of the 60-odd trees has about 8 or 9 objects that directly interact with the user; each of these trees has an average of 4 note cards, hence the estimate of 2,000 note cards in the TIME application.

Innovation

Innovation is a relative term that is difficult to judge in the short run. The innovations described here are potential advances, but only time will tell how important they truly are.

Innovation in the Milestone Decision Process

The Army wants to exploit the potential of AI to reduce the heuristic reasoning errors, biases, and foibles, plus the communications obstacles, that traditionally slow system acquisitions. From the perspective of the military, the innovations that AI offers with TIME include the following:

First, TIME successfully integrates and interactively communicates three types of knowledge that were previously difficult to factor into the document-writing process. The previously used hard-copy handbook, which was difficult to read, held many useful principles, good and bad examples, and format instructions that TIME now delivers to the user as needed. The knowledge of the headquarters decision makers of the latest policies, preferences, and decision rules is now delivered by TIME to affect the content and emphases of the material in the document. TIME offers an online, easy-to-access copy-paste-modify library of previously successful issues and criteria phrases from past documents in the domain of the individual authors. TIME brings all three forms of knowledge together—the handbook information, the decision makers' knowledge, and the online library—and shows them to the user at appropriate intervals in the authoring and critiquing process.

Second, TIME persistently refocuses the user away from distractions, selective perceptions, and other potential errors and biases. TIME is the first expert critic system built and deployed for the U.S. military. It not only delivers the knowledge, it also verifies that the author is making maximum use of this knowledge. Thus, TIME serves as a forerunner for similarly transforming the way hundreds of other types of milestone documents are produced in the military and elsewhere.

Third, TIME helps manage knowledge as a corporate asset. Given the

mandatory rotation of personnel in the military every two to three years, an important innovation is to capture and retain what is learned so that the next person on the job can easily maintain continuity. The knowledge-acquisition, case-based reasoning, and dynamic memory features of TIME represent an innovative step toward better managing knowledge assets.

Innovation in the AI and Decision Support Fields

This case study illustrates the applicability of COPE to a broad array of reasoning, problem-solving, and knowledge-acquisition subtasks. The TRADOC domain requires that COPE offer criticism-based problem solving in numerous tasks. These tasks cover knowledge base acquisition, report writing, forecasting, and quantitative estimating. The decision network of positive and negative criticism strategies implemented to serve these task requirements cover hinting, default and analogical reasoning, tutoring, debiasing, persuading, and so on. To cover these requirements, TIME synthesizes and extends many ideas from a broad array of AI technologies.

The TRADOC domain also serves as a robust test of COPE's theory of bugs in expert intuition. Numerous cognitive and judgment biases were encountered and successfully reduced or eliminated after extensive field tests. These study results confirm and extend much of the critic design information presented here. Research is ongoing to empirically isolate and confirm additional design insight for the critiquing paradigm.

Finally, this case study demonstrates how TIME and COPE are occupying the space between knowledge-rich, replace-the-expert technology, such as expert systems, and theory-rich, support-the-expert technology, such as decision analysis. Criticism-based problem solving provides a knowledge-rich, heuristic approach to decision-theoretic, support-the-expert situations. This area of investigation is relatively new not only in knowledge acquisition but also in problem solving at large. The TIME case study advances and extends the criticism approach.

Deployment and Use

As already mentioned, the current version of TIME was knowledge engineered in the first 6 months of 1989. In August 1989, 10 Army authors from 4 separate disciplines met in a 4-day workshop to validate the content of TIME's knowledge bases. These participants covered 3 levels of skill in each discipline: novice, intermediate, and expert. This exercise was primarily paper based: A real document was authored by the group while they interacted with a 300-odd–page notebook of the screens

they would ultimately see in the finished system. Throughout the workshop, thinking-aloud protocols were recorded, as were user reactions to the screen-by-screen information. After the workshop, the participants returned a detailed questionnaire describing their reactions to many of the system's features.

Also after the workshop, the participants returned to their four installations (one installation for each discipline) and drafted four assessments of the validity and potential usability of the system. Based on these assessments, a follow-up effort was approved, and the construction of the system began in earnest. From September 1989 until November 1990, the equivalent of four full-time people (1) incorporated the improvements suggested by the workshop participants; (2) coded the knowledge bases in the COPE language; (3) optimized and extended some of the COPE features and screen interfaces for the Army-approved Zenith 248 environment; (4) debugged the TIME knowledge bases; (5) prepared training materials; and (6) further refined TIME's knowledge bases, interfaces, user dialogue modes (expert and novice modes are possible), help facilities, and numerous other features.

From June 1990 until the first week of December 1990, a series of field tests were conducted, one at each of the 4 installations that sent participants to the original workshop. The purpose of these field tests was initially to identify bugs that needed to be removed and later to verify the system was ready for deployment. Each field test consisted of the same mix of participants that was sought for the workshop. This time each user tackled a separate, real-world problem and generated his(her) own (COIC) document with the aid of the COPE-TIME system. The field tests began with a ⅔ day of training and user qualification, followed by as many as 3 days of a user running and testing the system. The documents produced during the field tests were evaluated by the participants' immediate supervisors. The supervisor also prepared an installation-level assessment of whether TIME passed the test and should continue being funded. The criteria for evaluating the benefits of the system were documented in headquarters' instructions to the installations:

(1) Can players install the program on the computer using con
tractor provided instructions?
(2) Can players clearly understand instructions and information
displayed by the program?
(3) Does the program teach/guide the player to make assess
ments of the following aspects of the (weapon) system
(A) Operational mode summary/mission profile,
(B) Threat,

(C) Need,
(D) Operational characteristics and supporting rationale,
(E) Doctrine and tactics?

(4) Are error indications and explanations accurate and understandable?
(5) Does the program identify inconsistent player responses and input?
(6) Are the draft COIC (documents) produced essentially in proper format and consistent with current guidance?
(7) Did using the program result in time or effort savings?
(8) Additional comments/observations. (TRADOC 1990, pp. 4–5)

Based on successfully meeting these criteria, TIME graduated from the field-test stage. It was disseminated for use in the four original installations plus two new ones added in late January 1991. During the period of use from January to April, approximately 2 dozen users produced documents with TIME, serving in the "maiden voyage" capacity. All 17 user sites will eventually receive TIME, annually producing as many as 600 reports.

As of this writing (early March), there are only 2 sets of results from the maiden voyage. In both cases, the users found their initial experience with TIME to be slow and painstaking. One user stated, "I used to be able to write a COIC in 2 hours. I've already spent 4 hours with TIME and I'm only half done." This reaction is precisely what TIME should precipitate if it is to reduce the errors, making the user do a more thorough job. The second user indicated that it took him 3 days to produce his first COIC, and he was initially discouraged with TIME. However, he now sees its value and believes he can use it to write a better COIC in under a day. He is now eagerly training all his subordinates to use it.

Payoff

Payoff, like innovation, is another area that is difficult to fully and accurately assess. Also, only preliminary data and expectations are currently available. When measuring payoff, as we must, in terms of the reduction in the frustration of all participants, a lessening of errors in draft documents, and the satisfaction of the eight criteria, then much of the acknowledgment of the system's benefit must come from the sponsors and the users' supervisors. This situation is particularly true given the security and inaccessibility factors that prevent the author from making precise payoff measurements in this environment.

In terms of the payoff, TIME was used to produce 12 documents during the field tests. It is currently being used to create as many as 2

dozen more. Because these documents are passing the eight criteria, they are returning benefits to TRADOC in terms of the originally stated objectives. That is, with TIME, (1) users of all skill levels are receiving error and bias reduction support; (2) novice and intermediate users are being guided and tutored past commonly recurring difficulties; (3) headquarters' personnel are benefitting from reduced workload, higher-quality documents, and faster turnaround; and (4) from a more subjective perspective, the overall morale of the authors has been improved by headquarters' effort to improve their work environment and reduce frustration throughout the organization. The tentative attempts to interview users and their supervisors verify that these items are correct.

For reasons unrelated to the deployment of TIME, the five-person (plus secretarial support) headquarters branch that reviewed all COIC documents was closed in late 1990. The review function (Roger's job) was reassigned to 90 officers stationed elsewhere in the headquarters operation. With this diffusion of expertise, a number of people now view TIME as an increasingly important repository and the place where corporate memory assets can and must be managed. This situation raises a number of interesting and important payoff issues for the maintenance features of the TIME system.

Maintenance and Administration of TIME's Knowledge Bases

After passing the eight criteria for field-test verification and assuring the payoff in terms of these same criteria during initial deployment, a third development-period effort was awarded. This effort is under way and will be concluded by September 1991. In particular, it was decided that development of the TIME maintenance and administrator module should be undertaken after the initial deployment stages were complete. Hooks and interfaces to this module were previously created. However, it was thought prudent to undertake this process as a "backfill" operation to avoid overloading the development team during the earlier phases. This approach also ensures the team members are still around for much of the first year after deployment. Following September 1991, the maintenance and administration of TIME will fully reside with Army personnel. The developer will no longer be under any contractual responsibility to the Army.

To assure the Army can maintain the system on its own, two types of knowledge bases must be able to be updated as easily as possible: the static or fixed knowledge bases, which hold the basic guidance on how to author a good document plus the latest decision maker preferences and heuristics, and the dynamic memory of completed documents,

which serves as a cut-and-paste world of cases that can be retrieved and adapted for reuse.

Keeping the fixed knowledge base elements up to date is potentially difficult for three reasons. First, the advice in these elements regularly changes as new decision makers assume office, and Congress passes new regulations. Second, TIME is relatively large sized. As already mentioned, it contains over 5,000 knowledge chunks. Third, the maintainer must learn the COPE language to modify a COPE knowledge base. The goal of the administrator module is to minimize the effort needed to overcome the second and third sources of difficulty. The first item is beyond my purview and is the reason an administrators module is useful. The solution to these problems includes a visual index of the knowledge. This index is supplemented by a hypertext-based online manual, a graphic editor for modifying the rule trees, and a change-control assistant that warns of errors committed and unfinished changes. Even with these aids, it is expected that a TIME administrator will have to spend almost a week of full-time effort to initially learn the COPE language and become adept at update actions.

Maintaining the dynamic memory is far easier than updating the fixed memory. In particular, a previously built and verified case-based reasoning system (IntelliTek, 1989a) is being incorporated into the COPE environment for dynamic memory updating. The 300-odd analogs now available in TIME were all hand coded into databases that are accessed in a context-dependent manner. Each finished document must now similarly be hand coded to add it to these databases. This process is a waste of effort because what the computer collects through knowledge acquisition, it should be able to remember. Also, once the deployment of TIME is complete, and as many as 600 new cases are generated each year, only the computer will be able to keep up with this flow rate in a reasonable time frame.

In concept, TIME can automatically perform this task on its own with the aid of a form of case-based reasoning. Several concerns, however, are being addressed in this final stage of effort, including (1) providing a reasonable scheme by which each of the 17 installations can extend the dictionary of terms in the world model in different directions but simultaneously allow headquarters to keep its version of TIME abreast of all the changes, (2) offering a password-protected function that can be used to delete entire groups of cases (or portions of these cases) that used to be successful examples but now violate the latest approved guidance for the construction of good documents, and (3) assuring that documents aren't assimilated into the case base until they are approved and that their dozens of rules and phrases are properly cross-indexed when they are assimilated. Numerous subtleties are connected

with each of these three concerns that are not addressed here.

The goal of this section was to introduce the plan for the final stage of postdeployment effort. The AI and computer science fields, at least in this instance, already have the technologies needed to successfully manage knowledge as a corporate asset. Aside from scaleup, the challenges in applying these technologies are largely organizational rather than technical. They are (1) keeping physically distributed versions in sync when security, budgetary, and organizational factors come into play and (b) balancing the legitimate desire of headquarters to retain control of the knowledge considered acceptable and the equally legitimate need for the field personnel to specialize and advance the precision of the case base residing at their location. Adapting the technology to suit the real concerns of the various interest groups is an important sociotechnical challenge that will be addressed in this final stage of the application.

Concluding Remarks

The TIME application is interesting as an organizational support system. It helps headquarters communicate good job practice information to the field, and it reduces the number of field-created errors and biases that headquarters must deal with. At the field level, TIME reduces the frustration of repeatedly receiving marked-up drafts back from headquarters. It simultaneously supports the field personnel with libraries of examples, principles, analogs, defaults, and so on, that speed their document-authoring task. Finally, for the organization as a whole, TIME serves as a knowledge-capture device that holds the promise of helping to better manage knowledge as a corporate asset.

Technologically, TIME is an example of how a wide variety of hardware, AI, and decision science techniques can be combined to solve the problems encountered in organizations. It addresses the interaction and problem-solving needs of large groups of collaborating employees. From a hardware perspective, much effort was expended to assure TIME could deliver its capabilities into the existing automation environment of low-cost desktop personal computers and local and wide area networks. From the AI perspective, it was necessary to combine expert critiquing systems, knowledge-acquisition systems, hypertext, intelligent tutoring, and case-based reasoning into TIME. From a decision support perspective, it was useful to exploit psychological models of cognitive bias. It was also necessary to adapt math-based, theory-rich decision aids into knowledge-rich, heuristic counterparts that are more natural and acceptable to the users.

Although TIME is the first system of its type in the document genera-

tion process of acquiring large-scale systems, it will not be the last. TIME has transformed one step of the process in an operation fraught with difficulty and frustration into a showcase for how the other steps might similarly function. By being innovative and applying new technology, the Army has added a new strategic weapon in the fight to improve the system acquisition process.

Acknowledgments

The financial sponsorship of the U.S. Army Training and Doctrine Command is gratefully acknowledged, as is the contractual assistance of the Defense Systems Management College (PMSS Directorate). When he was at TRADOC, Don Reich showed great vision in starting us down this road. Also, the support of two Small Business Innovative Research Phase II grants (NASA and WSMR/VAL) and several years of Research Incentive Awards from the George Washington University were instrumental in the creation of COPE, the critic test bed used here.

I share the IAAI honor with Greg Wenig, Toufik Mehzer, Bill Rodi, Inhou Chang, and a cast of other helpers too numerous to name here. Also, I thank Mike Donnell who prodded me to write the TIME application in a cogent fashion. All these people and organizations are relieved of responsibility for any erroneous opinions, findings, or conclusions offered here: The author alone bears this responsibility. The Army is also commended for taking a leadership role in admitting where biases occur so that improvement efforts could be attempted. More organizations need to be so forward looking if this field of endeavor is to be advanced.

References

IntelliTek. 1990. COPE User's Guide, IntelliTek, Potomac, Maryland.

IntelliTek. 1989a. ARIEL User's Guide: A Computational Model of Case-Based Reasoning, IntelliTek, Potomac, Maryland.

IntelliTek. 1989b. Knowledge Engineer's Guide to COPE, IntelliTek, Potomac, Maryland.

Kahneman, D.; Slovic, P.; and Tversky, A. 1982. *Judgment under Uncertainty: Heuristics and Biases.* Cambridge: Cambridge University Press.

Klein, G. A., and Calderwood, R. 1986. Human Factors Considerations for Expert Systems. In Proceedings of the National Aerospace and Electronics Conference, 921–925. Washington, D.C.: IEEE Computer Society.

Langlotz, C. P., and Shortliffe, E. H. 1983. Adapting a Consultation Sys-

tem to Critique User Plans. *International Journal of Man-Machine Studies* 19: 479–496.

Miller, P. L. 1983. ATTENDING: Critiquing a Physician's Management Plan. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 5(5): 449–461.

Roth, E. M.; Bennett, K. B.; and Woods, D. D. 1988. Human Interaction with an Intelligent Machine. In *Cognitive Engineering in Complex Dynamic Worlds,* eds. E. Hollnagel, M. Mancini, and D. Woods, 23–70. New York: Academic.

Silverman, B. G. 1992. Building Expert Critics: Unifying Theory and Knowledge to Reduce Expert Error. Unpublished manuscript.

Silverman, B. G. 1991. Expert Critics: Operationalizing the Judgment-Decision-Making Literature as a Theory of "Bugs" and Repair Strategies. *Knowledge Acquisition*. Forthcoming.

Silverman, B. G. 1990. Critiquing Human Judgment Via Knowledge-Acquisition Systems. *AI Magazine* 11(3): 60–79.

Silverman, B. G.; Fritz, D.; Teaque, A.; and Baramvik, S. 1987. COPE: A Case-Oriented Processing Environment. In Proceedings of the European Computing Conference, 399–417. Avignon, France: European Computing Conference.

U.S. Army Training and Document Command. 1990. Memorandum, ATCD-ET, 4–5. Fort Monroe, Va.: U.S. Army Training and Document Command.

Wenger, E. 1987. *Artificial Intelligence and Tutoring Systems*. San Mateo, Calif.: Morgan Kaufmann.