Technical Report 1147

A Dialog-Based Intelligent Tutoring System For Practicing Battle Command Reasoning

Joan M. Ryder
CHI Systems, Inc.

Arthur C. Graesser
University of Memphis

Jean-Christophe Le Mentec
CHI Systems, Inc.

Max M. Louwerse
University of Memphis

Ashish Karnavat & Edward A. Popp
CHI Systems, Inc.

Xiangen Hu
University of Memphis

June 2004

BEST AVAILABLE COPY

20040809 075

United States Army Research Institute
for the Behavioral and Social Sciences

Approved for public release; distribution is unlimited.

U.S. Army Research Institute for the Behavioral and Social Sciences

A Directorate of the U.S. Army Human Resources Command

ZITA M. SIMUTIS
Director

Research accomplished under contract
for the Department of the Army

Chi Systems, Inc.

Technical Review by

Joseph Psotka, U.S. Army Research Institute
William R. Sanders, U.S. Army Research Institute

NOTICES

DISTRIBUTION: Primary distribution of this Technical Report has been made by ARI. Please address correspondence concerning distribution of reports to: U.S. Army Research Institute for the Behavioral and Social Sciences, Attn: DAPE-ARI-PO, 2511 Jefferson Davis Highway, Arlington, Virginia 22202-3926

FINAL DISPOSITION: This Technical Report may be destroyed when it is no longer needed. Please do not return it to the U.S. Army Research Institute for the Behavioral and Social Sciences.

NOTE: The findings in this Technical Report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

REPORT DOCUMENTATION PAGE

1. REPORT DATE (dd-mm-yy) June 2004	2. REPORT TYPE Final	3. DATES COVERED (from. . . to) 30 Sep 01 to 29 Sep 03

4. TITLE AND SUBTITLE A Dialog-Based Intelligent Tutoring System for Practicing Battle Command Reasoning	5a. CONTRACT OR GRANT NUMBER DASW01-01-C-0040
	5b. PROGRAM ELEMENT NUMBER 0602785A
6. AUTHOR(S) Ryder, J. M. (CHI Systems, Inc.), Graesser, A. C. (University of Memphis), Le Mentec, J. C. (CHI Systems, Inc.), Louwerse, M. M. (University of Memphis), Karnavat, A., Popp, E. A. (CHI Systems, Inc.), Hu, X. (University of Memphis)	5c. PROJECT NUMBER A790
	5d. TASK NUMBER 211
	5e. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) CHI Systems, Inc. 1035 Virginia Drive Fort Washington, PA 19034 Institute for Intelligent Systems 403C FedEx Institute of Technology The University of Memphis Memphis, TN 38152	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) U.S. Army Research Institute for the Behavioral and Social Sciences ATTN: DAPE-ARI-IK 2511 Jefferson Davis Highway Arlington, VA 22202-3926	10. MONITOR ACRONYM ARI
	11. MONITOR REPORT NUMBER Technical Report 1147

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

Report developed under a Small Business Innovation Research Program 00.2 contract for topic OSD00-CR02

14. ABSTRACT *(Maximum 200 words)*:

This Phase II Small Business Innovation Research (SBIR) developed a dialog-based intelligent tutoring system (ITS) for interactive self-training of battle command reasoning. The system, called "Automated Tutoring Environment for Command" (ATEC), adapted the dialog management capability from AutoTutor (a dialog-based tutor developed by Graesser and colleagues at the University of Memphis) and integrated it with a cognitive model-based instructional agent (using CHI Systems' iGEN cognitive agent framework). The ATEC system presents a battlefield situation and then initiates a dialog between a virtual mentor (instructional agent) and a student as they collaboratively discuss the situation. The virtual mentor poses questions, evaluates student responses, determines the sequence of questions, and ultimately assesses performance on the basis of the specificity of questioning and the depth of probing and hinting that is needed to adequately answer the questions. The results of the ATEC development effort showed some of the capabilities and limitations of tutorial dialog systems, and indicated areas for additional research and development.

15. SUBJECT TERMS

Instructional Agent	Tutorial Dialog	Interactive Human-Computer Dialog	Deliberate Practice	
Training	Human Performance	Think Like A Commander	Web-Based Instruction	SBIR Report

SECURITY CLASSIFICATION OF			19. LIMITATION OF ABSTRACT	20. NUMBER OF PAGES	21. RESPONSIBLE PERSON (Name and Telephone Number)
16. REPORT	17. ABSTRACT	18. THIS PAGE			Dr. James W. Lussier (502) 624-3450
Unclassified	Unclassified	Unclassified	Unlimited		

Technical Report 1147

A Dialog-Based Intelligent Tutoring System For Practicing Battle Command Reasoning

Joan M. Ryder
CHI Systems, Inc.

Arthur C. Graesser
University of Memphis

Jean-Christophe Le Mentec
CHI Systems, Inc.

Max M. Louwerse
University of Memphis

Ashish Karnavat & Edward A. Popp
CHI Systems, Inc.

Xiangen Hu
University of Memphis

U.S. Army Research Institute for the Behavioral and Social Sciences
2511 Jefferson Davis Highway, Arlington, Virginia 22202-3926

Armored Forces Research Unit
Barbara A. Black, Chief

June 2004

Army Project Number 2O262785A790	Personnel Performance and Training Technology

Approved for public release; distribution is unlimited

FOREWORD

The U.S. Army Research Institute for the Behavioral and Social Sciences (ARI) conducts Training, Leader Development, and Soldier research for the Army. Largely, the ARI mission involves taking proven methods in the behavioral sciences and applying them to significant Army problems. In addition, a smaller portion of the ARI effort involves attempts to develop new advanced methods to meet future Army requirements. This report describes work of the latter kind; an attempt was made to apply intelligent tutor technology, which has lately become practicable for training procedural tasks in well-defined domains, to battle command reasoning, a difficult cognitive task. This report describes a Phase II Small Business Innovation Research Program (SBIR) effort that involves developing an intelligent tutoring system for high-level battle command reasoning skills. Research of this nature tends to be higher risk, and this project was no exception. At its conclusion, ARI researchers concluded that although substantial advances have been made in computerized training systems during the last decade, automated tutoring of battle command reasoning is still beyond the current state of the technology and is not likely to be a feasible solution to Army training requirements in the near future. Further, it raised questions about whether the potential value of the future technology justifies expending Army resources in development efforts. Nonetheless, the failed effort to develop a workable prototype had some value. It showed some of the capabilities and limitations of tutorial dialog systems. It advanced the methods for developing intelligent tutoring systems in domains that are appropriate. Further, when briefed to training developers for Future Combat Systems it provided them with a realistic assessment of future capabilities upon which to base their design.

This project is part of ARI's Future Battlefield Conditions (FBC) team efforts to enhance Soldier preparedness through development of training and evaluation methods to meet future battlefield conditions. This report represents efforts for Work Package 211, Techniques and Tools for Command, Control, Communications, Computer, Intelligence, Surveillance, and Reconnaissance (C4ISR) Training of Future Brigade Combat Team Commanders and Staffs (FUTURETRAIN).

Initial work in the project was presented at the 2002 Annual Meeting of Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC). At the conclusion of the project, results were presented to representatives of the Armor School responsible for developing and conducting training and to the training developers for Future Combat Systems acquisition program.

STEPHEN L. GOLDBERG
Acting Technical Director

A DIALOG-BASED INTELLIGENT TUTORING SYSTEM FOR PRACTICING BATTLE COMMAND REASONING

EXECUTIVE SUMMARY

Research Requirement:

Expert thinking strategies, such as those exhibited by successful Army commanders, are often well understood conceptually but not applied routinely during realistic tactical problem solving by less experienced commanders. The goal of this effort was to develop an intelligent tutoring system (ITS) for interactive self-training of thinking skills, such as battle command reasoning, within a deliberate practice framework to promote practical application.

Procedure:

The approach was to couple two technologies used successfully elsewhere to address different aspects of the current research requirement, develop an intelligent tutoring system for battle command reasoning referred to here as "Automated Tutoring Environment for Command" (ATEC). The ATEC ITS adapted the dialog management capability from AutoTutor, a dialog-based tutor developed by Graesser and colleagues at the University of Memphis. It integrated this dialog management technology with a cognitive model-based instructional agent, a cognitive agent framework called iGEN by CHI Systems. The agent framework attempted to replicate the knowledge and role of the human mentor for such tactical instructional programs as "Think Like A Commander" (TLAC).

The procedure for developing ATEC ITS included the following elements:

- Developing a pedagogical approach, functional architecture, and software architecture for integrating AutoTutor dialog management capabilities into the initial iGEN-based ATEC system.
- Conducting analyses on mentoring discussions in the context of TLAC vignettes and developing the questions and expected answers for the ATEC mentor model.
- Evolving the initial user interface into a multi-user web environment that has a complete and self-contained application (with instructions and background materials).
- Developing and integrating dialog management components into the ATEC system.
- Enhancing iGEN to include capabilities for modeling tutorial dialog.
- Developing a complete virtual mentor model.
- Developing a performance assessment approach and incorporating it into the mentor model.
- Developing tools to facilitate testing and refinement of the dialog mechanisms.

Findings:

The findings are based on the approach used to develop and refine a prototype ATEC system using one tactical vignette from the TLAC training program. The ATEC program was developed as a web-based application that users can log onto from any computer with an Internet connection and a browser (with Flash and Java). Introductory material was included as well as links to relevant documents, making it a self-contained application. The iGEN technology served as the reasoning engine and core computational architecture for ATEC. It handled the domain knowledge and reasoning facilities associated with the vignette, the student model, and performance assessment components. The language processor (including syntactic parser and speech act classifier) and statistically grounded conceptual comparison components were derivatives of the AutoTutor system. The curriculum script and dialog management processes of AutoTutor were integrated into the iGEN mentor model.

The ATEC system was designed to present a battlefield situation and then initiate a dialog between a virtual mentor (instructional agent) and a student in a collaborative discussion of the tactical situation. As designed, the virtual mentor poses questions, evaluates student responses, determines the sequence of questions, and ultimately assesses performance on the basis of the specificity of questioning and the depth of probing and hinting that is needed to adequately answer the questions. The dialog is organized around the eight themes in TLAC. For each theme, there is a general question meant to start discussion of that aspect of the problem. Associated with each general question, there are anticipated good answers (called expectations) based on reasonable approaches to the problem posed. The virtual mentor assesses the student's response in relation to the possible good answers using a statistical comparison algorithm. There is also a set of progressively more specific questions for the virtual mentor to ask to prompt the student into thinking about any aspect of the theme that is not discussed in response to the initial question.

A principal finding is that severe technical challenges remain in developing a conversation-based tutoring system to assist military personnel in acquiring and practicing flexible tactical reasoning strategies in realistic battle situations. The goal of using open-ended, non-leading questions to stimulate broad consideration of all relevant aspects of a vignette made it difficult to evaluate student inputs accurately, leading to unnaturalness in the tutorial dialog. Additional research is therefore warranted to improve the evaluation algorithm and dialog mechanisms. Furthermore, additional effort is needed to make the web implementation of the system more robust and efficient.

Utilization of Findings:

Overall, the results of the ATEC development effort underscore areas requiring additional research and development in tutorial dialog systems to fully meet the research requirement, an intelligent tutoring system (ITS) for higher-order thinking skills such as battle command reasoning.

From the outset of this innovative research effort, there was uncertainty as to the feasibility of building a natural language dialog system for developing thinking skills. Computer-based natural language dialog systems are feasible for some classes of tutoring environments, namely those in which domain knowledge is qualitative and the shared knowledge (common ground) between the tutor and learner is low to moderate rather than high. Some aspects of tactical thinking require high precision and that student and tutor begin with at least a moderate amount of shared knowledge about the situation; thus ATEC was a borderline candidate for tutorial dialog, and the dialog was not always appropriate to the situation. The most promising applications of tutorial dialog systems are to conceptual domains in which the goal is to impart knowledge.

In addition, incremental changes were identified that could potentially improve ATEC. These include: changing or improving the tutor's conceptual pattern matching algorithm, refining the dialog management strategies and question hierarchy, and re-implementing the system for efficiency as a web application. However, it is an open question whether these changes would be sufficient for the type of tutoring problem addressed.

In sum, the value of the ATEC development effort is twofold. Lessons learned on technical challenges and changes required should be useful in future efforts on higher-order thinking skills, such as battle command reasoning. Technologies developed, including refinements to the tutoring architecture and underlying pedagogical approach, should readily apply to other training problems more amenable to conversational dialog.

A DIALOG-BASED INTELLIGENT TUTORING SYSTEM FOR PRACTICING BATTLE COMMAND REASONING

CONTENTS

Page

INTRODUCTION 1

Background 1
Phase I ATEC Approach 2
Phase II Research Objectives 3
Overview of this Report 4

COMPONENT TECHNOLOGIES 4

Dialog-based Intelligent Tutoring Systems 4
AutoTutor 8
Instructional Agents in Intelligent Tutoring Systems 11
Integration of Component Technologies 12

ATEC FUNCTIONAL DESCRIPTION AND SYSTEM ARCHITECTURE 13

Development Approach 13
Pedagogical Approach 13
User View 14
Functional Architecture 16
Virtual Mentor 18
Dialog Management and Curriculum Script 19
Language Processing Components 21
Performance Assessment 25
Software Architecture 30
Summary of ATEC Description 31

ANALYSES CONDUCTED 31

Initial Data Collection 31
Domain/Vignette Analyses 33
Analysis of Statistical Methods for Input Evaluation 33
Improvement of Statistical Methods 36
Semantic Reasoning 38

iGEN ENHANCEMENTS FOR DIALOG AND PERFORMANCE ASSESSMENT 42

Page

AUTHORING/TESTING TOOLS 43

iGEN Authoring 43
Logging Tools 43
Separate Text Only Interface for Evaluating Without Full System 44
Testing Parsers 45
FrameNet Evaluation Tool 45

LESSONS LEARNED 45

Four Quadrant Framework 45
Limitations of Open-ended Questions 47
Intelligent Conceptual Pattern Matching 48
Improvement of IDF as Pattern Matching Algorithm 48
Improvement of ATEC as a Web Application 49
Conclusions 49

REFERENCES 51

APPENDIX A Acronyms A-1

APPENDIX B Example Annotated Log B-1

LIST OF TABLES

Table 1. Four Quadrant Analysis 46

LIST OF FIGURES

Figure 1. Menu Screen 14
Figure 2. Vignette Interaction Screen 15
Figure 3. Supplementary Information Display 16
Figure 4. Functional Architecture 17
Figure 5. Example Extract from Curriculum Script 22
Figure 6. Example Session Dialog Extract 23
Figure 7. Four Examples of Performance Assessment Calculations 29
Figure 8. Screen Layout for Performance Assessment Summary 30
Figure 9. Software Architecture 31

A DIALOG-BASED INTELLIGENT TUTORING SYSTEM FOR PRACTICING BATTLE COMMAND REASONING

Introduction

The goal of this Phase II Small Business Innovative Research (SBIR[1]) effort was to develop an Intelligent Tutoring System (ITS) for interactive self-training of thinking skills, such as battle command reasoning, within a deliberate practice framework. In an attempt to achieve this goal, a dialog-based intelligent tutoring system was developed called "Automated Tutoring Environment for Command" (ATEC). This system involves the use of a dialog management capability based on the AutoTutor system, coupled with an iGEN-based instructional agent that replicates the knowledge and role of a human tutor, and a web-based personalized interface that manages the interaction between instructional agent and student.

The ATEC operates by first presenting a battlefield situation in a brief video on the ATEC interface. The system then initiates a text-based dialog between a virtual mentor (instructional agent) and a student as they collaboratively discuss the situation. The virtual mentor (a) poses questions, (b) evaluates student responses, (c) determines the sequence of questions, and (d) ultimately assesses performance on the basis of the specificity of questioning and the depth of probing and hinting that is needed to encourage the learner to adequately answer the questions. The system includes various natural language processing capabilities, including information extraction and dialog management.

Background

Training Needs in Battle Command Reasoning. The Army Research Institute for the Behavioral and Social Sciences (ARI) has developed training and instructional materials in a program called "Think Like A Commander" (TLAC). The TLAC program coaches command reasoning through adaptive thinking exercises using battlefield situations (Ross & Lussier, 1999; Lussier, Ross, & Mayes, 2000). The TLAC program deals with battlefield thinking habits that are characteristic of expert tactical thinkers, but are often absent during realistic tactical problem solving of less experienced commanders even though they are understood conceptually at a theoretical level. Schoolhouse learning involves primarily declarative knowledge about command principles and tactics, with training in task-specific procedures (in declarative form) based on these principles and tactics. Full-scale exercises and real command situations require integrated and ingrained expertise to include: determining what facts and principles are applicable to the problem, retrieving them, mapping the situation to the appropriate parts of the principles, and drawing inferences about the problem situation and its solution (e.g., VanLehn, 1996; Zachary & Ryder, 1997).

In essence, an intelligent knowledgeable coach is needed who can guide the user in applying principles and tactics to real-world problems. At first this process is slow and effortful, and the principles are applied one at a time (e.g., VanLehn, 1996; Zachary & Ryder, 1997). However, real problem situations require coordinated application of multiple facts and principles. Repeated real-time practice allows for "proceduralization" and chunking of skills

---

[1] Acronyms are defined in Appendix A.

(e.g., deriving domain-specific problem-solving strategies, integrating separate pieces of declarative knowledge) and development of automaticity of component skills (see Fisk & Rogers, 1992). It takes about 10 years to develop truly expert levels of performance and understanding (Ericsson, Krampe & Tesch-Romer, 1993), such as Klein's expert level of recognition-primed decision-making (Klein, 1989). This sophisticated expertise allows the appropriate quick interpretation and course of action to be derived directly (and almost instantaneously) from a recognition of key problem-instance features. As described below, TLAC addresses the transition from schoolhouse learning to adaptive expertise by providing deliberate practice opportunities.

Think Like A Commander Program. The TLAC program has been used with Brigade Command designees attending the School for Command Preparation of the Command and General Staff College (CGSC) at Fort Leavenworth, KS (U.S. Army Research Institute, 2001). It is also currently used in the Armor Captain's Career Course at Fort Knox, KY (Shadrick & Lussier, 2002; Lussier, Shadrick & Prevou, 2003). In its current form, TLAC presents tactical situations (called vignettes) as short movies in a classroom setting. Following the presentation of a vignette, there is a classroom discussion of the vignette led by an instructor acting as tutor or mentor. The instructor begins by asking general questions to stimulate thinking, and then asks increasingly more directed questions to probe for themes that have not been addressed. The discussion is organized around eight themes that underlie common patterns of expert tactical thinking:

1. Keep focus on mission and higher commander's intent.
2. Model a thinking enemy.
3. Consider effects of terrain.
4. Use all assets available.
5. Consider timing.
6. See the bigger picture.
7. Visualize the battlefield.
8. Consider the contingencies and remain flexible.

A set of questions or considerations are distinctly tailored to each theme. A number of TLAC vignettes have been developed and used across a mix of tactical situations.

Phase I ATEC Approach

In Phase I of this research, we proposed to develop an interactive practice environment using instructional agent technology (using CHI Systems' iGEN cognitive agent framework), an approach we had used successfully in previous research. Our subsequent assessment indicated that our original concept of an action-based interactive practice environment would not meet the requirements for interactive self-training of thinking skills in the manner that would accommodate the TLAC program. Instead, we determined that incorporation of a dialog management capability into the ATEC concept would provide the capabilities required to achieve the functionality needed. Moreover, a dialog management facility appeared to be technically feasible at this point in research and development, although it did push the state-of-the-art. We developed a revised architecture and operational concept that incorporated natural language processing and tutorial dialog.

The ATEC, like the TLAC training, begins with the viewing of a vignette. After the vignette had been viewed, the instructional agent would conduct a dialog probing the student understanding of the situation and approach to handling it. A conceptual prototype was developed that demonstrated the planned Phase II architecture. The conceptual prototype incorporated an initial version of an instructional agent that focused on one theme from one vignette, an initial version of the user interface, and a simple placeholder for the dialog management capability envisioned for Phase II development.

The instructional agent maintained a hierarchical list of questions that should be asked to evaluate what knowledge the student had demonstrated. Parsed student responses were analyzed to evaluate each specific response and to update a tree-like representation of student performance, which was maintained as a student model. The student model matched the structure of the question tree and allowed the instructional agent to monitor the student's responses to each particular question by populating the branch of the tree that specifically correlated with the stated question. The instructional agent also maintained a record of which questions were asked and which questions elicited the matching concepts to use in evaluation. In conducting the evaluation, the agent compared how far down the tree, or how specific and leading, the questions had to be asked before the student demonstrated satisfactory understanding of the relevant concepts. Because the dialog management capability was an addition to the original Phase I plan, the conceptual prototype implemented a very simple keyword-spotting algorithm as a placeholder for a full dialog management system planned for Phase II. An initial version of the user interface subsystem was developed in Phase I. The interface featured a display map panel where the vignette is displayed, control buttons to play the vignette (and/or re-play, zoom-in, zoom-out, stop), a 'talking head' box where narrator and mentor images appeared at appropriate times, dialog boxes for mentor output and student input, and buttons that allowed the student to link to supplementary materials.

Phase II Research Objectives

Building on the Phase I work, there were four research objectives for Phase II, as follows:

1. *Develop the dialog/tutoring management system* for the ATEC system. This was the key objective for Phase II since the decision to incorporate a dialog-based approach was made at the end of Phase I. While all other components of the ATEC architecture were at least partially implemented in Phase I, the dialog management approach was only approached in a 'placeholder' manner, using a minimal keyword-spotting algorithm. In the Phase II approach, the development of a suitable natural language-based dialog processor and manager constituted a major portion of the effort. This objective involved adaptation and integration of an existing and proven technology into the ATEC system called iGEN, a cognitive agent software toolkit developed by CHI systems. The additional technology was the AutoTutor system developed by Graesser and colleagues at the University of Memphis.

2. *Develop domain analysis tools to support semi-automated vignette authoring and analysis.* An initial vignette was to be developed by hand and used to test and develop the dialog/tutoring management system. Subsequent to the initial vignette development, authoring tools would be developed to facilitate development of additional vignettes.
3. *Implement the instructional management subsystem*, based on the instructional model developed in Phase I. The instructional agent approach used in Phase I needed to be fleshed out and integrated with the dialog management system.
4. *Develop a system to measure the students' ATEC performance.* The rough concept for evaluating performance from Phase I had to be developed into a full capability in conjunction with the instructional management system.

Overview of this Report

The next section of this report discusses the component technologies used in this effort, followed by a description of the ATEC system as it was developed. Subsequent sections describe the analyses conducted to support development decisions, the enhancements made to the iGEN agent development system to support integration of dialog management capabilities with instructional management, and the authoring and testing tools created to support system development. A final section provides an assessment of ATEC, lessons learned from the development effort, and an assessment of the state-of-the-art of natural language intelligent tutoring systems.

Component Technologies

Dialog-based Intelligent Tutoring Systems

The vision of having a computer communicate with users in natural language was entertained shortly after the computer was invented, but it was not until Weizenbaum's (1966) ELIZA program that a reasonably successful conversation system could be explored. Subsequent efforts at dialog-based tutors include:

1. Collin's tutoring system on South American geography called SCHOLAR (Collins, Warnock, & Passafiume, 1975).
2. Woods' program that syntactically parsed questions and answered user's queries about moonrocks (Woods, 1977).
3. Work by Schank and his colleagues in building computer models of natural language understanding and rudimentary dialog about scripted activities (Lehnert & Ringle, 1982; Schank, 1986; Schank & Reisbeck, 1982).
4. Winograd's SHRDLU system that interacted with a user on manipulating simple objects in a blocks world (Winograd, 1972).
5. A speech recognition system that handles airline reservations, called Hear What I Mean, (HWIM) (Cohen, Perrault, & Allen, 1982).

Unfortunately, two decades of exploring human-computer dialog systems had a less than encouraging outcome. By the mid-1980's, most researchers in artificial intelligence were convinced that the prospects of building good conversation systems was well beyond the horizon. This belief was based upon the following: (a) inherent complexities of natural language processing, (b) the unconstrained, open-ended nature of world knowledge, (c) the lack of research on lengthy threads of connected discourse, and (d) the time and expertise constraints in building student models.

The early pessimism about natural language processing and conversational dialog systems was arguably premature. Because of a sufficient number of technical advances in the last eight years, researchers are revisiting the vision of building such dialog systems. The field of computational linguistics has recently produced an impressive array of lexicons, syntactic parsers, semantic interpretation modules, and dialog analyzers that are capable of rapidly extracting information from naturalistic text and discourse (Allen, 1995; DARPA, 1995; Harabagiu, Maiorano, & Pasca, 2002; Jurafsky & Martin, 2000; Manning & Schutze, 1999; Voorhees, 2001). Lenat's CYC system represents a large volume of mundane world knowledge in symbolic forms that can be integrated with a diverse set of processing architectures (Lenat, 1995).

The world knowledge contained in an encyclopedia can be represented statistically in high dimensional spaces, such as Latent Semantic Analyses (LSA) (Foltz, Gilliam, & Kendall, 2000; Landauer, Foltz, & Laham, 1998). An LSA space (which can be considered a kind of student model) can be created overnight, a space that produces semantic judgments on whether two text excerpts are conceptually similar. The representation and processing of connected discourse is much less mysterious after two decades of research in discourse processing (Graesser, Gernsbacher, & Goldman, 2003). There are now generic computational modules for building dialog facilities that attempt to track and manage the beliefs, knowledge, intentions, goals, and attentional states of agents in two party dialogs (Core, Moore, & Zinn, 2000; Gratch et al., 2002; Moore & Wiemer-Hastings, 2003; Pellom, Ward, & Pradhan, 2000; Rich & Sidner, 1998; Rickel, Lesh, Rich, Sidner, & Gertner, 2002; Graesser, VanLehn, Rose, Jordan, & Harter, 2001).

Computer-based natural language dialog is particularly feasible in some classes of tutoring environments. First, the feasibility of tutorial dialog in natural language depends on the subject matter, the knowledge of the learner, and the sophistication of tutoring strategies. It is sometimes more feasible when the knowledge domain is qualitative (e.g., verbal reasoning, open-ended qualitative knowledge) rather than precise (e.g., mathematics, logic). Although precise domain tutors have been developed (e.g., Heffernan & Koedinger, 1998), the large number of computational linguistic modules available, as well as the pedagogical opportunities in natural language dialog, make qualitative domains often preferred (e.g., Graesser, VanLehn, et al., 2001). But the choice of qualitative versus precise depends on the domain. Natural language dialog systems would not be well suited to an eCommerce application that manages precise budgets, but are surprisingly good in coaching students on topics that involve verbal reasoning.

Second, tutorial dialog in natural language is feasible when the shared knowledge (common ground) between the tutor and learner is low to moderate rather than high. If the common ground is high, then both speech participants (i.e., the computer and the learner) will be expecting a higher level of precision of mutual understanding and therefore will have a higher risk of failing to meet each other's expectations. In contrast, it is entirely reasonable to build a natural language dialog system when the computer and tutor do not track what each other knows at a fine-grained level and when the computer produces dialog moves (e.g., questions, hints, assertions, short responses) that advance the dialog to achieve the learning goals.

It is noteworthy that human tutors are not able to monitor the knowledge of students at a fine-grained level because much of what students express is vague, underspecified, ambiguous, fragmentary, and error-ridden (Fox, 1993; Shah, Evens, Michael, & Rovick, 2002; Graesser & Person, 1994; Graesser, Person, & Magliano, 1995). It ordinarily would not be worthwhile to dissect and correct each of these deficits because it is more worthwhile to help build new correct knowledge (Sweller & Chandler, 1994). Tutors do have an approximate sense of what a student knows and they do provide productive dialog moves that lead to significant learning gains in the student (Chi, Siler, Jeong, Yamauchi, & Hausmann, 2001; Cohen, Kulik, & Kulik, 1982; Graesser et al., 1995). These considerations indeed motivated the design of AutoTutor (Graesser, Person, Harter, & Tutoring Research Group (TRG), 2001; Graesser, VanLehn, Rose, Jordan & Harter, 2001; Graesser, Wiemer-Hastings, Wiemer-Hastings, Kreuz, & TRG, 1999) as well as the current ATEC system. In essence, dialog can be useful when it advances the dialog and learning agenda, even when the tutor does not fully understand a student. To use an analogous dialog situation, a native speaker of English can often express utterances that help a visitor from another country (with broken English), even though the visitor is only approximately understood.

Third, tutorial dialog in natural language is feasible when the tutoring strategies follow what most human tutors do rather than the strategies that are highly sophisticated. Most human tutors anticipate particular correct answers (called expectations) and misconceptions when they ask the learner's questions and trace the learner's reasoning. As the learner articulates the answer or solves the problem, this content is constantly being compared with the expectations and misconceptions; the tutor responds adaptively and appropriately when each expectation or misconception is expressed. We refer to this tutoring mechanism as *expectation and misconception tailored (EMT) dialog* (Graesser, Hu, & McNamara, in preparation). The EMT dialog moves of most human tutors are not particularly sophisticated from the standpoint of ideal tutoring strategies that have been proposed in the fields of education and artificial intelligence (Graesser et al., 1995).

Graesser and colleagues (Graesser & Person, 1994; Graesser et al., 1995) videotaped over 100 hours of naturalistic tutoring, transcribed the data, classified the speech act utterances into discourse categories, and analyzed the rate of particular discourse patterns. These analyses revealed that human tutors rarely implement intelligent pedagogical techniques such as *bona fide* Socratic tutoring strategies, modeling-scaffolding-fading, reciprocal teaching, frontier learning, building on prerequisites, cascade learning, or diagnosis/remediation of deep misconceptions (Collins, Brown, & Newman, 1989; Palincsar & Brown, 1984; Sleeman & Brown, 1982). Instead, tutors tend to coach students in constructing explanations according to the EMT dialog patterns. Fortunately, the EMT dialog strategy is substantially easier to implement computationally than are the sophisticated tutoring strategies.

During the last decade, researchers have developed a half dozen intelligent tutoring systems with dialog in natural language. Four of these are listed below.

1. *AutoTutor* and *Why/AutoTutor* (Graesser, Hu & McNamara, in preparation; Graesser, Person et al., 2001; Graesser, Wiemer-Hastings et al., 1999). This system will be described in the next section. It has been developed for introductory computer literacy and Newtonian physics. These systems scaffold college students on applying higher order cognitive strategies, explanations, and knowledge-based reasoning to particular problems.
2. *Why/Atlas* (VanLehn et al., 2002). Students learn about conceptual physics by a coach that helps build explanations of conceptual physics problems. It has modules with syntactic parsers, a lexicon, semantic interpreters, symbolic reasoning modules, and finite state machines to manage the dialog (called *knowledge construction dialogs*). It also uses Bayesian networks, latent semantic analysis, and other statistical techniques in modules that perform pattern recognition and comparison operations. Why/Atlas has been tested in one study and has produced learning gains approximately the same as Why/AutoTutor and as computer-mediated communication with expert physicists with extensive experience in pedagogy serving as tutors.
3. *Circsim Tutor* (Freedman, 1999; Hume, Michael, Rovick, & Evens, 1996; Shah, Evens, Michael, & Rovick, 2002). Medical students learn about the circulation system by interacting in natural language. The computer tutor attempts to implement strategies of an accomplished tutor with a medical degree. The system has a spelling checker, a lexicon, a syntactic parser, rudimentary semantic analyzers, and a dialog planner. There have been informal evaluations of learning gains, but no formal evaluation.
4. *Pedagogical Agent for Collogen (PACO)* (Rickel, Lesh, Rich, Sidner, & Gertner, 2002). The PACO assists learners in interacting with mechanical equipment and completing tasks by interacting in natural language. The PACO integrates Collagen, the generic dialog planning system developed by Rich and Sidner (1998), with an existing intelligent tutoring system called Virtual Interactive ITS Development Shell (VIVIDS). There have been no evaluations of PACO on learning gains.

From the above four systems that have been evaluated, two noteworthy generalizations can be made. The first generalization applies to dialog management. Finite state machines for dialog management have provided an architecture that can be applied to produce working systems (as in AutoTutor, Why/AutoTutor, and Why/Atlas). In contrast, there have been no full-fledged dialog planners in working systems that perform well enough to be evaluated (as in Circsim Tutor and PACO). Dialog planning is extremely difficult because it requires the precise recognition of knowledge states (goals, intentions, beliefs, knowledge) and a closed system of formal reasoning. Unfortunately, dialog contributions of learners are often too vague and underspecified to afford precise recognition of knowledge states. The second generalization addresses the representation of world knowledge. The LSA-based statistical representation of world knowledge allows the researcher to very quickly (measured in hours or days) have some world knowledge component up and running, whereas the symbolic representation of world knowledge takes years or decades to develop. AutoTutor and Why/AutoTutor routinely incorporates LSA in its knowledge representation so it is a tutoring system in which a new subject matter can be quickly developed.

AutoTutor

AutoTutor is a dialog-based tutor developed by Graesser and colleagues at the University of Memphis (Graesser, et al., in preparation; Graesser, Person et al., 2001; Graesser, Wiemer-Hastings et al., 1999). AutoTutor asks questions or presents problems that require approximately a paragraph of information (e.g., 3-7 sentences, or 50-100 words) to produce an ideal answer. Of course, it is possible to accommodate questions with answers that are longer or shorter; the paragraph span is simply the length of answers that have been implemented in AutoTutor so far, in an attempt to handle open-ended questions that invite qualitative reasoning in the answer. Although an ideal answer is approximately 3-7 sentences in length, the initial answers to these questions by learners are typically only 1-2 sentences in length. This is where tutorial dialog is particularly helpful. AutoTutor engages the learner in a mixed initiative dialog that assists the learner in the evolution of an improved answer that draws out more of the learner's knowledge that is relevant to the answer. The dialog between AutoTutor and the learner typically lasts 30-100 *turns* (i.e., the learner expresses something, then the tutor, then the learner, and so on).

There is an important reason for using sentences as the basic metric in measuring content in AutoTutor. One of the goals is to have subject matter experts create the content of question-answer items in the *curriculum script*. The experts simply type in the question in English, followed by sentences on the ideal answer, and other specifications of content. Most subject matter experts are not accomplished experts in artificial intelligence or cognitive engineering so it is unrealistic to have them compose structured code. Sentences are a familiar unit of analysis and are reasonably self-contained packages of information.

AutoTutor produces several categories of *dialog moves* that facilitate covering information that is anticipated by AutoTutor's *curriculum script*. AutoTutor delivers its dialog moves via an animated conversational *agent* (synthesized speech, facial expressions, gestures), whereas learners enter their answers via keyboard. AutoTutor provides *feedback* to the learner (positive, neutral, negative feedback), *pumps* the learner for more information ("What else"), *prompts* the learner to fill in missing words, gives *hints*, fills in missing information with *assertions*, identifies and *corrects* bad answers, *answers* learners' questions, and *summarizes* answers. As the learner expresses information over many turns, the information in the 3-7 sentences is eventually covered and the question is answered. During the process of supplying the ideal answer, the learner periodically articulates misconceptions and false assertions. If these misconceptions have been anticipated in advance and incorporated into the program, AutoTutor provides the learner with information to correct the misconceptions. Therefore, as the learner expresses information over the turns, this information is compared to expectations and misconceptions, and AutoTutor formulates its dialog moves in a fashion that is sensitive to the learner input. That is, AutoTutor implements *expectation and misconception tailored dialog* (EMT dialog), which is known to be common in human tutors. The design of AutoTutor was also inspired by:

1. The explanation-based constructivist theories of learning (Chi, deLeeuw, Chiu, LaVancher, 1994; VanLehn, Jones, & Chi, 1992). Learning is deeper when the learner must actively generate explanations, justifications, and functional procedures than when merely given information to read.
2. Anderson's cognitive tutors that adaptively respond to learner knowledge (Anderson, Corbett, Koedinger, & Pelletier, 1995). The tutors give immediate feedback to learner's

actions and guide the learner on what to do next in a fashion that is sensitive to what the system believes the learner knows.

3. Previous empirical research that has documented the collaborative constructive activities that routinely occur during human tutoring (Chi, Siler, Jeong, Yamauchi, & Hausmann, 2001; Fox, 1993; Graesser & Person, 1994; Graesser et al., 1995). After these researchers analyzed videotaped or audiotaped tutoring sessions in detail, they discovered patterns of dialog that frequently occur and compared the incidence of these patterns to theoretical claims from pedagogical frameworks.

AutoTutor uses LSA for its conceptual pattern matching algorithm when evaluating whether student input matches the expectations and misconceptions. The LSA is a high-dimensional, statistical technique that, among other things, measures the conceptual similarity of any two pieces of text, such as a word, sentence, paragraph, or lengthier document (Foltz, Gilliam, & Kendall, 2000; Kintsch, Steinhart, Stahl & LSA Research Group, 2000; Kintsch, 1998, 2001; Landauer & Dumais, 1997; Landauer, Foltz, & Laham, 1998). A cosine between the LSA vector associated with expectation E (or misconception M) and the vector associated with learner input (I) is calculated. An E or M is scored as covered if the match between E or M and the learner's text input I meets some threshold, which has varied between .40 and .65 in previous instantiations of AutoTutor.

Suppose that there are four key expectations embedded within an ideal answer. AutoTutor expects these answers to be covered in a complete answer and will direct the dialog in a fashion that finesses the students to articulate these expectations (through prompts and hints). AutoTutor stays on topic by completing the sub-dialog that covers expectation E before starting a sub-dialog on another expectation. For example, suppose an expectation (*The earth exerts a gravitational force on the sun*) needs to be articulated within the answer. The following family of prompts is available to encourage the student to articulate particular content words in the expectation:

1. The gravitational force of the earth is exerted on the ____.
2. The sun has exerted on it the gravitational force of the ____.
3. What force is exerted between the sun and earth? ____.
4. The earth exerts on the sun a gravitational ____.

AutoTutor first considers everything the student expresses during conversation turns 1 through N to evaluate whether expectation E is covered. If the student has failed to articulate one of the four content words (*sun, earth, gravitational, force*), AutoTutor selects the corresponding prompt (1, 2, 3, and 4, respectively). One obvious alternative might be to simply have AutoTutor assert the missing information, but that would be incompatible with the pedagogical goal of encouraging the learner to actively construct knowledge, as discussed earlier. If the student has made three assertions at a particular point in the dialog, then all possible combinations of assertions X, Y, and Z would be considered in the matches [i.e., cosine (vector E, vector I)]: X, Y, Z, XY, XZ, YZ, XYZ. The maximum cosine match score is used to assess whether expectation E is covered. If the match meets or exceeds threshold T, then expectation E is covered. If the match is less than T, then AutoTutor selects the prompt (or hint) that has the best chance of improving the match (that is, if the learner provides the correct answer to the prompt). Only explicit statements by the learner are considered when determining whether expectations are covered. As such, this approach is compatible with constructivist learning theories that emphasize the importance of the learner generating the answer.

The conversation is finished for the question when all expectations are covered. In the meantime, if the student articulates information that matches any misconception, the misconception is corrected as a sub-dialog and then the conversation returns to finishing coverage of the expectations. Again, the process of covering all expectations and correcting misconceptions that arise normally requires a dialog of 30-100 turns (or 15-50 student turns).

The conversational interactions between AutoTutor and the student are lengthy because of the pedagogical goal, expressed above, of getting the student to construct the explanation, as opposed to merely having AutoTutor be an information delivery system. The pedagogical goals could be entirely different in some learning environments. For example, an alternative pedagogical goal would be to efficiently cover the material by minimizing interaction time and number of turns. That could be easily implemented in AutoTutor by simply turning off the prompt and hint dialog moves in the curriculum script and having AutoTutor delivering assertions that fill in missing pieces of information. At the extreme, AutoTutor would simply present the question-answer item and not solicit information from the student at all. However, such a system would be a standard computer-based training system rather than an intelligent tutoring system that adapts to the student's performance. The design of the curriculum script was sufficiently general to accommodate a diversity of pedagogical goals and conversational styles that a designer wished to implement.

In addition to asking questions, AutoTutor attempts to handle questions posed by the learner. However, somewhat surprisingly, students rarely ask questions in classrooms, human tutoring sessions, or AutoTutor sessions (Graesser & Person, 1994; Graesser & Olde, 2003). The rate of learner questions is 1 learner question per 6-7 hours in a classroom environment and 1 per 10 minutes in tutoring. Although it is pedagogically disappointing that learners ask so few questions, the good news is that this aspect of human tutor interaction makes it easier to build a dialog-based intelligent tutoring system such as AutoTutor. It is computationally straightforward to compare learner input with computer expectations through pattern matching operations. It is extremely difficult, if not impossible, to interpret any arbitrary learner question from scratch and to construct a mental space that adequately captures what the learner has in mind. These claims are widely acknowledged in the computational linguistics and natural language processing communities. Therefore, what human tutors and learners do is compatible with what currently can be handled computationally within AutoTutor.

A goal of the research was to fine-tune the LSA-based pattern matches between learner input and AutoTutor's expected input (see Hu, Cai, Graesser et al., 2003; Hu, Cai, Franceschetti et al., 2003; Olde, Franceschetti, Karnavat, Graesser & TRG, 2002). The good news is that LSA does a moderately impressive job of determining whether the information in learner essays match particular expectations associated with an ideal answer. For example, in one recent study, experts in physics or computer literacy were asked to make judgments concerning whether particular expectations were covered within learner essays. A coverage score was computed as the proportion of expectations in the learner essay that judges believed were covered, using either stringent or lenient criteria. Similarly, LSA was used to compute the proportion of expectations covered, using varying thresholds of cosine values on whether information in the learner essay matched each expectation. Correlations between the LSA scores and the judges' coverage scores were approximately .50 for both conceptual physics (Olde, Franceschetti, Karnavat, Graesser, & TRG, 2002) and computer literacy (Graesser, Wiemer-Hastings et al., 2000). Correlations generally increase as the length of the text increases, yielding correlations as high as .73 (Foltz et al., 2000). The LSA metrics also did a reasonable job tracking the coverage of expectations and the identification of misconceptions during the course of AutoTutor's tutorial dialogs.

The question arises whether AutoTutor is successful in promoting learning gains. Previous versions of AutoTutor have produced gains of .4 to 1.5 sigma depending on the learning performance measure, the comparison condition (either pretest scores or a control condition in which the learner reads the textbook for an equivalent amount of time as the tutoring session), the subject matter, and the version of AutoTutor (Graesser, Jackson, Mathews, et al., 2003; Graesser, Moreno, et al., 2003; Person, Graesser, Bautista, Mathews, & TRG, 2001). These results place previous versions of AutoTutor somewhere between an unaccomplished human tutor of .4 sigma to an intelligent tutoring system of 1 sigma. Moreover, one recent evaluation of physics tutoring remarkably reported that the learning gains produced by accomplished human tutors in computer-mediated communication were equivalent to the gains produced by AutoTutor (Graesser, Jackson, Mathews, et al., 2003).

AutoTutor has many other components that are needed to manage a mixed initiative dialog with the learner. AutoTutor attempts to handle any input that the learner types in, whether it is grammatical or ungrammatical. This is possible in part because of the recent advances in computational linguistics that have provided lexicons, corpora, syntactic parsers, shallow semantic interpreters, and a repository of free automated modules. AutoTutor currently manages a surprisingly smooth conversation with the student, even though it does not deeply analyze the meaning of the student contributions, does not build a detailed common ground, and does not have an intelligent symbolic planner. The dialog facilities of AutoTutor have been tuned to the point where bystanders cannot accurately decide whether a particular dialog move was generated by AutoTutor or a human tutor (Person, Graesser, & TRG, 2002). The next steps in the AutoTutor enterprise include blending in deeper comprehension modules, dialog planners, and pedagogical strategies, and determining the extent to which these sophisticated components improve learning gains.

Instructional Agents in Intelligent Tutoring Systems

The canonical ITS architecture includes, at a minimum, the following three components: (a) an *expert module* that contains a representation of the knowledge to be presented and a standard for evaluating student performance, (b) a *student module* that represents the student's current understanding of the domain, and (c) an *instructional module* that contains pedagogical strategies and guides the presentation of instructional material (Polson & Richardson, 1988; Sleeman & Brown, 1982; Wenger, 1987). These three aspects of intelligence need not be separate components. Current thinking is that the key to intelligent training is designing the system to behave intelligently by providing adaptive instruction that is sensitive to an approximate diagnosis of the student's knowledge structures or skills (Shute & Psotka, 1995). The indeterminacy and complexity of many domains, including battlefield reasoning, preclude the use of model tracing approaches to student modeling, which are only applicable to procedural learning and reasoning in well-structured domains. Furthermore, recent pedagogical theories have focused on collaborative learning, situated learning, deliberate practice, constructive learning, and distributed interactive simulation, all of which call for modifications of the traditional ITS paradigm and the creation of alternative interactive learning environments.

A different approach has been to use cognitive modeling technology to create a model of an instructor that can be embedded in an interactive learning environment for the more complex, indeterminate domains. These models, called *instructional agents*, embody the reasoning of a human instructor and include all three aspects of tutoring intelligence in one model: domain knowledge, diagnostic reasoning, and pedagogical reasoning. The difficulty of diagnosing deficiencies in knowledge and skill or of selecting appropriate pedagogical strategies is not diminished using instructional agents.

However, the problem becomes more tractable when we analyze the expertise of an instructor using cognitive task analysis methods, and we create an executable model of the tutorial knowledge that is applicable in the instructional domain. Cognitive modeling may provide a more natural methodology for representing human expertise than other artificial intelligence (AI) formalisms.

The associated cognitive task analysis provides a richer method for acquiring that knowledge than other knowledge engineering techniques. CHI Systems has developed a cognitive agent technology called iGEN (Zachary, Ryder, Ross & Weiland, 1992; Zachary, Le Mentec, & Ryder, 1996) that can be used to create instructional agents. iGEN-based instructional agents have been used successfully in other complex domains that preclude the use of model tracing approaches to student modeling (Ryder, Santarelli, Scolaro, Hicinbothom, & Zachary, 2000; Zachary, Santarelli, Lyons, Bergondy, & Johnston, 2001).

Integration of Component Technologies

The iGEN technology serves as the reasoning engine and core computational architecture for ATEC, as described below. However, the instructional agent approach was modified for this application to incorporate the pedagogical approach of AutoTutor and to integrate its language processing mechanisms. Combining a system that models human thought and problem-solving and a system that excels in conversational tutoring seems ideal.

As pointed out earlier, language processing mechanisms are particularly useful in qualitative domains. Battle command reasoning can be considered qualitative rather than precise for various reasons. First, officers moving into command positions understand the fundamentals of command, but have difficulties with using all the principles across a range of situations. Secondly, instead of learning what to think, officers are taught how to think. They will have to apply fundamentals to command adaptively. Finally, one of the limitations of a commander under stress is cognitive tunneling: the inability to consider all aspects of the situation.

The next sections will describe how the various components have been implemented in the ATEC tutoring system.

ATEC Functional Description and System Architecture

Development Approach

The ATEC Phase II development began with the integration of AutoTutor with the ATEC interface and an initial iGEN instructional agent. There was a need for a software integration of the technologies as well as a conceptual integration. In addition, the integrated system needed to address the battlefield reasoning problem, which was somewhat different from those that AutoTutor had addressed.

The overall development approach began with adapted AutoTutor components, then migrated many of AutoTutor's functions into iGEN, and ended up with the final product having the controlling logic for the system within the iGEN instructional agent. As part of the integration, it was necessary to analyze AutoTutor functions and components and to determine what aspects to include and how to accomplish the integration. The rationale for these decisions is discussed throughout this section of the report. The integrated system is described in the subsection on Functional Architecture.

Pedagogical Approach

The ATEC system presents a battlefield situation and then initiates a dialog between a virtual mentor (instructional agent) and a student as they collaboratively discuss the situation. The virtual mentor poses questions, evaluates student responses, determines the sequence of questions, and ultimately assesses performance on the basis of the specificity of questioning and the depth of probing and hinting that is needed to adequately answer the questions.

Responses are not considered correct or incorrect, but rather starting points for a dialog about the important considerations in the vignette. In fact, there are multiple reasonable ways to approach the problem in any vignette, all leading to reasonable answers to a specific question. AutoTutor has been applied to computer literacy and conceptual physics, both of which are domains that require conceptual reasoning and that have one correct answer to each question or problem to solve. Thus, the pedagogical approach from AutoTutor had to be adapted.

The ATEC attempts to replicate the coaching and scaffolding that human instructors/mentors provide in the TLAC program. The ATEC is organized around the eight themes in TLAC. For each theme, there is a general question meant to start discussion of that aspect of the problem. Associated with each general question, there are anticipated good answers (called expectations) based on reasonable approaches to the problem posed. The virtual mentor assesses the student's response in relation to the possible good answers.

There is also a set of progressively more specific questions for the virtual mentor to ask to prompt the student into thinking about any aspect of the theme not discussed in response to the initial question. This approach is based on the AutoTutor curriculum script approach, but was modified to provide a mentoring style of dialog rather than the tutoring style previously used in AutoTutor for teaching computer literacy or physics. For example, ATEC did not include very specific pumps (e.g., "The mechanism in a computer that stores data between sessions is called the ...") as they are too leading for a mentoring dialog; or corrective splices that correct misconceptions in bad answers, as that would imply an incorrect answer had been given.

User View

The ATEC is a web-based application that users can log onto from any computer with an Internet connection and a browser (with Flash and Java). Upon entering the ATEC system, a menu screen (Figure 1) is presented allowing the user to access instructions, view the Road to War, choose a vignette for a training session, or end the training session.

The instructions provide an explanation of ATEC and TLAC and describe the process for using the system. The Road to War button brings up a Flash movie containing the background situating information for all the vignettes.

In order to focus on the design issues, we have used one vignette, Vignette 5, as our example. When a vignette is selected, the main vignette interaction screen appears (Figure 2). This is the main screen for viewing the vignettes, interacting with the tutor, and accessing supplementary materials. The components of the main vignette interaction screen are described below. The numbers correspond to the labels in Figure 2.

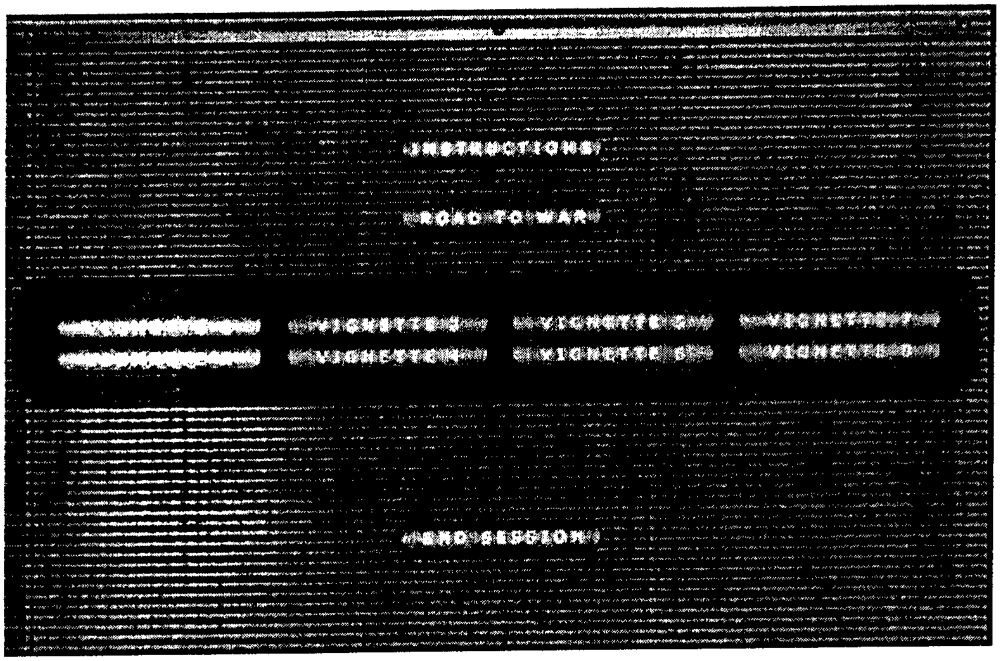


Figure 1. Menu screen.

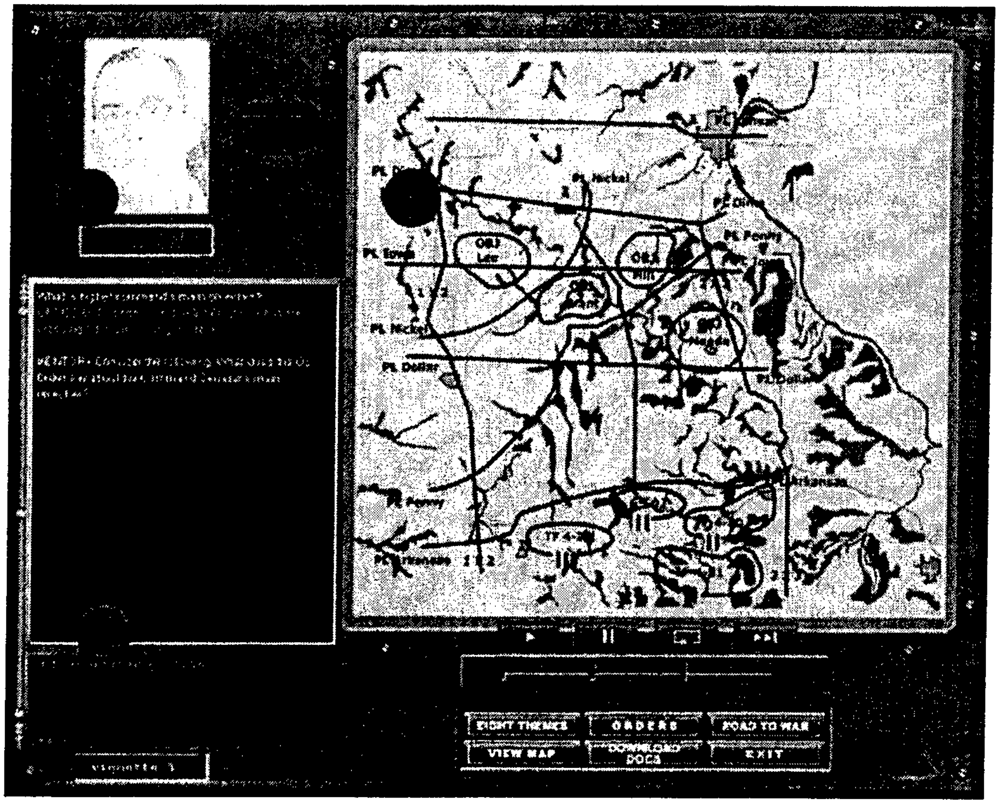


Figure 2. Vignette interaction screen.

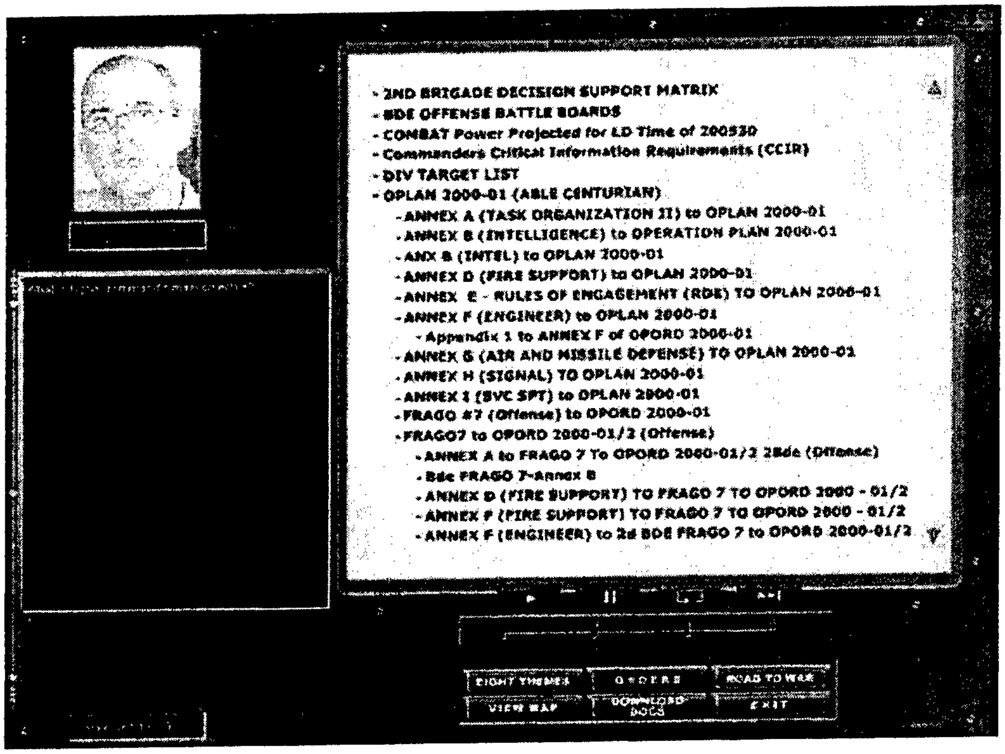
1. This is the speaker identification area, which displays a picture of the person speaking and his position.
2. This is the main information display area. Flash movies of the Road to War and Vignettes are displayed here. Supplementary information is also displayed in this area (see Figure 3). There are four VCR-like controls associated with this box to allow the user to control the presentation of the Road to War and Vignette: PLAY, PAUSE, RE-START, and END.

   - PLAY and PAUSE are used together to pause the presentation at any time and to resume playing it from where it was paused.
   - RE-START re-starts the presentation from the beginning.
   - END jumps to the last segment of the presentation.

3. This area is the virtual mentor interaction area. All dialog is conducted using this area. The top box contains the running dialog between the mentor and the user. Mentor dialog comes up here. The bottom box is for user input. Once the user is satisfied with his or her input and presses the ENTER key on the keyboard, the input becomes part of the running dialog in the top box.

4. This control area provides buttons for controlling the session and accessing supplementary information. There are six buttons: EIGHT THEMES, ORDERS, ROAD TO WAR, VIEW MAP, DOWNLOAD DOCS, and EXIT.

   - EIGHT THEMES brings up the eight themes that have been indicated by the Army Research Institute as representing necessary components of expert patterns of battlefield thinking.
   - ORDERS brings up a list of the relevant orders that can be viewed. Selection of an order causes the order to appear in a new window.
   - ROAD TO WAR allows the user to review the Road to War while responding to Mentor questions.
   - VIEW MAP returns the vignette map display after viewing any other information.
   - DOWNLOAD DOCS provides a list of documents (i.e., field manuals) available for downloading. Selection of a document causes it to appear in a new window.
   - EXIT exits the current vignette session and returns to the menu screen.



- 2ND BRIGADE DECISION SUPPORT MATRIX
- BDE OFFENSE BATTLE BOARDS
- COMBAT Power Projected for LD Time of 200530
- Commanders Critical Information Requirements (CCIR)
- DIV TARGET LIST
- OPLAN 2000-01 (ABLE CENTURIAN)
  - ANNEX A (TASK ORGANIZATION II) to OPLAN 2000-01
  - ANNEX B (INTELLIGENCE) to OPERATION PLAN 2000-01
  - ANX B (INTEL) to OPLAN 2000-01
  - ANNEX D (FIRE SUPPORT) to OPLAN 2000-01
  - ANNEX E - RULES OF ENGAGEMENT (ROE) TO OPLAN 2000-01
  - ANNEX F (ENGINEER) to OPLAN 2000-01
    - Appendix 1 to ANNEX F of OPORD 2000-01
  - ANNEX G (AIR AND MISSILE DEFENSE) TO OPLAN 2000-01
  - ANNEX H (SIGNAL) TO OPLAN 2000-01
  - ANNEX I (SVC SPT) to OPLAN 2000-01
  - FRAGO #7 (Offense) to OPORD 2000-01
  - FRAGO7 to OPORD 2000-01/2 (Offense)
    - ANNEX A to FRAGO 7 To OPORD 2000-01/2 2Bde (Offense)
    - Bde FRAGO 7-Annex B
    - ANNEX D (FIRE SUPPORT) TO FRAGO 7 TO OPORD 2000 - 01/2
    - ANNEX F (FIRE SUPPORT) TO FRAGO 7 TO OPORD 2000 - 01/2
    - ANNEX F (ENGINEER) to 2d BDE FRAGO 7 to OPORD 2000-01/2

Figure 3. Supplementary information display.

Functional Architecture

Figure 4 shows the functional architecture of the ATEC system, indicating which components are handled by AutoTutor components, iGEN components, or Flash/Java components. The user interface components are implemented as Flash and Java. The language processor (including syntactic parser and speech act classifier) and statistical comparison components are derivatives of the AutoTutor system, whereas iGEN handles the domain knowledge and reasoning facilities associated with each vignette, the student model, and performance assessment components. In addition, the curriculum script and dialog management processes of AutoTutor have been integrated into the iGEN virtual mentor.

The Language Processor parses the student input, classifies it into speech act categories, and passes a parsed and tagged representation of the student input to the Student Input Evaluation process within the Virtual Mentor. The Virtual Mentor (instructional agent): (a) controls and maintains the list of questions that should be asked (functionally a Curriculum Script), (b) evaluates what knowledge the student has demonstrated (Student Input Evaluation), and (c) maintains a representation of the student's discussion of vignette aspects (Student Model). Frozen expressions (e.g., "I don't understand," "Could you repeat that?") are handled directly by the Dialog Management processes, while contributions and questions are handled by the Student Input Evaluation processes. The system is designed to incorporate a hybrid approach for evaluating student inputs. One type of evaluation uses statistical techniques (Statistical Comparison) for comparing the student input to the expected good answer(s). The second method for evaluating student input involves deep reasoning based on domain knowledge. ATEC does not currently have any deep reasoning logic implemented.

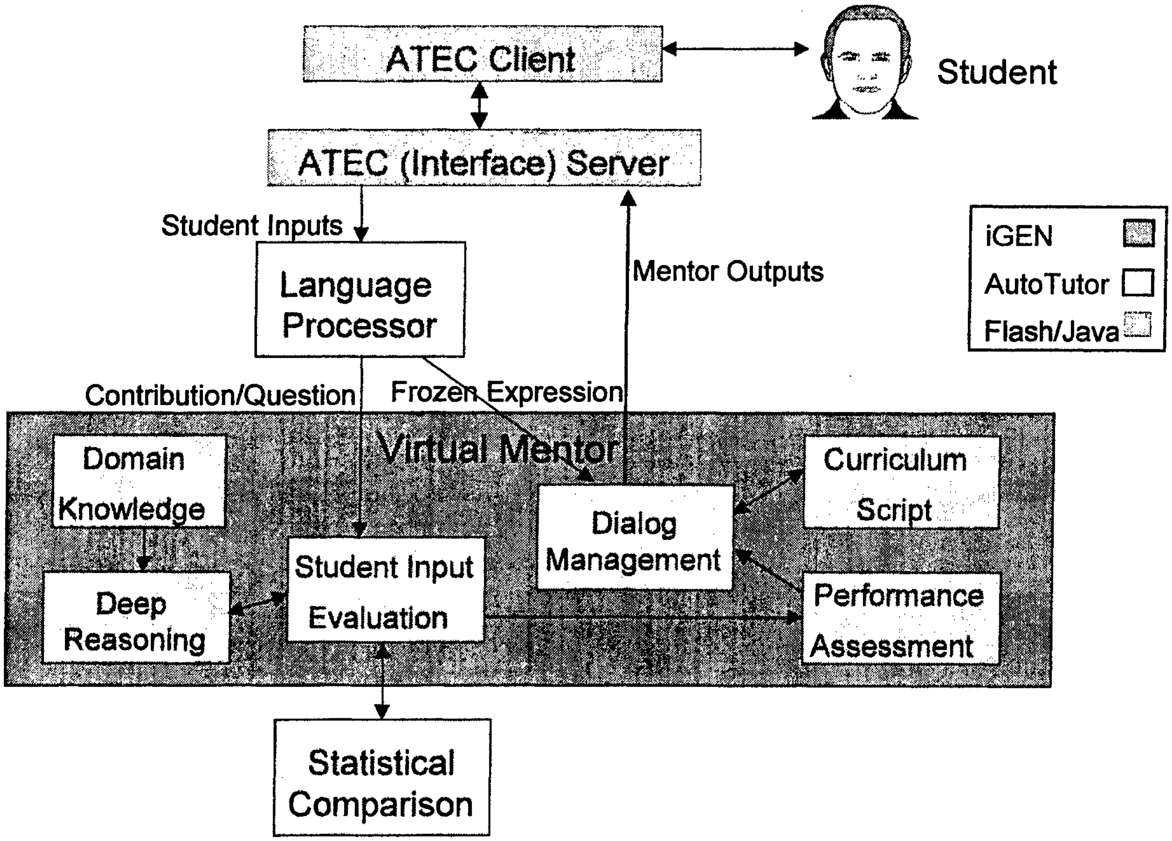


Figure 4. Functional architecture.

Dialog Management processes handle the microstructure of the tutorial dialog by specifying discourse markers, dialog moves, and responses to frozen expressions, as well as the logic for constructing dialog moves that are responsive to the student input and its evaluation.

Virtual Mentor

The Virtual Mentor (VM) provides the instructional reasoning of ATEC. As described above, it controls and maintains the list of questions that should be asked, evaluates what knowledge the student has demonstrated, and maintains a representation of the student's discussion of vignette aspects. It is implemented as an iGEN model, using extensions to iGEN that were developed to handle dialog (see section on iGEN Enhancements for Dialog and Performance Assessment below). Functionally, it incorporates the Curriculum Script and Dialog Management of AutoTutor as part of its instructional reasoning.

The main components of an iGEN model are: a problem representation *blackboard* containing declarative knowledge about the situation (including a metacognitive blackboard representing about the status of cognitive processing), procedural knowledge represented as *tasks*, and mechanisms for sensing the external environment (*perceptual demons*) and acting on it (*actions*). A communication shell allows the model to obtain perceptual inputs from the environment (in this case, student inputs parsed and classified into speech act categories), define actions on the external environment (in this case, mentor outputs), and invoke functions from external modules as part of its task procedures (in this case, the statistical comparator).

The VM contains tasks for asking questions related to each TLAC theme. Each task includes one or more dialog chunks—a hierarchy of questions and associated expectations (possible good answer aspects). Each dialog chunk has a leading question, any number of nested subquestions (more specific questions) and expected good answers (expectations) associated with questions. It is also associated with a theme. Whenever the VM asks a question and the student provides a response (contribution), the VM compares the contribution to all expectations to the question asked and to all subquestions. Any expectations that are satisfied are marked. Subquestions are not asked if their expectations have already been satisfied. Both the questions and the associated expectations are combined conjunctively or disjunctively, resulting in various types of AND/OR trees. This approach allowed the VM to use the mentoring style of TLAC, starting with general questions and only asking specific ones for topics the student did not discuss. These tasks comprise the Curriculum Script component of the VM.

In addition to the tasks for asking questions, there were tasks for managing local aspects of the dialog, including special speech acts (questions and frozen expressions), acknowledging good answers, and providing other discourse markers as needed to make the conversation smooth (see Louwerse & Mitchell, 2003; Louwerse, Graesser, Olney, & TRG, 2002). Discourse markers are defined and stored in a specific level of the blackboard called "Markers" which contains one sub-level for each type of marker. The tasks using the markers randomly select one marker from the needed category to make the conversation less rigid. These tasks along with the dialog processing mechanisms built into iGEN comprise the Dialog Management components of the VM.

The VM starts the execution of its highest priority tasks (a task with questions about one theme). The execution of the first dialog chunk sends the text of the leading question to the dialog interface window, and prepares a set of expected answers. The execution of the task is put in a suspension mode until the dialog interface window receives an answer from the user. When an answer is received, it is parsed and classified by the speech act classifier, which may also identify a set of key nouns that can be extracted from the answer (to aid in question answering). The answer is then posted in the metacognitive blackboard, along with its classification and attached nouns. If the classification corresponds to a special speech act, the task for handling special speech acts is triggered. If the speech act is not special and satisfies at least one of the expectations, then a task is triggered and executed which provides dialog transition markers for acknowledging success. This task initiates an acknowledgment marker (e.g., "ok"), and if more answers to the same question are still expected, it also initiates a next marker (e.g., "anything else"). Once the interrupting task (if ever triggered) is completed, the normal execution of the original task continues. Depending on the status of the question that was previously asked, the execution may continue by asking subquestions to this question or continue the normal execution of the task. Once a task is completed, the task with the next priority begins (usually a task related to another theme).

The task for handling special speech acts has sub-goals for answering definition and yes/no questions, avoiding questions it cannot answer (e.g., saying "That is a good question. What do you think?"), and responding to frozen expressions indicating lack of knowledge or lack of understanding.

Dialog Management and Curriculum Script

Dialog management can be viewed from the standpoint of macrostructure and microstructure. Macrostructure consists of major chunks of lessons to be covered and the points to be covered within each lesson. Microstructure consists of ATEC's dialog moves and micro-adaptation to the student during the course covering each point. The curriculum script plays the primary role in managing the macrostructure whereas a more complex dialog management mechanism is needed for handling microstructure.

The analyses conducted to determine the content for the curriculum script at the macrostructure level are described in the Domain/Vignette Analyses subsection of the Analyses Conducted section of this report. Although the topics included those related to multiple ways to approach the problem presents, there was no guarantee that all possible approaches any student might want to discuss were included. The development of the microstructure (i.e., specific subquestions) was determined by an analytical process of partitioning the content into smaller units for subquestions.

As discussed previously, the curriculum script consists of a hierarchical organization of themes, general and specific questions associated with themes, and expectations associated with each question. This can be expressed in the following simple rewrite rules, with * signifying there can be one or more of the designated constituents.

Curriculum Script → Theme$_1$ + Theme$_2$ + … Theme$_8$
Theme$_n$ → Theme-description + Question-answer-structure*
Theme-description → <verbal description that summarizes the theme>
Question-answer-structure → General-question + General-expectation +
Specific-subquestion$_i$ + Specific-sub-expectation$_i$
Specific-subquestion$_i$ → <verbal articulation of a question>
Specific-subquestion$_i$ → Specific-subquestion$_j$ + Specific-subquestion$_k$
Specific-subexpectation$_i$ → <verbal articulation of a subexpectation>
Specific-subexpectation$_i$ → Specific-subexpectation$_j$ + Specific-subexpectation$_k$
General-question → <verbal articulation of a general question>
General-expectation$_i$ → <verbal articulation of a general expectation>

The dialog macrostructure is organized around the eight TLAC themes. The themes are currently introduced in a fixed order. The questions within a theme are asked in an ordering that goes from general to specific following the nested structure; if the student articulates an expectation at a higher level, then all nested subordinate questions need not be asked. If the content of the user contributions has a very low match to the expectations, then the same sequence of questions would be produced for a theme, following a progressive deepening algorithm. If the content of the user contributions has a high match to the expectations, then many of the questions can be deleted so there is fluctuation in questions asked, depending on what the user articulates.

For example, a portion of the curriculum script dealing with the theme 'Mission' is shown in Figure 5 in text format extracted from the iGEN mentor model. There is one leading question, Q2, with four subquestions, some of which have their own subquestions. The associated expectations have corresponding numbers beginning with 'E' and are either disjunctive or conjunctive combinations. For a disjunctive set of expectations, only one must be satisfied for the question to be satisfied; while for a conjunctive set, all must be satisfied.

The dialog microstructure attempts to micro-adapt to a variety of potential student contributions during the process of attempting to have expectations covered. The ATEC handles some questions that users ask. In particular, it can potentially answer definitional questions (What does X mean?) and verification questions (Is X true?). Definitional questions are answered by consulting a glossary. If X is an entry in the glossary, then the meaning of X is produced and then ATEC returns to the same point in the dialog prior to the user question. If X is not an entry in the glossary, then ATEC announces that it cannot answer the question and goes on in the dialog as it normally would. The ATEC also responds appropriately to a variety of frozen expressions that users often type in.

For example, a metacognitive expression is "I don't follow" whereas a metacommunicative expression is "Could you say that again?" The ATEC attempts to respond appropriately to a large set of alternative speech acts that the user might enter. ATEC also adds dialog markers to its responses to improve the smoothness of the dialog. For example, it acknowledges user input with an Acknowledgement Marker chosen randomly from a set including 'Alright,' 'Good,' 'Okay,' 'Right' prior to proceeding with another question. There are also sets of markers for requesting further information when a response partially satisfies a question, markers for responding when the user input is not relevant to the question, among other things. An example of dialog corresponding to the curriculum script chunk in Figure 5 is shown in Figure 6. Appendix B shows the annotated log corresponding to this example dialog.

Language Processing Components

Most of the language processing components in ATEC are based on those used in AutoTutor. They include speech act classifier, syntactic parsers, and statistical comparison of text strings for evaluating conceptual similarity. Each is discussed in turn.

Speech Act Classifier. An important function in ATEC is that of speech act classification. As with AutoTutor, ATEC needs to determine student intent in order to flexibly respond to the student. A speaker who asks "can you repeat this" would probably not like to have a "yes" or "no" as a response, but would in fact like to have the previous utterance repeated.

The ATEC uses speech act classification in order to determine the student's intentions. This classification system is based upon the AutoTutor classifier that identifies 20 illocutionary categories. These categories range from assertions to metacommunicative and metacognitive expressions like "Can you repeat that?" and "I don't know" to 17 questions categories, as proposed by Graesser, Person and Huber (1992). The classifier uses a combination of syntactic templates and keywords. Syntactic tagging is provided by the Apple Pie Parser (APP) (Sekine & Grishman, 1995) together with cascaded finite state transducers.

Q2: What is the plan to achieve higher command's main objective?
  Q 2.1: What is your mission?
    Q 2.1.1: What is your objective?
    [E2.1.1. (Objective Sword | Sword | The crossing sites over Stranger Creek)].
    Q 2.1.2: What is your planned scheme of maneuver?
    [E2.1.2a. To infiltrate the security zone, attack and seize (Objective Sword | Sword | the crossing sites over Stranger Creek)
    OR
    E2.1.2b. Conduct an infiltration and attack to seize (Objective Sword | Sword | the crossing sites over Stranger Creek).
    OR
    E2.1.2c. (I | we | 2nd Bde | 2nd Brigade | Bde | Brigade | 2nd Bde Commander) will infiltrate, attack and seize (Objective Sword | Sword | the crossing sites over Stranger Creek).]
    Q 2.1.3: What is the planned end-stated of your mission?
    [E2.1.3a. Passages lanes are open for 25 AD attack.
    OR
    E2.1.3b. Seize (Objective Sword | Sword | crossing sites over Stranger Creek) to allow 25AD freedom of movement to PL Dime.
    OR
    E2.1.3c. Help 25 AD to move unhindered by enemy direct and unobserved indirect fires from PL Arkansas to PL Dime]
  Q 2.2: What are the key tasks you must accomplish to succeed in your mission?
  [E2.2a (Create an assailable flank | create and exploit an assailable flank)
  AND
  E2.2b (Penetrate and destroy the security zone. | Destroy the enemy security zone.)
  AND
  E2.2c (Support the destruction of the security zone combined arms reserve | destroy the security arms reserve)
  AND
  E2.2d (Deceive the enemy | deceiving the enemy as to the main attack | deceive the enemy as to the location of the main attack.)
  AND
  E2.2e (Clear passage lanes for the 25th Armor. | Clear the passage lanes to give 25th Armored Division freedom of movement from Phase Line Arkansas to Phase Line Dime.)
  AND
  E2.2f Cover the passage of the 25th Armored Division at EENT.]
  Q 2.3: How is the infiltration being conducted?
    Q 2.3.1: Who is conducting the infiltration?
    [E2.3.1 6-502 Air Assault Infantry will conduct the infiltration.]
    Q 2.3.2: When will the infiltration occur?
    [E2.3.2 (During the hours of darkness | At night.)]
  Q 2.4: What are the objectives of the infiltration?
  {E2.4a 6-502 will attack and seize (Objective Meade | Meade) to threaten the enemy's eastern flank
  AND
  E2.4b (Seizing (Objective Meade | Meade) will also protect 2nd Brigade's eastern flank. | Protect the Bde's eastern flank | Protect my flank.)
  AND
  E2.4c (Seize | capture) (Objective Meade | crossing sites) to facilitate the (mounted attacks of the 4-25 and 4-81 | mounted attacks)]

Figure 5. Example extract from curriculum script.

```
C What is the plan to achieve higher command's main objective?
U seize objs meade, hill, grant, and lee
C Lets discuss something more specific.
C What are the key tasks you must accomplish to succeed in your mission?
U identify or create an assailable enemy flank; destroy the CAR; seize key terrain along PL
   Dime; destroy enemy forces along passage lanes
C Okay,
C Can you add to that?
U those are the key tasks
C What are the objectives of the infiltration?
U force the enemy to withdraw
C What is your mission?
C Okay,
C Anything else?
U we will commence nlt 210530may
C Lets discuss something more specific.
C What is your planned scheme of maneuver?
U lead with a mech tf; then follow on with a mech and armor tf; w/ a mech tf in reserve
C How is the infiltration being conducted?
U the lead tf will conduct an air assault to establish contact with the main body following on
C Lets discuss something more specific.
C Who is conducting the infiltration?
U tf 6-502
C When will the infiltration occur?
U 6-502 will LD at 202100may
```

Figure 6. Example session dialog extract.

The finite state transducers consist of a keyword transducer (e.g., difference and comparison in the comparison question category) and syntactic templates. Extensive testing of the classifier showed that the accuracy of the classifier was better than 65% overall and as high as 87% for certain question categories, thereby similar to human performance (see Louwerse, et al., 2002; Olney et al., 2003). The ATEC responds to student contributions (assertions), frozen expressions (metacommunicative and metacognitive expressions), and three question categories (concept completion, definition, and yes/no). All other questions are handled by discourse markers indicating the VM does not know the answer. The speech-act-category tagged student input is used by the VM.

Glossary. Another component is a glossary used to answer certain types of questions. There is a glossary of terms and definitions that are used in the battle command reasoning. Whenever a learner asks a definitional or concept completion question ("What does X mean?"), then the glossary is consulted and a definition is produced for an entry in the glossary.

Syntactic Parsers. Syntactic parsing is needed in ATEC to provide the input for speech act classification, as discussed above, and for answering questions. Because parsers differ in their characteristics and output format, as well as the operating system they run under, different parsers may be needed for different purposes. In addition to the Apple Pie Parser used by the speech act classifier, ATEC uses the SCOL parser in conjunction with the glossary. Both these parsers are relatively fast, freely available, and commonly used (Louwerse et al., 2002).

The Apple Pie Parser (APP) currently has three versions, with one more in the works. These versions run on different systems including Linux, Windows NT, and Solaris, depending upon which version is used. The APP is a bottom-up probabilistic chart parser that looks for the parse tree with the best score by a best-first search algorithm. The output consists of codes that tag the words, phrases, clauses, and sentences; and brackets that divide the text into phrases, clauses, etc., appropriately (that is, a syntactic tree with bracketing is generated). The APP tries to make a parse tree as accurately as possible for reasonable well-formed sentences (e.g., sentences in newspapers or well-written documents). It is not designed to parse many ill-formed, but still seemingly reasonable, sentences, as for example those found in typical conversation.

The SCOL parser (Abney, 1996, 1997) is written for UNIX or LINUX systems. It generates parses and the corresponding parse trees. Parses tend to be rather long since the system is more fragment-oriented and less grammar-oriented than the APP. The parser does not necessarily generate a complete parse for a particular sentence, so you can end up with multiple parse trees in one of your files of parsed results. The SCOL is not focused on identifying complete grammatical parses of a sentence. It is only interested in identifying phrases and fragments like noun phrases and prepositional phrases. The SCOL parser uses 'nx' to identify structures it thinks of as noun phrases, and 'pp' for prepositional phrases. While for some purpose, it is a drawback of the system that it is very 'noun' oriented, i.e., words that cannot be classified are automatically classified as nouns, this is a benefit for input to a glossary used for answering questions as only nouns are tagged.

Statistical Comparison. The statistical comparison component is a pattern matching component comparing the student's contribution with a set of possible expected answers. Take, for instance, the following tutor and student turn and the expected answer.

- Virtual Mentor: "What does the enemy know about you?"
- Student Contribution: "The enemy knows nothing."
- Expected answer: "It is likely the enemy does not know 2nd Bde's intent and plans at this time."

The degree of similarity is computed, and if the similarity is greater than a selected threshold value (currently set to .5) the input is determined to match the expected response. In ATEC, multiple similarity values are computed – any given student contribution is compared to all expectations to the question asked (each unitary component of the answer is considered a separate expectation) and to all expectations associated with all subquestions in the question hierarchy. The match value for all expectations is posted on the metacognitive blackboard for the iGEN model to use in its evaluation of student performance and for dialog management.

The ATEC used the Inverse Document Frequency (IDF) algorithm that is quite similar to the algorithm that Graesser, Hu, Person, Jackson, Steward, and Toth (2002) had successfully used in the past in the context of query-based information. The IDF is an extended weighted keyword matching algorithm. That is, the comparison assigns weights by the inverse of the frequency of the words in a corpus representing the domain. Since the less frequent words are usually the more important in terms of information conveyed, a student input containing many of the important content words in the expected answer will be given a high similarity value, and if greater than the threshold set, will be considered as a good answer to the question. The algorithm can be extended to include using domain-specific synonyms in addition to using the actual words in the expectation (e.g., Division Commander and Commanding General).

For any given corpus, IDF can be obtained for each word as follows: The corpus is processed so that each word is reduced to its stem form. For example, attacking, attacked, all are reduced to ATTACK. Then the fraction (N/n_i) is taken, where N is the number of paragraphs in the corpus and n_i is the number of documents in which the word 'i' occurs. The IDF of each term is used as the weight of that word in the given corpus. IDF for any set of words is the sum of the IDF for each individual word in the set. To calculate the similarity, the following procedure is used: Assume that S_1 is the set of words in the user input and S_2 is the set of words in the expectation against which the user input is compared.

1. $S_1 \cap S_2$ contains words that are present both in S_1 and S_2
2. Similarity between S_1 and S_2 is computed using the following formula:

$$2\frac{idf(S_1 \cap S_2)}{idf(S_2)} \tag{1}$$

For the example question and answer earlier in this section,

- S_1= {"the," "enemy," "know," "nothing"}
- S_2={"It," "is," "likely," "the," "enemy," "do," "not," "know," "2nd Bde," "intent," "and," "plan," "at," "this," "time"}. The join of S_1 and S_2, ($S_1 \cap S_2$)= {"the," "enemy," "know"}.

For each individual word, IDF is obtained, then the similarity between S_1 and S_2 is computed as in (1), where

- IDF($S_1 \cap S_2$)=IDF("the")+IDF("enemy")+IDF("know")
- IDF(S_2)= IDF("it")+IDF("is")+IDF("likely")+ IDF("the")+IDF("enemy")+IDF("do")+ IDF("not")+IDF("know")+IDF("2nd Bde")+ IDF("intent")+IDF("and")+IDF("plan")+ IDF("at")+IDF("this")+IDF("time").

The value of the similarity computed is compared with a threshold. If the value exceeds the threshold, then the expectation is considered met.

Alternative methods of statistical comparison and the evaluation conducted to select IDF are provided in the section below on Analysis of Statistical Methods for Input Evaluation.

Performance Assessment

The goal of performance assessment in ATEC is to evaluate the student's critical thinking skills. Since the system is dialog-based and there are no correct or incorrect answers, the goal of the training is to be able to consider all relevant information within the eight themes. Our approach to assessment is based on the depth of questioning needed in each aspect of each of the eight patterns of expert thinking (themes). Performance is considered better if the student generates a response to a high-level question that incorporates all information relevant to that theme. Performance is deemed poorer to the extent that additional, more specific questions and prompts are needed to elicit responses incorporating the additional relevant information. Following this logic, a student who presents a thorough response to the vignette, a response that touches upon the relevant information with a few probe questions, would be assessed higher than a student who provides the same amount of information but requires more extensive probing or prompting. Scores are calculated for each theme and for overall performance (average of the theme scores).

Automatic performance assessment calculation. A score will be calculated for each dialog chunk (leading question), ranging from 0 to 1, with 1 indicating all expectations satisfied when only the leading question has been asked, and 0 indicating no expectations satisfied and all subquestions asked. (If a line of questioning is not productive, not all subquestions may be asked even if no expectations are satisfied. In those cases the score will be 0.) If a partial answer is given, and the VM asks "Anything more?" that will not be counted as another question, since no further information is given. Intermediate scores are calculated as follows:

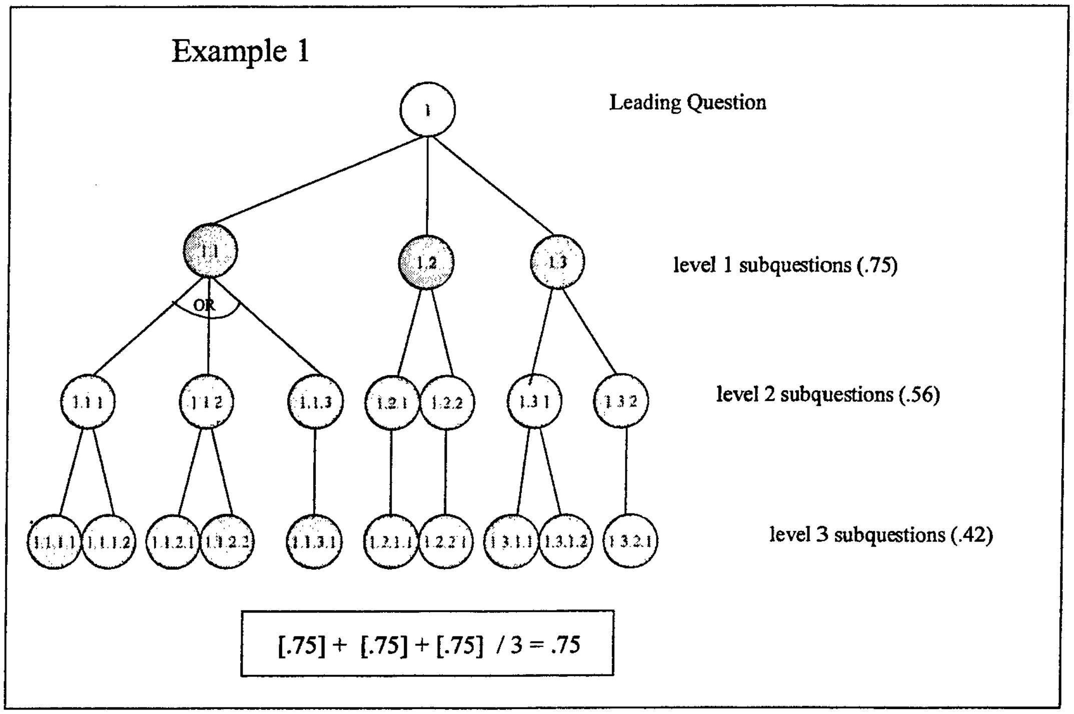
1. The score is aggregated at each level of the question hierarchy. For x first-level AND subquestions, each subquestion contributes 1/x of the dialog chunk score.
2. If a question or subquestion is not fully answered, that 1/x of the dialog chunk score is calculated based on the next-level subquestions. If there are y AND subquestions, each contributes 1/y of the 1/x contribution yielding 1/xy of the total score.
3. At any further sub-level, the contribution to the total score is again partitioned based on the number of subquestions at the next lower level.
4. If the subquestions at any level are OR questions, full credit is given if all expectations within any part of the disjunction are satisfied. If no part is fully satisfied, then the maximum partial score is used.
5. At the terminal level of subquestions, if all expectations are satisfied, full credit is given. However, partial credit is given based on the proportion of expectations satisfied, with OR expectations calculated similarly to OR questions.
6. The score at each level is weighted, with the score discounted by the depth attenuation factor. In other words, less credit is given as lower levels of subquestions need to be asked.
7. The default depth attenuation factor has been set at .75 yielding the following weights:
   1st level subquestion - .75
   2nd level subquestion - .75 (.75) = .563
   3rd level subquestion - .75 (.75) (.75) = . 422
   4th level subquestion - .75 (.75) (.75) (.75) = . 316
   etc.

If all subquestions (to whatever depth the question hierarchy includes) are asked and no expectations are satisfied, the score for that dialog chunk is 0.

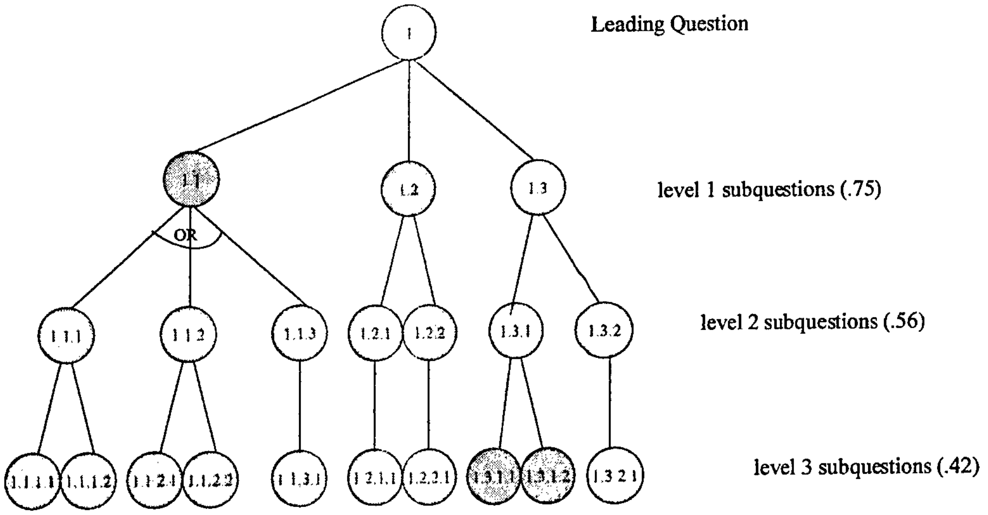
The depth attenuation weights are set at initialization, so they can be easily modified.

Examples of question hierarchies and scores (using color coding key below) are provided in Figure 7.

●	Asked – Satisfied when asked	○	Asked – Not Satisfied when asked	◌	Not asked

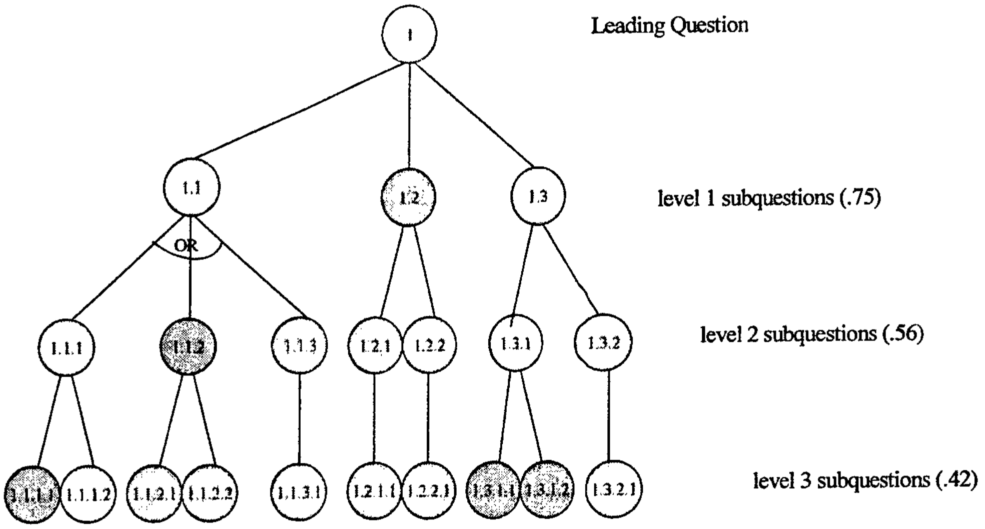


Example 2



$$[\,.75] + [\,.75] + [(.42 + 0)/2]\,/3 = .57$$

Example 3



$$[(.56)] + [(.75)] + [\,(.42 + 0)/2)\,]\,/3 = .51$$

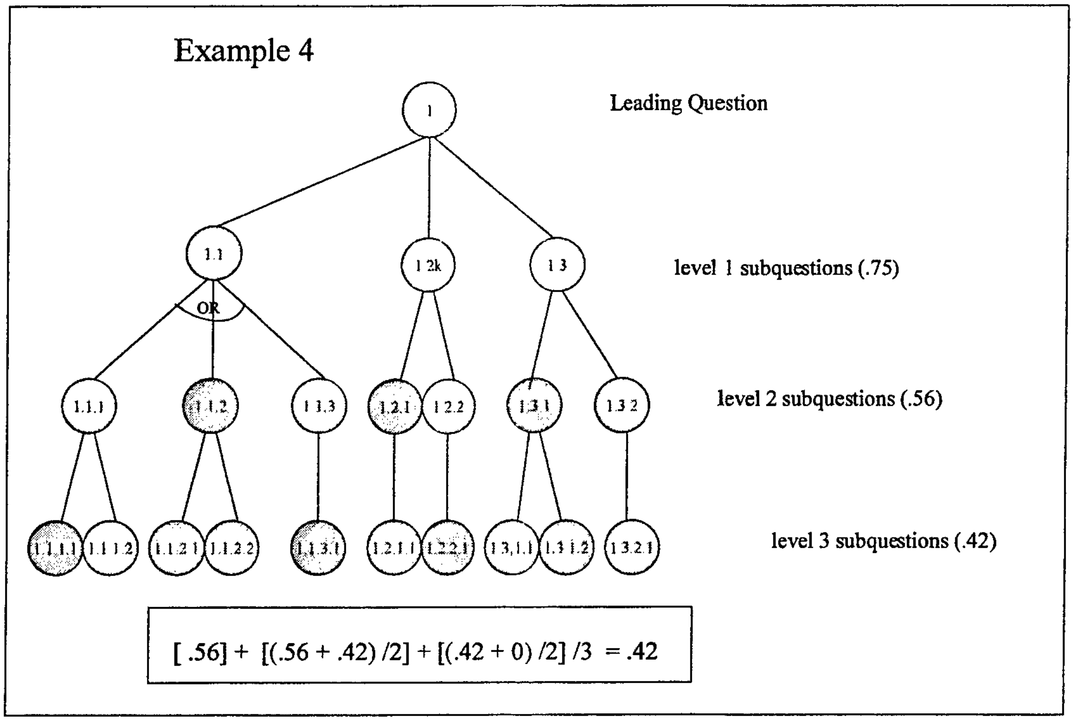


Figure 7. Four examples of performance assessment calculations.

Performance Assessment Summarization in the Model. The dialog-chunk scores will be aggregated to yield theme scores and overall scores. The theme score will be the average of all associated dialog chunks. Similarly, the overall score will be the average of the theme scores. A graphic summary is shown in Figure 8.

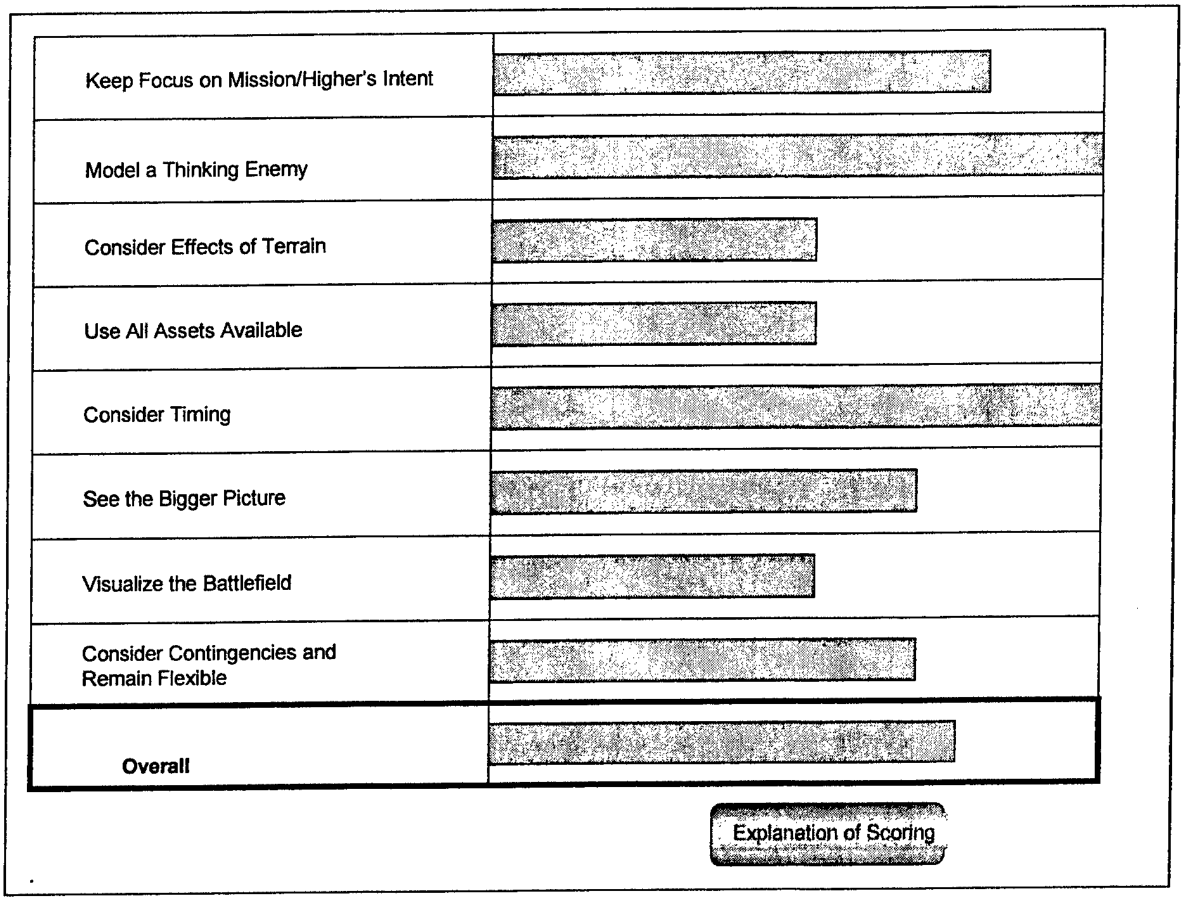

Keep Focus on Mission/Higher's Intent
Model a Thinking Enemy
Consider Effects of Terrain
Use All Assets Available
Consider Timing
See the Bigger Picture
Visualize the Battlefield
Consider Contingencies and Remain Flexible
Overall
Explanation of Scoring

Figure 8. Screen layout for performance assessment summary.

Software Architecture

The ATEC software architecture uses a distributed design to allow multiple simultaneous users to access the system over the Internet. This design consists of the following components (from right to left in Figure 9) the: (a) client side of the system, (b) the web server, (c) iGEN components, and (d) language components. The student accesses the system through a web browser (currently only Microsoft Internet Explorer is supported). An Apache Web Server is used to serve the ATEC website and Flash animations to each Student Client. Although not implemented (as indicated by gray), the design allows for a Student Client to log student performance and maintain a persistent student profile between sessions. The iGEN components are served by a python Session Manager that initiates an instance of the Blackboard Architecture for Task-Oriented Networking (BATON) engine of iGEN for each user logging onto the system. A python Dialog Web Server accepts student dialog inputs and sends them to the appropriate iGEN instance. A Simple Object Access Protocol (SOAP) client initiates the needed language modules for each iGEN instance. The language modules are implemented as web services using AXIS (Apache). Once the web service is initiated it takes the text passed from iGEN and calls the needed language service—the language module incorporating the speech act classifier (which calls the parser) or the statistical comparison logic using the MySQL Database (which contains the matrices for the statistical comparison). The web server and iGEN components run under

Windows and are currently hosted on one processor. The language components run under Linux and are hosted on another processor.

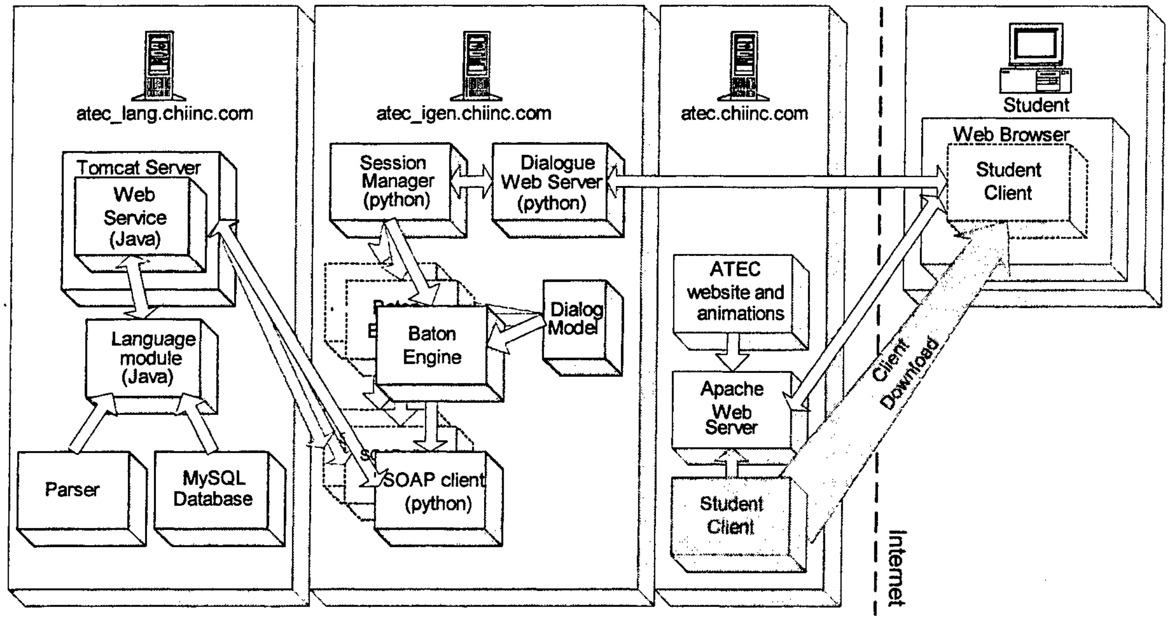


Figure 9. Software architecture.

Summary of ATEC Description

In summary, this section described the ATEC system. First, it described how the system design evolved from the component technologies of iGEN and AutoTutor and the overall pedagogical approach. It then provided a description of the system from the point of view of a user, showing the user interface. From the user view, the description moved into the system functions and components, describing the functional architecture first, followed by component descriptions, including the virtual mentor, the approach to dialog management and the curriculum script, the language processing components, and the approach to performance assessment for student modeling. The final subsection provided the software architecture.

The next section provides details on the analyses conducted to make system design decisions. It also includes discussion of research conducted to enhance ATEC or similar dialog-based tutoring systems in the future.

Analyses Conducted

Initial Data Collection

An initial data collection trip was undertaken at Fort Hood, Texas, from 1-4 April 2002. There were three goals for this data collection trip:

- Validate and extend the set of questions and expected good student responses for the initial vignette selected for development (Vignette 5),

- Determine how people refer to spatial locations – both verbally and using pointing in order to determine whether to incorporate a mechanism for allowing students to point to the situation display using the mouse rather than having to say/type location information, and,
- Determine whether there was any difference in communication content or amount when speaking versus typing.

Participants were three First Lieutenants, five Captains, and five Majors from the 1st Cavalry and 4th Infantry Divisions at Fort Hood. After being briefed about the purposes of the research effort, participants were provided hard copies of background information, including the "The Road to War" and supporting orders. After reading these documents, the participants viewed an electronic presentation of a TLAC vignette. The participants responded to this vignette by: (a) typing responses to 24 questions from the curriculum script (three participants), (b) providing verbal responses to the 24 questions (four participants), or (c) working in pairs to develop a verbal summary of intended actions (six participants). In the conversation only sessions, we provided printed schematic maps similar to those used in TLAC (roughly 1:50,000 scale but with less detail) to see if and how spatial references are made via pointing. At the end of each session the participant was debriefed and asked to provide feedback on the Road to War materials, the vignette presentation, the questions asked during the data collection and anything else related to the session. The less experienced officers (the three First Lieutenants and four Captains) were assigned to conditions (a) and (b) which addressed communication content and amount when speaking versus typing and spatial reference, and the more experienced officers (the five Majors and one Captain) were used in condition (c), to validate the curriculum script.

Feedback from the participants on the "Road to War" and Vignette 5 presentation was very constructive and provided inputs for improvements. The data collection provided the following findings:

- It was observed that the officers would perform the amount of typing required to make the ATEC system effective.
- Feedback from the participants on the "Road to War" and Vignette 5 presentation was very constructive and provided inputs for improvements. These improvements included elimination of information that was unimportant to the participants and implementation of "Road to War"/Vignette presentation suggestions. This feedback also allowed us to expand the user responses in the curriculum script and restructure or eliminate poor, ambiguous questions.
- It was observed that a better presentation of the background information was needed, as participants tended to spend between 30 and 45 minutes going over them. It was determined that this information should be presented electronically.
- Spatial references were most frequently linguistic indicating that the need to include mouse pointing was not needed.
- Although we were able to obtain good data from the pool of participants available, it became clear that higher grade level participants, Lieutenant Colonel and above, would be needed to obtain a better validation/expansion of the curriculum script content.

- An issue arose in which the participants wanted to see the system use the real maps they are most familiar with (i.e., 1:50,000 to 1:250,000 scale maps). After discussions about this issue with ARI, we determined that the TLAC-like maps were sufficient for Phase II and its focus on brigade level operations.

Domain/Vignette Analyses

Development of the curriculum script began with an analysis of the vignettes and supporting materials created by the ARI, Fort Leavenworth, KS and Fort Knox, KY. To provide as extensive coverage of expectations as possible, the analysis was conducted in three phases on two of the vignettes. The first phase of the analysis was done with an in-house Subject Matter Expert (SME) to develop an initial assessment of the vignettes based on the eight TLAC themes. This SME was a Captain in the Army with extensive experience working as a Battalion and Brigade level staff officer. This assessment resulted in a series of probing questions designed to help a learner think about the vignette in terms of the eight themes and a series of expected answers for each question.

The second phase of the analysis involved the data collection at Fort Hood described in the previous subsection. These interviews helped to validate the questions, and to expand the content of the expectations with the answers provided by the users. The final phase of the analysis was an independent assessment of the vignette by a retired Brigade level commander. This SME conducted his assessment based on the eight TLAC themes to further validate the questions developed in phase I and to expand the question expectations even further. Once the analysis was completed, the curriculum script was developed with questions and their associated expectations grouped according to the eight TLAC themes. The curriculum script question and expectation structure was used as the baseline for the development of the ATEC model.

Analysis of Statistical Methods for Input Evaluation

The ATEC needed a flexible algorithm for performing pattern match operations between tutor expectations and responses typed by learners. The algorithm would need to allow partial matches based on similarity in meaning. AutoTutor uses LSA as its mechanism to represent meaning in comparing user inputs with expected answers. An alternative using a weighted inverse frequency keyword algorithm was used successfully in the past in the context of query-based information retrieval (Graesser, Hu, et al., 2002). Thus, a range of algorithms were analyzed to determine whether there were alternatives to LSA that could provide similar function without having to purchase a commercial license. There are various ways to obtain this similarity measure: word matching, keyword matching, weighted keyword matching, extended weighted keyword matching, and latent semantic analysis. Each will be briefly discussed next.

Word Matching. Word matching can be considered as the simplest comparison. It calculates how many common words appeared in the expectation and the student's contribution. This method does not depend on context and does not depend on domain knowledge. Quality of the contribution is a function of the total words (the union of the two) and the common words (the join of the two). The mechanism of the computation can be understood as a Venn-diagram. The most commonly used formula for the computation is the ratio of the common words and the total words, which restricts the similarity measure to between 0 (completely different) and 1 (completely identical). The advantage of "word matching" is its intuitiveness and its ease of computation.

The disadvantage of this method is the lack of emphasis on important words. For example, in the expected answer "*It is likely the enemy does not know 2nd Bde's intent and plans at this time*" the word "*enemy*" carries more information than words such as "*it*" and "*and*" that provide a more common function. Word matching simply ignores this difference. A technique that does take into account important words, while ignoring function words, is the keyword matching comparison.

Keyword Matching. In this comparison method only "keywords" are considered. Function words like "it" or "the" are ignored. The advantage of this method is the emphasis on the important words. Keyword matching is the most widely used method in document retrieval. However, there are some requirements that need to be satisfied in order to have this method work well. This is particularly true for narrowly defined domains, where it depends on the domain whether a word can be qualified as a keyword. In most of the cases, the list of key words is simply the list of glossary items. The advantage of the keyword matching method is that it distinguishes between common function words and meaningful keywords. The disadvantage is that the method is very domain dependent. Furthermore, there is still differential importance as to how much information it carries. Weighted keyword matching differentiates the relative importance of keywords.

Weighted Keyword Matching. The advantage of weighting keywords differently is to emphasize special terms based on the domain. For example, in the expected answer " "*It is likely the enemy does not know 2nd Bde's intent and plans at this time*" the words "*enemy*" and "*plans*" may be weighted differently. There are varieties of ways one can assign weights to words. For example, one can assign a weight to a word based on its frequency in ordinary written language. Another way to assign weight is to consider how important the word is in the context. The former is easier because it can be obtained either by computing the relative frequency from a given corpus, or by using an established database (e.g., CELEX, Piepenbrock, 1993). The latter is harder, because it will be heavily dependent on domain expertise. From the performance point of view, a proper combination of both is the best. The disadvantage is the difficulty of assigning the weights in advance. Like the keyword matching method, the weighted keyword matching method is heavily domain dependent.

Extended Weighted Keyword Matching. All of the above three methods are very simple matching methods. Even though the weighted keyword matching is very flexible due to the variety of ways of assigning weights, it is still based on exact matches of words. A more sophisticated method of matching would include synonymous words. For example, the word "*enemy*" has synonyms like "*opponent*" or "*foe*" words. The original expected answer "*It is likely the enemy does not know 2nd Bde's intent and plans at this time*" should therefore be extended to a larger set of words that includes all synonyms of the original keywords. This means that even if student's input does not exactly match the original keywords, it is still possible to obtain the similarity between the expected answer and the student's contribution.

The advantage of this method is that it allows for nonzero-similarity even if there are no common words between expected answer and student's contribution.

All methods mentioned so far are, in essence, matching strings of letters. Each word is represented as a string of letters (words or synonyms) and the importance of a word (weights based on frequency or domain importance). In order to make all work well, there are several computational linguistic tools needed. For example, in order to obtain the word frequency information, one needs to have word frequency lists (Kucera & Francis, 1967; Piepenbrock, 1993). In order to obtain synonyms, one needs to use lexical databases like WordNet (Fellbaum, 1998). However, existing lists and databases are constructed from corpora representing general English usage and do not cover specialized terminology in any particular domain. When dealing with Army command reasoning these lists and databases will not give accurate frequency counts or synonyms. Specialized lexicons and/or synonym lists must be constructed.

However, there is another way of computing similarity between texts that is not based on matching letter strings. The smallest unit in the above mentioned "matching method" is the word. Matching of two words is the basic computation. Any two words are either matched or not matched. There is no intermediate. The new method starts with a new representation of each word. Instead of using a letter string, it uses a numerical vector in a multi-dimensional space. In this case, the lowest level of computation is no longer at the level of the word; it is at the level of each dimension of the vector for the word. One example of such method is LSA.

Latent Semantic Analysis (LSA). The LSA represents each word as a real vector (up to 500 dimensions). The similarity between any two words is simply the normalized dot-product (cosine) of the two corresponding vectors. The key to LSA is the vector representation of the words. The way LSA obtains the vectors for a given domain is simple. First, the domain knowledge in the form of a repository of documents is collected. For example, the repository may be all the available texts in a given domain (combat guidelines for the Army, for example), or even in a general world knowledge, like the encyclopedia. Next, the documents are indexed so each document has a unique number. This number may be as high as a million. Each word is then represented as a vector with the same dimensions as the number of the documents in the repository.

For each word, the values in the corresponding vector are determined as follows: if the word appears in document i n_i times, then the ith entry will take a value that is a function of n_i and i. All words together form a huge matrix with the number of rows equaling the number of words, and the number of columns equaling the number of documents in the repository. After the huge matrix is obtained, the most important dimensions are found by conducting a principle component analysis to the matrix. The LSA then uses only the most relevant dimensions to represent each word and use it to obtain similarity between the words (see Hu, Cai, Graesser et al., 2003; Hu, Cai, Franceschetti et al., 2003). When LSA is used to compare two documents, it simply first adds the vectors from each document separately (vector summation) and computes cosine values from the vectors.

Evaluation. There are several advantages to using LSA. First, computationally it is a rather simple technique. For instance, it does not need a frequency database for all the words, but automatically retrieves the frequency from the repository of available documents. Secondly, LSA does not need additional methods to obtain synonyms. Instead, the co-occurrence of the words in the documents estimates the similarity of the meanings. The use of LSA in commercial systems requires a moderately expensive license ($50,000 initially and $25,000 per year at the time this effort began, although the initial cost has since been eliminated). Because alternative and less costly methods are available, ATEC does not use LSA. Instead, we use the next best method, that of extended weighted keyword matching. Graesser, Hu, et al. (in preparation) performed a number of comparisons between matching algorithms in a system that answers questions posed by learners in natural language. The extended weighted keyword matching algorithm alone did slightly better than LSA did alone, but a hybrid system that combined LSA and the keyword algorithm did approximately 8% better than the extended keyword algorithm alone. The general wisdom is that LSA is expected to be particularly powerful when there is a greater reliance on inferences and implicit world knowledge than on exact matches to explicit text.

Improvement of Statistical Methods

Threshold. The information needed to determine a reasonable threshold is the statistical properties of all the inputs. It may also involve systematic evaluations from field experts. For example, one needs to obtain the statistical distributions of all the similarity measures for any given input size. Let's assume n_1 and n_2 are the size of the student's input and the size of the expected answer for a given seed question. The mean and standard deviation of the similarity measure are: $m(n_1,n_2)$ and $s(n_1, n_2)$. This specifies the distribution of similarity measures for all students' inputs (with n_1 words) and the expected answer (here n_2 is the number of words in the expected answer). For any specific input from a student, a z-score can be obtained, which measures how good the input is in comparison to the population (all other possible inputs from students). The arbitrary value of 0.5 (currently used in ATEC) may be appropriate for some document sizes, but not for others. One method for potentially improving the responsiveness of ATEC is to fine-tune the threshold.

To address the issue of threshold, there are some possible steps to take for fine-tuning:

1. For any of the seed questions, collect all the answers from real students (with variable domain knowledge).
2. For each seed question, apply the above-mentioned formula to all possible inputs for the question and the expected answer. This will result in two empirical distributions: the distribution of the length of the response, and the distribution of the similarity measures as a function of the input size.
3. For each student input, have the experts rate the quality.
4. Find the threshold (z-score) for good, median, or poor answers.
5. Use the threshold in the evaluation of the students' inputs.

If the threshold is determined in this way, the similarity measure of 0.5, will correspond to different z-scores for different document sizes, hence it may exceed the criteria (as good answer) in some cases and fail in some other document sizes.

An analysis was conducted using a small set of input logs generated by six ARI personnel to determine how closely the IDF algorithm matched expert judgments as to whether the inputs were good or not. A SME rated the 351 answers input to ATEC. The mean of the ratings was 4.37 (1 = best and 6 = worst) with SD of 1.09. However, comparable IDF ratings had mean of 0.031 (0 = worst and 1 = best) with SD 0.98. The correlation between human and IDF algorithm was substantial (r = 0.47), but low. Comparing human ratings with IDF, among only the 48 answers rated as good (1 = good versus 2-6 = not good) by the SME, only 4 of the 48 good answers exceeded IDF of 0.5.

Further analysis of recall and precision were conducted to evaluate the relationship of IDF to expert judgments. Recall is the ratio of true positives (both IDF and expert evaluate as good) to all expert positives, while precision is the ratio of true positives to all IDF good responses. Considering only good responses, an IDF threshold of .5 yielded a recall value of 0.18 and a precision value of 0.75. If a threshold of .4 was used, recall increased to .31 and precision dropped slightly to 0.71. Recall is particularly important as a goal to minimize false negatives (inputs that an expert rates as good, but the IDF does not) since those are most likely to make the system respond inappropriately to the user. In sum, this analysis indicated a threshold of .4 might result in smoother dialog.

Corpus. The IDF method is also improved with the size of the document corpus used. Thus, enlarging the corpus may improve the quality of the match. However, the improvement is only possible with relevant additional documents; that is, those using the same terminology as the expected student inputs. The initial corpus was enlarged by adding text from additional documents. It is an open question whether further improvement could be accomplished with additional documents.

Spell Checking. Not surprisingly student input contains spelling errors. In fact, research has shown that human typewritten text contains approximately 1%-3% spelling errors (Grudin, 1983). Correcting these errors in intelligent tutoring systems could be important, particularly if word comparison techniques are used. A typo in a correct answer could be interpreted by the system as a wrong answer, particularly if there are no accompanying words supporting the misspelled word. Furthermore, given that ATEC students are monitoring the situation, evaluating possibilities and thinking through potential solutions, attention on typing and typing errors may be reduced.

Implementing spell-checking utilities in intelligent tutoring system is difficult. First, the system cannot anticipate what errors the student will make. A possible solution to this problem lies in matching the student input to a large lexicon and calculating the probability of the correct spelling of a word. The problem with this is that the lexicon will have to be extremely large, accounting for words both within and outside of the domain. Besides, matching each word against this lexicon will slow down the responsiveness of the system considerably.

Various spell checking software packages are available, http://www.spellchecker.com and http://www.spelling-software.com, that are mostly comparable to the spell check facility in Microsoft Word. Spell checking can of course also be developed using Bayesian networks and n-gram analyses. Currently web-based spell check facilities that work with specific domains are not implemented in the ATEC tutor. Because the statistical comparison method used in ATEC is not based on a word match only, misspelled words will still allow the system to interpret and respond naturally.

Semantic Reasoning

One of the goals for ATEC was to incorporate a hybrid approach for evaluating student inputs, combining statistical techniques with deep reasoning using a semantic representation of selected aspects of domain knowledge. For example, statistical methods do not allow distinctions between "move from Phase Line Dime to Objective Meade" and "move from Objective Meade to Phase Line Dime." Such distinctions could be made using symbolic representations using a frame or propositional structure. An iGEN-based instructional agent could then reason about the semantic representation of student input in comparison with semantic representations of expected answers to questions or in comparison with a representation of the vignette semantics and possible appropriate actions in the circumstances. A number of issues would need to be addressed to develop a workable solution, including: (a) natural language understanding, (b) semantic representation format, (c) reasoning methods for evaluation of student inputs, and (d) natural language generation.

The first problem to solve is natural language understanding. Thus, we undertook an investigation to determine whether there are tractable methods for automatically extracting and representing the underlying meaning (semantics) of unconstrained natural language. The results of that analysis are presented here.

Deep Versus Shallow Natural Language Understanding. One of the important distinctions in natural language processing techniques is between shallow and deep symbolic language understanding. In a deep language analysis a complete spanning analysis of each input sentence is required. On the other hand, in a shallow semantic analysis a sequence of smaller constituents is identified, even if they are interrupted by un-analyzable spans (Stede, 2003). Deep is complete, shallow is partial.

Clearly, the distinction between shallow and deep is one of gradation, rather than a binary distinction. On one side of the spectrum, a complete understanding of an utterance is unlikely and inefficient; on the other, a partial understanding of an utterance without a deep knowledge of the essential building blocks sometimes falls short. Dialog planning based on formal reasoning is extremely difficult. However, our research goal was to determine whether there were reasoning methods based on shallow semantics that could be used to supplement the pattern matching of statistical methods to enhance the dialog.

Lexical Databases for Computational Linguistics. Despite the fact that various computational linguistic techniques like LSA, IDF and neural networks have been developed to derive the semantic and syntactic relation between words, sentences and paragraphs, there has always been a need for lexical databases. One of the reasons databases are needed relates to the high precision and detailed information that is often required in natural language processing. Identifying relations with a high precision rate for a large number of words remains a task best handled by databases. The price that has to be paid for this is that databases are extremely labor expensive and very inflexible.

Currently, the most developed and most commonly used database for English is WordNet (Fellbaum, 1998), an online lexical reference system that identifies a large number of nouns, verbs, adjectives and adverbs and lists their sense entries. But the database is more than just an electronic dictionary. In addition, WordNet organizes these entries in sets of synonymous relations. These relations can hold between words, their senses or even between synonymous relations themselves.

Because WordNet deals with lexical relations between words, it is no surprise that the database only consists of lexical (open class, meaningful) items. Take for instance the entry "attack." WordNet will divide this entry into a noun, verb or adjective, each with a number of senses (9, 6 and 1 respectively). For instance, the verb senses range from "launch an attack or assault on; begin hostilities with, as in warfare" like in "Serbian forces assailed Bosnian towns all week" to "set to work upon; turn one's energies vigorously to a task" like in "I attacked the problem as soon as I got out of bed."

Let's focus on the first sense, the warfare domain, and related WordNet returns:

2a. fight, struggle -- (be engaged in a fight; carry on a fight; "the tribesmen fought each other" or "Siblings are always fighting")
2b. =>defend -- (be on the defensive; act against an attack)
2c. => submarine -- (attack by submarine; "The Germans submarined the Allies.")
    => pepper, pelt -- (attack with missiles or questions)
    => strike, hit –
2d. *> Somebody ----s something; *> Somebody ----s somebody

WordNet returns synonyms as given in 2a above, antonyms as given in 2b, troponyms (particular ways to attack) as given in 2c above (2 of the 28 given here), and sentence frames as given in 2d and familiarity counts. In addition, a range of other relations are identified by WordNet including whether words are causal (e.g., 'break') or meronymous relations (parts of an attack).

WordNet does not deal with specialized domain vocabularies, and consequently could not assist with determining that "CG" or "Commanding General" are essentially used interchangeably. Thus, any use of WordNet to determine equivalency of words would need to be supplemented by a domain specific synonym list. Furthermore, WordNet does not deal with word order or semantic relationships among words in a sentence.

A project related to WordNet can solve the problem of word order and therefore is an attractive computational linguistic addition. The FrameNet project (Ruppenhofer, Baker, & Fillmore, 2002) is a lexicon-building effort in which frames and conceptual structures that underlie words are identified. Like WordNet, FrameNet is a dictionary and thesaurus and like WordNet it goes far beyond that. FrameNet is founded on corpus attestation, identifies frames going far beyond WordNet's "Somebody ----s something" and contrary to WordNet identifies relationships among words in a single frame that are of different parts of speech. These frames are intuitive constructs that formalize the links between syntax and semantics forming schematic representations of situations involving various participants, props, and other conceptual roles (the frame elements). For instance, FrameNet will tell us that the verb "attack" will involve an assailant, a victim and a weapon, in addition to place, time, reason and purpose. In addition, FrameNet states that this frame is related to lexical units like ambush, assault, charge, fall, invade, offensive, onslaught, strike, etc. This makes FrameNet extremely useful for word sense disambiguation, machine translation, information extraction, and question answering. It is the information extraction capability of FrameNet that provides the most potential for automated meaning extraction for an ITS such as ATEC.

The emphasis of FrameNet is on semantic relations. More emphasis on syntax might be needed for a more advanced use of verb information for intelligent systems. VerbNet offers this additional information. VerbNet is a verb lexicon with syntactic and semantic information for English verbs. It uses verb classes categorized by Levin (1993) to systematically construct lexical entries. VerbNet identifies a set of semantic predicates associated with each syntactic frame in a verb class. Whereas FrameNet uses lexical semantics as a basis, VerbNet uses argument syntax. The result of this is that FrameNet provides frame descriptions, VerbNet provides verb classes and alternations.

The other difference lies in the assumption underlying VerbNet. It assumes that verbs across languages are semantically similar. See Baker and Ruppenhofer (2002) for a more extensive comparison. For instance, if we take a verb like "to seize" VerbNet provides the following information:

Thematic Roles
Agent[+animate OR +organization]
Beneficiary[+animate]
Source[+animate OR [+location -region]]
Theme[]

Frames
-Basic Transitive
-Benefactive Alternation (for variant + Source PP)
-Benefactive Alternation (for variant)
-Transitive (+ Source PP)

[abduct, cabbage, capture, confiscate, cop, emancipate, embezzle, extort, filch, flog, grab, hook, kidnap, knock_off, liberate, lift, nab, nobble, pilfer, pinch, pirate, plagiarize, purloin, rustle, seize, smuggle, sequester, snatch, sneak, steal, swipe, take, thieve, wangle, weasel_out, wrest]

By having more precise information than what is available in FrameNet, VerbNet can provide more specific questions, for instance regarding the source and benefactive (entity that benefits in the identified event). However, it is not important to illustrate how VerbNet can provide information exclusively. Instead, it is important to notice how databases like WordNet, FrameNet and VerbNet are complementary. What one database cannot provide or cannot provide well, can be taken from another database in order to make the intelligent tutoring more natural.

In our discussion of the databases we have moved from strictly semantic information, to a merge between semantic and syntactic information. By taking the main verb of a sentence various semantic entries become accessible, including various elements that accompany the (frame of the) verb. For instance, we know that the verb "to seize" has an Animate Agent, an Animate Beneficiary, a Theme and a Source and that this Source generally occurs in a prepositional phrase.

Shallow Semantics for Automated Meaning Extraction. The most promising approach for automated meaning extraction lies in shallow semantics using FrameNet. By taking into account the frame of the verbs that are expressed by the student, additional information about the student's input can be extracted. Such semantic frames are schematic representations of situations involving various participants, props, and other conceptual roles, so called frame elements (see Fillmore, Wooters, & Baker, 2001). Used in combination with WordNet and a domain-specific synonym database, equivalences in underlying representation can be derived from inputs using different terminology.

The advantage of using propositional symbolic representations is that they support theorem proving and other systematic forms of inference generation. The disadvantage is that they are brittle when there is incomplete information. The statistical component is less brittle so a hybrid mechanism would still be desired. The exact pattern matching function of such a hybrid mechanism would need to be worked out.

One possible use of semantic representations would be to identify a small set of categories of *canonical reasoning packages* that are used frequently in battlefield tactics. Examples of such packages are Transporting Troops from Location A to Location B and Taking Control of a Town. Each package would have a distinctive set of Questions for the curriculum script, inference rules, and semantic composition. The hope is that these packages could be recognized on the fly and instantiated dynamically during the tutorial session.

Shallow Semantics for Question Generation. Another use of shallow semantics is to generate a large number of relevant mentor questions on the fly, which can motivate students thinking through the problem. By knowing which frame elements accompany verbs, we can anticipate questions related to student answers without understanding the student's answer thoroughly (deep semantics). For instance, if to the question "What can I make the enemy do?" a student answers "deceive the enemy," frame semantics tells us that the verb "deceive" is a transitive (can be accompanied by somebody or something undergoing the activity) or resultative (have clear, recognizable result and bring about changes) activity and has at least (a) a cause (the deceiver), (b) an animate cause, (c) an experiencer (the deceived), and an optional (d) instrument to deceive the experiencer with. Based on this information we can ask questions like:

> How would you deceive the enemy?
> Can you be more specific about who will be deceived?

Who would actually be doing the deceiving?
What would be the result of the deceiving?
What would you use to deceive the enemy?

The phrasing of the question is not important at this point. What is important is the amount of information that could be questioned once the verb and subsequently its frame are known. The generated questions can then be embellished for instance by using synonyms of their elements, such as "How would you distract the opposing military force?" and "Can you be more specific about which part of the military force will be deceived?" and "What would be the effect of drawing the enemy's attention away?"

Summary. In addition to the statistical comparison process for evaluating student inputs, conversations between intelligent tutoring systems and students could benefit from shallow semantic processing. This is particularly true for ATEC tutoring sessions where students are encouraged to *think through* the problem, rather than to solve the problem, like in AutoTutor. The curriculum script can ensure that the student stays on track, shallow semantic modules could allow the student to side track and for instance consider information at a more fine-grained level, consider information that does not directly need to be explicitly expressed to solve a problem but should be considered. However, the complexity of analysis and the issues with integrating the various components needed precluded development of the semantic processing components within this Phase II effort.

iGEN Enhancements for Dialog and Performance Assessment

A dialog extension of iGEN was created to develop ATEC and other tutoring dialog applications. In such applications, the end user interacts with the computer through a textual window that can be embedded into any type of user interface. The main extensions to iGEN are:

- Set of new operators to handle tutorial dialog.
- Extension to the metacognitive blackboard that automatically stores all information related to the dialog so that it can be accessed by some tasks or other components of the model.
- Text comparator to compare user inputs to expectations generated by the model when asking a question.

The new operators included an operator for specifying a dialog chunk, including: a leading question, a set of expectations (a single expectation, a conjunction or disjunction of expectations, or a combination of both), and subquestions with expectations. The execution engine handles comparing student input with all nested expectations and determining which have been satisfied at whatever criterion is set. It then follows the tree of subquestions, asking only the subquestions whose expectations have not been satisfied. There is another dialog operator that is for statements to be used for non-question dialog (e.g., responding to frozen expressions).

New components in the metacognitive blackboard of iGEN track the status of the dialog including the status of all questions and expectations (what has been asked and what expectations have been met) and calculate performance assessment scores.

The text comparator allows comparison of user input to expectations within a dialog chunk. Currently, this match is calculated by invoking the statistical comparison module. The mechanism allows specification of any valid iGEN instructions to be executed when an expectation is met (e.g., posting new blackboard elements, which may in turn trigger other tasks or update a student model).

Authoring/Testing Tools

A number of authoring and testing tools were developed as part of this research as described in this section.

iGEN Authoring

One authoring tool was an extension of the iGEN authoring capability. The iGEN incorporates a graphical interface for developing models. This is the Cognitive Network (COGNET) Execution Language (CEL) for COGNET Graphical Representation (CGR). The translation between CEL and CGR is done automatically and continuously by iGEN. The CGR is a visual representation of the structure of the code, using lines and nodes to show the relationships among operators and attributes. Authoring involves clicking on a node showing places where additions are possible. The authoring environment provides pop-up menus of the syntactically correct set of operators possible at any point. Inserting one is a matter of menu selection. In addition, debugging facilities test the model to determine completeness and consistency. With the addition of operators for handling tutorial dialog discussed above, the authoring environment was modified to include all the dialog operators, allowing easy development of models incorporating dialog.

Logging Tools

In order to improve the system it is necessary to identify which components failed, why they failed, areas where improvements could be made, etc. In order to accomplish this we developed various logging mechanisms in which detailed logs of the user interactions, system interactions, and interactions between different components were maintained. These detailed logs are then filtered down to a granularity desired using scripts written in PERL. Some types of filtering were used to look specifically at the dialog management, while others were used to look at the tracking of the student input in the metacognitive blackboard.

An example of detailed log followed by a filtered log to improve on the dialog management is shown below. The detailed log has a lot of information that is not useful for improving the dialog smoothness hence using a PERL script we abstract only the actual interactions between the system and the user, which is then analyzed further.

Detailed Log:

```
D ask question:
Q1 BEING_ASKED
 Q1.1 NOT_ASKED
  (or E1.1a E1.1b E1.1c)
Q What is higher command's main objective?
```

```
E current_expectations: E1.1a E1.1b E1.1c
E “E1.1a” The CG's main objective is to destroy the 2 TD before it can withdraw to Centralia.
E “E1.1b” Attack to seize OBJs HAMMER, SWORD, and SABRE in order to cover the passage of 25th AD
E “E1.1c” Control key terrain along PL IOWA and secure passage lanes through our zone to facilitate 25th AD's attack
U to defeat the 2nd mech army
D classification:  CONTRIBUTION nouns: army
D match result:  0.0028 E1.1c
D match result:  0.0037 E1.1b
D match result:  0.0105 E1.1a
D resulting question status:
Q1 BEING_ASKED
 Q1.1 BEING_ASKED
  (or E1.1a E1.1b E1.1c)
C Consider the following.
D ask question:
Q1.1 BEING_ASKED
 (or E1.1a E1.1b E1.1c)
Q What does the Op Order say about the Command General's main objective?
E current_expectations: E1.1a E1.1b E1.1c
E “E1.1a” The CG's main objective is to destroy the 2 TD before it can withdraw to Centralia.
E “E1.1b” Attack to seize OBJs HAMMER, SWORD, and SABRE in order to cover the passage of 25th AD
E “E1.1c” Control key terrain along PL IOWA and secure passage lanes through our zone to facilitate 25th AD's attack
U corps or division?
```

Filtered Log:

```
Q What is higher command's main objective?
U to defeat the 2nd mech army
C Consider the following.
Q What does the Op Order say about the Command General's main objective?
U corps or division?
```

Separate Text Only Interface for Evaluating Without Full System

To test the language components quickly and efficiently, we developed a text only interface using Python and Tool Command Language (TCL)/Toolkit (TK). This helped in ensuring that the interface components were developed in parallel with the language development. Using this tool, the language components could be exercised without the user interface or multi-user web services.

Testing Parsers

As mentioned above, different computational linguistics components need specific parser outputs for efficient operation. Evaluating different parsers is difficult without a visualization tool. In order to facilitate parser evaluation we built a web-based visualization tool using Java. The tool visually presented the parse as a tree in which a given node in the tree can be collapsed or expanded as desired. This tool also showed the parse of a given sentence using two different parsers side by side thus readily presenting the differences in both parsers.

FrameNet Evaluation Tool

As part of our investigation of semantic representation, we evaluated the use of FrameNet. In order to evaluate FrameNet we took the existing Extensible Markup Language (XML) files and put it into a MySQL database so that the frames of a given word could be readily obtained by querying the database using Java.

Lessons Learned

This section documents some of the lessons learned in our attempt to build a dialog-based tutoring system that assists military personnel in acquiring and practicing flexible tactical reasoning strategies in realistic war scenarios. It is widely acknowledged that automated comprehension and generation of natural language in dialog is an extremely difficult technical challenge even though significant progress has been made in recent years (Allen, 1995; Graesser, VanLehn, et al., 2001; Jurafsky & Martin, 2000). Furthermore, application of dialog-based tutoring to higher-order cognitive skills makes the problem even more difficult. A perfect natural language tutorial dialog system has not yet been built, but some significant achievements are immediately within grasp. The question arises, however, whether the dialog systems that can be built are sufficiently high in quality that they can be used in practice. The answer to this question is not clear-cut, but rather "Yes, No, and Maybe." This section will identify what is versus what is not feasible according to available evidence and functioning systems and the experience gained from the ATEC development.

Four Quadrant Framework

When is it feasible to build a conversational system with natural language dialog? We find it convenient to consider two dimensions when answering this question: shared knowledge and precision. To the extent conversational maxims and principle are followed (Grice, 1975), the content of learner-tutor system dialog is generally truthful, accurate, relevant, and stylistically smooth. However, shared knowledge and expected precision determine to what extent the participants (system and learner) must obey the conversational principle in order for the dialog not to fall apart.

The first dimension is the amount of *shared knowledge* between the dialog participants, in this case the learner and tutor. When the amount of shared knowledge is high, the speech participants are more picky on what each other says, so the bar is raised on what the dialog system can deliver. For instance, when talking to members of the 2nd Brigade, the amount of shared knowledge is likely to be so high that any deviation from that knowledge becomes awkward.

On the other hand, when the shared knowledge is low (a member of the 2nd Brigade and a tutor system) participants have more flexibility in the truth, accuracy, relevance and style of the dialog, because one participant does not know the other well enough. During the conversation a common ground needs to be created. Similarly, the tutor system can get away with more imperfections when the shared knowledge is low or intermediate because most learners would not know the difference between a moderate and a high fidelity tutor.

The second dimension is the amount of *precision* expected by learners or users of the system. If high precision is expected, but not delivered, users can be profoundly irritated. For example, it would not be smart to build a *conversational* system that talks through every single minute and every single coordinate of a high-risk military operation. Similarly, mathematics, statistics, and formal logic are not well suited to dialog in natural language. On the other hand, an example of low precision is chatting between officers in a bar. The information does not have to be highly precise, as long as it does not harm the conversation. The exact decisions behind an officer taking a short-cut to the cinema is not of crucial importance, whereas they might be for a commander deciding on a crucial step in a mission.

Four quadrants can be identified when shared knowledge is crossed with precision. These are depicted in Table 1. Quadrant 1 is an impossible achievement at this state of the science. The question can be raised whether this quadrant would even be desirable. It can be argued that for high precision and high shared knowledge dialog *conversational* systems are not the best solution. Quadrant 4 is entirely feasible and has been demonstrated to be successful for college students learning about introductory computer literacy. Despite its low to intermediate precision, there is strong evidence that students still learn from communicating with the intelligent tutoring system, simply because they need to think about the problem and actively articulate an answer (see Graesser, VanLehn, et al. 2001). In addition, the tutoring system keeps up the appearance that it knows what it is doing, even in cases where this is not entirely true (low precision).

Table 1

Four Quadrant Analysis

	High Shared Knowledge	Low Shared Knowledge
High Precision	**QUADRANT 1** Not feasible and sometimes undesirable. Conversation system to balance personal bank account is not recommended.	**QUADRANT 2** Feasibility is a research question. Being tested in Why/AutoTutor for conceptual physics.
Low to Intermediate Precision	**QUADRANT 3** Feasibility is a research question. A conversation system that serves as a good friend has not yet been tested.	**QUADRANT 4** Very feasible. Demonstrated in the design and tests of AutoTutor for introductory computer literacy

The jury is still out for Quadrants 2 and 3. There is evidence that even if high precision is required, intelligent tutoring systems are successful in conversations where tutor and student have limited shared knowledge (see Graesser, VanLehn et al., 2001). For Quadrant 3 less progress has been made. Some chatterbox entities are available, but their conversational skills are limited. Part of the problem is that high shared knowledge and low precision requires a sophisticated level of mixed initiative dialog (see Louwerse et al., 2002). Given that mixed initiative dialog is a complicated process (Allen, 1999), Quadrant 3 remains one of the key research areas in computational linguistics.

The ATEC is in some ways in Quadrant 2 and in others Quadrant 3. Consistent with Quadrant 3, the learners are hardly novices in the subject matter of battle tactics and there are not a large number of precise numbers, spatial indices, and time logs. Consistent with Quadrant 2, however, there are nontrivial consequences to confusing the relative locations of landmarks, paths, positions, people, weapons, and other components that need some degree of precision. But there is lower shared knowledge to the extent that the scenario envisioned by the tutor is not the same as the scenario envisioned by the student. The ATEC application fell in quadrants where there was some uncertainty as to the feasibility of building dialog systems in natural language. Additional research is therefore warranted.

Limitations of Open-ended Questions

Because battle-command reasoning rarely contains a single best response to a tactical situation, the decision was made to avoid constraining the learner with a specific solution strategy, set of tactics, or right answer advocated by the tutor. As a consequence, the tutor asked open-ended questions that addressed the eight themes and relied on the user to fill in information to answer the question. The tutor was designed not to give negative feedback on the learner's content and was not supposed to steer the user to a right answer. Instead, the learner received a set of theme-specific questions, at varying levels of specificity. This approach is quite different from AutoTutor's approach to coaching a user to answer a question (Graesser, Person, Harter, & TGR, 2001; Graesser, Hu, et al., in preparation). For AutoTutor, there is a set of expected answers, misconceptions that call for corrections, short feedback on what the learner expresses, and dialog contributions that steer the user to specific content.

Unfortunately, there were a number of liabilities to implementing open-ended questions. The tutor can never give exact feedback to the learner on the quality of their contributions. This results in the risk of the learner wondering whether the tutor is really listening and understanding. Also, because the tutor never steers the learner to a particular family of answers, there is an extremely low likelihood of the learner being a good "mind reader" and articulating what the tutor has in mind. Given the low matches between learner answers and the tutor's curriculum script content, the tutor's open-ended questions end up coming out of the blue and detached from the learner's mindset. The learner and tutor are operating under two different worlds, essentially blended monologs rather than integrated dialogs.

A better solution might have been to steer the learner to some semblance of expected content. This does not imply that a singular answer is expected. A sample of 20 experts could supply answers to the general questions, yielding a family of correct answers. If a learner's contributions matched any of the 20, then such contributions would be regarded as acceptable and would guide the tutor's feedback. If the learner's contributions matched none of the contributions, then the tutor would steer the learner on one of the more common threads. The family of answers could grow from 20 to a larger sample if new (good) answers of learners are stored; essentially, the computer learner would learn from experience (see Psotka, Streeter, Landauer, Lochbaum & Robinson, in preparation, for an example of this). After a while, it would be unlikely that contributions of a new learner would be novel.

Intelligent Conceptual Pattern Matching

Conceptual pattern matching is needed when the learner contributions are compared with the expected answers. It is widely acknowledged that brittle pattern matching algorithms, such as exact string matches or keyword matches are too brittle for going the distance in providing adequate conceptual pattern matches. In a nutshell, there is a very small proportion of successful matches, and a high error rate of missing reasonable matches. Stated differently there is a very low recall rate, where Recall = p (pattern matcher finds match|human would declare match). AutoTutor used latent semantic analysis (LSA) as a flexible conceptual pattern matcher (Graesser, Wiemer-Hastings et al., 2000; Graesser, Hu, et al., in preparation), as discussed earlier. Also as discussed earlier, ATEC used a substitute algorithm, IDF.

There are a number of approaches to improving the conceptual pattern matching of ATEC. One option is to purchase a license for LSA. A second is to improve the substitute algorithm (see next subsection). A third is to have brittle matches, but to consider the large family of 20+ expected threads, as discussed above. If a content word in a contribution matches any word in the 20+ threads, then the content word is scored as relevant. A fourth is to invent a new quantitative algorithm that operates on the family of 20+ expected threads. This is a question for future research. What we do know is that a brittle match to a small set of expected threads is entirely inadequate.

Improvement of IDF as Pattern Matching Algorithm

The response from ATEC after each student's input was not as appropriate as expected. Specifically, the system often determined the student's input was not a match and asked a follow-up question that was repetitious (see the example log in Appendix B).

To improve the IDF algorithm, the fixed threshold of 0.5 should be changed. A preliminary analysis mentioned above indicated that the threshold could be lowered to 0.4. Alternatively, the threshold could be a statistical value instead of a constant. Hu, Cai, Franceschetti, et al. (2003) have examined a similar issue when LSA values are used as estimates of similarity between documents. The further tuning of IDF methods requires obtaining more responses from students and rating them by field experts. In this way, a threshold function (sensitive to the size of the expectation and responses) can be obtained to optimize the accuracy of the conceptual pattern matching.

Improvement of ATEC as a Web Application

Currently ATEC is an application with a collection of modules that were originally designed for desktop applications, such as some AutoTutor modules and iGen Modules (BATON—the iGEN execution engine). A common issue for such migration (from desktop, single user application to web-based multi-user application) is the efficiency of memory management. For example, every time a new user is using BATON, a sizable memory is located (12 Mega Bytes (MB)) for each user until the user terminates the session. The number of simultaneous users for ATEC is limited by the amount of Random Access Memory (RAM) in the computer (about 10 users for 256 MB RAM) due to the capacity issues of certain modules.

The steps towards a truly web-based ATEC can be outlined as follows:

1. Examine how large the signature memory is for every request. This can be examined by simply comparing the size of the RAM used before a request versus during a request being accepted.
2. Perform a stress test on every module in the ATEC application. There are a lot stress test tools available. Having this test, the bottleneck modules in ATEC can be identified.
3. For those instances that requires more resources, consider separate resource modules. This allows large chunks of common memory to be loaded as read-only, to be shared by multiple users.
4. New programming technology can be introduced to integrate the modules, even rewriting some of the modules. For example, the dot-net framework works very well in the new version of AutoTutor, which is a 100% web based application.

Conclusions

Computer-based natural language dialog systems are feasible for some classes of tutoring environments, namely those in which domain knowledge is qualitative and the shared knowledge or common ground between the tutor and learner is low to moderate rather than high. The ATEC is in a borderline area. The goal of developing an ITS for a complex skill (e.g., battle-command reasoning) remains a challenging problem for the current generation of conversational systems. More promising applications are conceptual domains in which the goal is to impart knowledge.

Deep natural language understanding is still problematic. More promising approaches rely on being able to anticipate the user's possible inputs (both good answers and those indicating misconceptions). The system's response is then dependent on a process of matching the user input to one of the anticipated set of possible responses, and responding accordingly.

Some incremental changes were identified that could potentially improve ATEC. These include: changing or improving the conceptual pattern matching algorithm and re-implementing the system for efficiency as a web application. Also dialog management might have a richer state transition network or dynamic planning.

In sum, the value of the ATEC development effort is twofold. Lessons learned on technical challenges and changes required should be useful in future efforts on higher-order thinking skills, such as battle command reasoning. Technologies developed, including refinements to the tutoring architecture and underlying pedagogical approach, should readily apply to other training problems more amenable to conversational dialog.

References

Abney, S. (1996). Part-of-speech tagging and partial parsing. In K. Church, S. Young. and G. Bloothooft (eds.), *Corpus-based methods in language and speech* (pp. 118-136). Dordrecht, Kluwer.

Abney, S. (1997). *The SCOL Manual* (version 0.1b). Unpublished manuscript, Tuebingen, Germany. (can be downloaded from http://gross.sfs.nphil.uni-tuebingen.de:8080/ release/scol.ps)

Allen, J. (1995). *Natural language understanding.* Redwook City, CA: Benjamin/Cummings.

Allen, J. (1999). Mixed-initiative interaction. *IEEE Intelligent Systems. 14*(5), pp.14-16.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences, 4,* 167-207.

Baker, C. F. & Ruppenhofer, J. (2002). FrameNet's frames vs. Levin's verb classes. In J. Larson & M. Paster (eds.) *Proceedings of the 28th Annual Meeting of the Berkeley Linguistics Society* (pp. 27-38).

Chi, M. T. H., de Leeuw, N., Chiu, M., & LaVancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science, 18*, 439-477.

Chi, M. T. H., Siler, S. A., Jeong, H., Yamauchi, T. & Hausmann, R. G. (2001). Learning from human tutoring. *Cognitive Science, 25*, 471-533.

Cohen, P. A., Kulik, J. A., & Kulik, C. C. (1982). Educational outcomes of tutoring: A meta-analysis of findings. *American Educational Research Journal, 19*, 237-248.

Cohen, P. R., Perrault, C. R., & Allen, J. F. (1982). Beyond question answering. In Lehnert, W. G. & Ringle, M. H. (Eds.), Strategies for Natural Language Processing. Lawrence Erlbaum Associates.

Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the craft of reading, writing, and mathematics. In L. B. Resnick (Ed.), *Knowing, Learning, and Instruction: Essays in Honor of Robert Glaser* (pp. 453-494). Hillsdale, NJ: Erlbaum.

Collins, A., Warnock, E. H., & Passafiume, J. (1975). Analysis and synthesis of tutorial dialogs. In G. Bower (Ed.), *The psychology of learning and motivation* (Vol. IX). New York: Academic Press.

Core, M. G., Moore, J. D., & Zinn, C. (2000). Supporting constructive learning with a feedback planner. In *Papers from the 2000 AAAI Fall Symposium* (pp, 1-9). Menlo Park, CA: AAAI Press.

DARPA (1995). Proceedings of the Sixth Message Understanding Conference (MUC–6). San Francisco: Morgan Kaufmann.

Ericsson, K. A., Krampe, R., & Tesch-Romer, C. (1993). The role of deliberate practice in the acquisition of expert performance, *Psychological Review, 100(3),* 366-406.

Fellbaum, C. (Ed.) (1998). *WordNet: An electronic lexical database.* Cambridge, MA: MIT Press.

Fillmore, C. J., Wooters, C. & Baker, C. F. (2001): Building a large lexical databank which provides deep semantics. *Proceedings of the Pacific Asian Conference on Language, Information and Computation.* Hong Kong.

Fisk, A. D., & Rogers, W. A. (1992). The application of consistency principles for the assessment of skill development. In J. W. Regian & V. J. Shute (Eds.), *Cognitive approaches to automated instruction.* Hillsdale, NJ: Erlbaum.

Foltz, P. W., Gilliam, S., & Kendall, S. (2000). Supporting content-based feedback in on-line writing evaluation with LSA. *Interactive Learning Environments, 8,* 111-128.

Fox, B. (1993). The Human Tutorial Dialog Project. Hillsdale, NJ: Erlbaum.

Freedman, R. (1999). Atlas: A plan manager for mixed-initiative, multimodal dialog. *AAAI '99 Workshop on Mixed-Initiative Intelligence*, Orlando, FL.

Graesser, A. C., Gernsbacher, M. A., & Goldman, S. (Eds.) (2003). *Handbook of discourse processes.* Mahwah, NJ: Erlbaum.

Graesser, A. C., Hu, X., Person, P.,Jackson, T., & Toth, J. (in preparation). Modules and information retrieval facilities of the Human Use Regulatory Affairs Advisor (HURAA). *International Journal on eLearning.*

Graesser, A. C., Hu, X., Person, P., Jackson, T., & Toth, J. (2002). Modules and information retrieval facilities of the Human Use Regulatory Affairs Advisor (HURAA). In M. Driscoll and T. C. Reeves (Eds.), *Proceedings for E-Learning 2002: World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education* (pp. 353-360). Montreal, Canada: AACE.

Graesser, A. C., Hu, X., & McNamara, D. S. (in preparation). Computerized learning environments that incorporate research in discourse psychology, cognitive science, and computational linguistics. In A. F. Healy (Ed.), *Experimental cognitive psychology and its applications: Fetschrift in honor of Lyle Bourne, Walter Kintsch, and Thomas Landauer.* Washington, D. C.: American Psychological Association.

Graesser, A. C., Jackson, G. T., Mathews, E. C., Mitchell, H. H., Olney, A.,Ventura, M., Chipman, P., Franceschetti, D., Hu, X., Louwerse, M. M., Person, N. K., & Tutoring Research Group (2003). Why/AutoTutor: A test of learning gains from a physics tutor with natural language dialog. In R. Alterman & D. Hirsh (Eds.), *Proceedings of the 25th Annual Conference of the Cognitive Science Society* (pp. 1-6). Boston, MA: Cognitive Science Society.

Graesser, A. C., Moreno, K., Marineau, J., Adcock, A., Olney, A., & Person, N. (2003). AutoTutor improves deep learning of computer literacy: Is it the dialog or the talking head? In U. Hoppe, F. Verdejo, & J. Kay (Eds.), *Proceedings of Artificial Intelligence in Education.* (pp, 47-.54). Amsterdam: IOS Press.

Graesser, A. C., & Olde, B. A. (2003). How does one know whether a person understands a device? The quality of the questions the person asks when the device breaks down. *Journal of Educational Psychology, 95*, 524-536.

Graesser, A. C., & Person, N. K. (1994). Question asking during tutoring. *American Educational Research Journal,* 31, 104-137.

Graesser, A. C., Person, N., & Huber, J. (1992). Mechanisms that generate questions. In T. Lauer, E. Peacock, & A. C. Graesser (Eds.), *Questions and information systems.* Hillsdale, NJ: Erlbaum.

Graesser, A. C., Person, N., Harter, D., & Tutoring Research Group (2001). Teaching tactics and dialog in AutoTutor. *International Journal of Artificial Intelligence in Education, 12*, 257-279.

Graesser, A. C., Person, N. K., & Magliano, J. P. (1995). Collaborative dialog patterns in naturalistic one-on-one tutoring. *Applied Cognitive Psychology*, 9, 359-387.

Graesser, A C., VanLehn, K., Rose, C., Jordan, P., & Harter, D. (2001). Intelligent tutoring systems with conversational dialog. *AI Magazine*, 22, 39-51.

Graesser, A. C., Wiemer-Hastings, K., Wiemer-Hastings, P., Kreuz, R., & Tutoring Research Group (1999). AutoTutor: A simulation of a human tutor. *Journal of Cognitive Systems Research, 1*, 35-51.

Graesser, A. C., Wiemer-Hastings, P., Wiemer-Hastings, K., Harter, D., Person, N., & Tutoring Research Group (2000). Using latent semantic analysis to evaluate the contributions of students in AutoTutor. *Interactive Learning Environments, 8*, 129-148.

Grice, H. P. (1975*)*. Logic and conversation. In P. Cole & J. L. Morgan (Eds.). *Syntax and semantics, Volume 3: Speech acts* (41-58). New York: Academic Press.

Gratch, J., Rickel, J., Andre, J., Badler, N., Cassell, J., & Petajan, E. (2002). Creating interactive virtual humans: Some assembly required. *IEEE Intelligent Systems,* 54-63.

Grudin, J. (1983). Error patterns in skilled and novice transcription typing. In Cognitive Aspects of Skilled Typewriting, W. E. Cooper, Ed. Springer-Verlag, New York.

Harabagiu, S. M. Maiorano, S. J. & Pasca, M. A. (2002). Open-domain question answering techniques. *Natural Language Engineering, 1,* 1-38.

Heffernan, N. & Koedinger, K. R. (1998). A developmental model for algebra symbolization: The results of a difficulty factors assessment. In *Proceedings of the 20th Annual Conference of the Cognitive Science Society,* (pp. 484-489). Hillsdale, NJ: Erlbaum.

Hu, X., Cai, Z., Graesser, A., Louwerse, M., Penumatsa, P., Olney, A. & Tutoring Research Group (2003). An improved LSA algorithm to evaluate student contributions in tutoring dialog. Eighteenth International Joint Conference on Artificial Intelligence.

Hu, X., Cai, Z., Franceschetti, D., Penumatsa, P., Graesser, A., Louwerse, M., McNamara, D. S. & the Tutoring Research Group (2003). LSA: The first dimension and dimensional weighting. 25th Annual Meeting of Cognitive Science Society.

Hume, G. D., Michael, J. A., Rovick, A., & Evens, M. W. (1996). Hinting as a tactic in one-on-one tutoring. Journal of the Learning Sciences, 5, 23–47.

Jurafsky, D., & Martin, J. H. (2000). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.* Upper Saddle River, NJ: Prentice Hall.

Kintsch, E., Steinhart, D., Stahl, G. & LSA Research Group (2000). Developing summarization skills through the use of LSA-based feedback. *Interactive learning environments, 8,* 87-109

Kintsch, W. (1998). *Comprehension: A paradigm for cognition.* Cambridge: Cambridge University Press.

Kintsch, W. 2001). Predication. *Cognitive Science 25,* 173-202.

Klein, G. (1989). Recognition-primed decisions. In W. Rouse (Ed.), *Advances in man-machine systems research, 5,* 47-92. Greenwich, CT: JAI Press.

Kucera & Francis, W. N. (1967). *Computational Analysis of Present-Day American English.* Providence: Brown University Press.

Landauer, T., & Dumais, S. (1997). An answer to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review, 104,* 2111-240.

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, 25, 259-284.

Lehnert, W. & Ringle, M. (Eds.) (1982). *Strategies for Natural-Language Processing*, Hillsdale NJ: Erlbaum.

Lenat, D. B. (1995). CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM, 38*, 33-38.

Levin, B. (1993). *English verb classes and alternations: A preliminary investigation.* Chicago, University of Chicago Press.

Louwerse, M. M., Graesser, A. C., Olney, A. & Tutoring Research Group (2002). Good computational manners: mixed-initiative dialog in conversational agents. In C. Miller, *Etiquette for human-computer work. Papers from the 2002 fall symposium, Technical Report FS-02-02* (pp. 71-76).

Louwerse, M. M. & Mitchell, H. H. (2003). Towards a taxonomy of a set of discourse markers in dialog: a theoretical and computational linguistic account. *Discourse Processes, 35*(3), 199-239.

Lussier, J. W., Ross, K. G., & Mayes, B. (2000). Coaching techniques for adaptive thinking. *Proceedings of the 2000 Interservice/Industry Training Simulation, and Education Conference* [CD-ROM]. .Orlando, FL: NTSA.

Lussier, J. W., Shadrick, S. B., & Prevou, M. I., (2003). Think Like A Commander Prototype: Instructor's Guide to Adaptive Thinking (Research Product 2003-02) Fort Knox, KY U.S. Army Research Institute for the Behavioral and Social Sciences.

Manning, C. & Schutze, H. (1999), *Foundations of Statistical Natural Language Processing.* Cambridge, MA, MIT Press.

Moore, J. D., & Wiemer-Hastings, P. (2003). Discourse in computational linguistics and artificial intelligence. In A. C. Graesser, M. A. Gernsbacher, & S. R. Goldman (Eds.), Moore, J. D. (1995). *Participating in Explanatory Dialogs.* Cambridge, MA: MIT Press.

Olde, B. A., Franceschetti, D. R., Karnavat, Graesser, A. C. & Tutoring Research Group (2002). The right stuff: Do you need to sanitize your corpus when using Latent Semantic Analysis? *Proceedings of the 24th Annual Meeting of the Cognitive Science Society* (pp. 708-713). Mahwah, NJ: Erlbaum.

Olney, A., Louwerse, M., Matthews, E., Marineau, J., Hite Mitchell, H., & Graesser, A. (2003). Utterence classification in AutoTutor. In *Proceedings of the HLT-NAACL 03 Workshop on Educational Applications of NLP.*

Palinscar, A. S., & Brown, A. (1984). Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. *Cognition & Instruction*, 1, 117-175.

Pellom, B., Ward, W. & Pradhan, S. (2000). The CU Communicator: An architecture for dialog systems, *International Conference on Spoken Language Processing* (ICSLP), Beijing China.

Person, N. K., Graesser, A. C., Bautista, L., Mathews, E. C., & Tutoring Research Group (2001). Evaluating Student Learning Gains in Two Versions of AutoTutor. In J. D. Moore, C. L. Redfield, & W. L. Johnson (Eds.) *Artificial intelligence in education: AI-ED in the wired and wireless future* (pp. 286-293). Amsterdam, IOS Press.

Person, N. K., Graesser, A. C., & Tutoring Research Group (2002). Human or computer?: AutoTutor in a bystander Turing test. In S. A. Cerri, G. Gouarderes, & F. Paraguacu (Eds.), *Intelligent Tutoring Systems 2002* (pp. 821-830). Berlin, Germany: Springer.

Piepenbrock, R. (1993). A longer term view on the interaction between lexicons and text corpora in language investigation. In: C. Souter & E. Atwell (Eds.), *Corpus-based computational linguistics.* Amsterdam: Rodopi.

Polson, M., & Richardson, J. (Eds.) (1988). *Foundations of intelligent tutoring systems.* Hillsdale, NJ: Erlbaum.

Psotka, J., Streeter, L. A., Landauer, T., Lochbaum, K. E., & Robinson, K. (in preparation). Augmenting electronic environments for leadership. *Proceedings of the NATO Human Factors & Medicine Panel*, Genoa, Italy, October 13, 2003.

Rich, C., & Sidner, C. L. (1998). COLLAGEN: A collaborative manager for software interface agents. *User Modeling and User-adapted Interaction, 8*, 315-350.

Rickel, J., Lesh, N., Rich, C., Sidner, C. L. & Gertner, A. S. (2002). Collaborative discourse theory as a foundation for tutorial dialog. *Intelligent Tutoring Systems*, 542-551.

Ross, K. G., & Lussier, J. W. (1999). A training solution for adaptive battlefield performance. *Proceedings of the 1999 Interservice/Industry Training Simulation, and Education Conference* [CD-ROM]. Orlando, FL: NTSA.

Ruppenhofer, J., Baker, C. F., & Fillmore, C. J. (2002). The FrameNet Database and Software Tools. In Braasch, Anna & Claus Povlsen (Eds.), *Proceedings of the 10th Euralex International Congress*. Copenhagen, Denmark (Vol. I: 371-375).

Ryder, J., Santarelli, T., Scolaro, J., Hicinbothom, J., & Zachary, W. (2000). Comparison of cognitive model uses in intelligent training systems. In *Proceedings of IEA2000/HFES2000 Conference*, (pp. 2-374 to 372-377). Santa Monica, CA: Human Factors and Ergonomics Society.

Schank, R. & Reisbeck, C. (Eds.) (1982). *Inside Computer Understanding: Five Programs Plus Miniatures*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Schank, R. C. (1986). *Explanation patterns: Understanding mechanically and creatively.* Hillsdale, NJ: Erlbaum.

Sekine, S. & Grishman, R. (1995). A corpus-based probabilistic grammar with only two non-terminals. *Fourth international workshop on parsing technology.* Prague.

Shadrick, S. B., & Lussier, J. W. (2002). *Think Like A Commander: Captain's Edition Prototype version 1.0* (ARI Research Product 2002-02). Alexandria, VA: U.S. Army Research Institute for the Behavior and Social Sciences.

Shah, F., Evens, M. W., Michael, J. & Rovick, A. (2002). Classifying Student Initiatives and Tutor Responses in Human Keyboard-to-Keyboard Tutoring Sessions. *Discourse Processes,* 33, 23-52.

Shute, V. J., & Psotka, J. (1995). Intelligent tutoring systems: Past, present, and future. In D. Janassen (Ed.), *Handbook of Research on Educational Communications and Technology,* Scholastic Publications.

Sleeman, D., & Brown, J. (Eds.) (1982). *Intelligent Tutoring Systems.* New York: Academic Press.

Stede, M. (2003). Shallow - Deep - Robust. In: G. Willee, B. Schröder, H. C. Schmitz (eds.): *Computerlinguistik - Was geht, was kommt? Computational Linguistics - Achievements and Perspectives.* Saint Augustin: gardez!

Sweller, J., & Chandler, P. (1994). Why some material is difficult to learn. *Cognition & Instruction, 4,* 295-312.

U.S. Army Research Institute (2001). *Think Like A Commander* [Computer CD-ROM and Materials]. (Available from the U.S. Army Research Institute for the Behavioral and Social Sciences, 851 McClellan Ave, Fort Leavenworth, KS 66072).

VanLehn, K. (1996). Cognitive skill acquisition. *Annual Review of Psychology, 47,* 513-539.

VanLehn, K, Jordan, P., Rosé, C. P., Bhembe, D., Bottner, M., Gaydos, A., Maketchev, M., Pappuswamy, U., Ringenberg, M., Roque, A., Siler, S. & Srivastava, R. (2002). The architecture of Why2-Atlas: A coach for qualitative physics essay writing. In: S. A. Cerri, G. Gouarderes, & F. Paraguacu (Eds.) *Intelligent Tutoring Systems, 6th International Conference,* 158-167.

VanLehn, K., Jones, R. M., & Chi, M. T. (1992). A model of the self-explanation effect. *The Journal of the Learning Sciences,* 2, 1-59.

Voorhees, E. (2001). The TREC Question Answering Track. *Natural Language Engineering,* 7, 361-378.

Weizenbaum, J. (1966). ELIZA – A computer program for the study of natural language communication between man and machine. *Communications of the ACM, 9,* 36-45. Wenger, 1987

Wenger, E. (1987). *Artificial intelligence and tutoring systems.* Los Altos: Morgan Kaufmann.

Winograd, T. (1972). *Understanding natural language.* New York: Academic Press.

Woods, W. A. (1977). Lunar rocks in natural English: Explorations in natural language question answering. In A. Zampoli (Ed.), *Linguistic structures processing* (pp. 201-222). New York: Elsevier

Zachary, W. & Ryder, J. (1997). Decision support: Integrating training and aiding. In M. Helander, T. Landauer, & P. Prasad (Eds.), *Handbook of human computer interaction* (2nd ed., pp. 1235-1258). Amsterdam: North Holland.

Zachary, W. Santarelli, T., Lyons, D., Bergondy, M. & Johnston, J. (2001). Using a Community of Intelligent Synthetic Entities to Support Operational Team Training. In *Proceedings of the 10th Conference on Computer Generated Forces and Behavioral Representation.* Orlando: Institute for Simulation and Training.

Zachary, W., Le Mentec, J. C., & Ryder, J. (1996). Interface agents in complex systems. In C. Ntuen & E.H. Park (Eds.), *Human interaction with complex systems: Conceptual Principles and Design Practice.* Norwell, MA: Kluwer Academic Publishers.

Zachary, W. W., Ryder, J. M., Ross, L., & Weiland, M. Z. (1992). Intelligent computer-human interaction in real-time multi-tasking process control and monitoring systems. In M. Helander & M. Nagamachi (Eds.), *Design for Manufacturability.* New York: Taylor and Francis.

Appendix A

Acronyms

AI	Artificial Intelligence
APP	Apple Pie Parser
ARI	Army Research Institute for the Behavioral and Social Sciences
ATEC	Automated Tutoring Environment for Command
BATON	Blackboard Architecture for Task-Oriented Networking
C4ISR	Command, Control, Communications, Computer, Intelligence, Surveillance, and Reconnaissance
CEL	COGNET Execution Language
CG	Commanding General
CGR	COGNET Graphical Representation
CGSC	Command and General Staff College
COGNET	Cognitive Network
EMT	Expectation and Misconception Tailored
FBC	Future Battlefield Conditions
FUTURETRAIN	Training of Future Brigade Combat Team Commanders and Staffs
HWIM	Hear What I Mean
IDF	Inverse Document Frequency
I/ITSEC	Interservice/Industry Training, Simulation, and Education Conference
ITS	Intelligent Tutoring System
LSA	Latent Semantic Analysis
MB	Mega Byte
PACO	Pedagogical Agent for Collogen
PP	Prepositional Phrase
RAM	Random Access Memory
SBIR	Small Business Innovation Research
SD	Standard Deviation
SME	Subject Matter Expert
SOAP	Simple Object Access Protocol

TCL	Tool Command Language
TK	Toolkit
TLAC	Think Like A Commander
TRG	Tutoring Research Group
VIVIDS	Virtual Interactive ITS Development Shell
VM	Virtual Mentor
XML	Extensible Markup Language

Appendix B

Example Annotated Log

C: = Computer
U: = User
Italics = Annotation of computer dialog moves and internal processing
Bold = Scores from matching algorithm

C: What is higher command's main objective? [*Ask first question Q1*]
U: Seize objs hammer, sword, and sabre to cover the passage of 25AD.

[Compare to the following 3 or-expectations]
E1.1a The CG's main objective is to destroy the 2 TD before it can withdraw to Centralia. **0.0031**
E1.1b Attack to seize OBJs HAMMER, SWORD, and SABRE in order to cover the passage of 25th AD **0.9658* [satisfied]**
E1.1c Control key terrain along PL IOWA and secure passage lanes through our zone to facilitate 25th AD's attack **0.0342**
[Question answered because second expectation matched over threshold, so don't ask subquestion Q.1.1 "What does the Op Order say about the Commanding General's main objective?"]

C: Okay, I See. *[Say Acknowledgement marker]*
C: What is the plan to achieve higher command's main objective? *[Ask next question Q2]*
U: Seize objs meade, hill, grant, and lee.

[Compare to the expectations of all subquestions]
E2.1.1 The crossing sites over Stranger Creek. **0**
E2.1.2a To infiltrate the security zone, attack and seize Objective Sword. **0.0395**
E2.1.2b Conduct an infiltration and attack to seize Objective Sword **0.0494**
E2.1.2c 2nd Bde will infiltrate, attack and seize Objective Sword **0.0394**
E2.1.3a Passages lanes are open for 25 AD attack **0**
E2.1.3b Seize Objective Sword to allow 25AD freedom of movement to PL Dime **0.0429**
E2.1.3c Help 25 AD to move unhindered by enemy direct and unobserved indirect fires from PL Arkansas to PL Dime **0.0012**
E2.2a Create an assailable flank **0**

E2.2b Destroy the enemy security zone.**0**
E2.2c Destroy the security arms reserve **0**
E2.2e Clear the passage lanes to give 25th Armored Division freedom of movement from Phase Line Arkansas to Phase Line Dime. **0**
E2.2f Cover the passage of the 25th Armored Division at EENT. **0**
E2.3.1 6-502 Air Assault Infantry will conduct the infiltration. **0**
E2.3.2 At night. **0**
E2.4a 6-502 will attack and seize Objective Meade to threaten the enemy's eastern flank **0.1502**
E2.4b Seizing Objective Meade will also protect 2nd Brigade's eastern flank. **0.1222**
E2.4c Seize Objective Meade to facilitate the mounted attacks of the 4-25 and 4-81 **0.1305**
[No Expectations satisfied]

C: Lets discuss something more specific. *[Say Threshold Not Met marker]*
C: What are the key tasks you must accomplish to succeed in your mission? *[Ask subquestion with the lowest sub-threshold match Q2.2]*
U: Identify or create an assailable enemy flank; destroy the CAR; seize key terrain along PL Dime; destroy enemy forces along passage lanes.

[Compare to the nested expectations]
E2.2a Create an assailable flank **0.509* [satisfied]**
E2.2b Destroy the enemy security zone. **0.055**
E2.2c Destroy the security arms reserve 0.425
E2.2e Clear the passage lanes to give 25th Armored Division freedom of movement from Phase Line Arkansas to Phase Line Dime. **0.1288**
E2.2f Cover the passage of the 25th Armored Division at EENT. **0.0309**
[*Question partially answered*]

C: Okay, *[Say Acknowledgement marker]*
C: Can you add to that? *[Say More marker]*
U: Those are the key tasks.

[Compare to the nested expectations]
E2.2a Create an assailable flank **[satisfied]**
E2.2b Destroy the enemy security zone. **0.0231**
E2.2c Destroy the security arms reserve **0.0157**
E2.2e Clear the passage lanes to give 25th Armored Division freedom of movement from Phase Line Arkansas to Phase Line Dime. **0.0035**
E2.2f Cover the passage of the 25th Armored Division at EENT. **0.0015**
[Question not answered]

C: What are the objectives of the infiltration? *[Ask another subquestion Q2.4]*
U: Force the enemy to withdraw.

[Compare to the nested expectations]
E2.4a 6-502 will attack and seize Objective Meade to threaten the enemy's eastern flank **0.0083**
E2.4b Seizing Objective Meade will also protect 2nd Brigade's eastern flank. **0**

E2.4c Seize Objective Meade to facilitate the mounted attacks of the 4-25 and 4-81 **0.0063**
[Question not answered]

C: What is your mission? *[Ask another subquestion Q2.1]*
U: Atk and seize obj sword to allow 25AD to move unhindered by enemy fires from PL Arkansas to PL Dime.

[Compare to the nested expectations]
E2.1.1 The crossing sites over Stranger Creek). **0**
E2.1.2a To infiltrate the security zone, attack and seize Objective Sword. **0.1201**
E2.1.2b Conduct an infiltration and attack to seize Objective Sword **0.1387**
E2.1.2c 2nd Bde will infiltrate, attack and seize Objective Sword **.** **0.1184**
E2.1.3a Passages lanes are open for 25 AD attack **0**
E2.1.3b Seize Objective Sword to allow 25AD freedom of movement to PL Dime **0.3424**
E2.1.3c Help 25 AD to move unhindered by enemy direct and unobserved indirect fires from PL Arkansas to PL Dime **0.5678*[satisfied]**
[Question partially answered; subquestion Q2.1.3 answered]

C: Okay, *[Say Acknowledgement marker]*
C: Anything else? *[Say More marker]*
U: We will commence NLT 210530may.

[Compare to the remaining nested expectations]
E2.1.1 The crossing sites over Stranger Creek). **0**
E2.1.2a To infiltrate the security zone, attack and seize Objective Sword. **0**
E2.1.2b Conduct an infiltration and attack to seize Objective Sword **0**
E2.1.2c 2nd Bde will infiltrate, attack and seize Objective Sword. **0.0069**
E2.1.3a Passages lanes are open for 25 AD attack **0**
E2.1.3b Seize Objective Sword to allow 25AD freedom of movement to PL Dime **0**
E2.1.3c Help 25 AD to move unhindered by enemy direct and unobserved indirect fires from PL Arkansas to PL Dime **[satisfied]**
[Question partially answered; subquestion Q2.1.3 answered]

C: Lets discuss something more specific. *[Say Threshold Not Met marker]*
C: What is your objective? *[Ask subquestion Q2.1.1]*
U: Obj sword.

[Compare to the nested expectations]
E E2.1.1 The crossing sites over Stranger Creek **0**
[Question not answered]

C: What is your planned scheme of maneuver? *[Ask another subquestion Q2.1.2]*
U: Lead with a mech tf; then follow on with a mech and armor tf; w/ a mech tf in reserve.

[Compare to the nested expectations]

E2.1.2a To infiltrate the security zone, attack and seize Objective Sword. **0.0015**
E2.1.2b Conduct an infiltration and attack to seize Objective Sword. **0.0017**
E2.1.2c 2nd Bde will infiltrate, attack and seize Objective Sword. **0.0014**
[Question not answered]

C: How is the infiltration being conducted? **[Ask another subquestion Q2.3]**
U: The lead TF will conduct an air assault to establish contact with the main body following on.

[Compare to the nested expectations]
E2.3.1 6-502 Air Assault Infantry will conduct the infiltration. 0.1749
E2.3.2 At night. **0**
[Question not answered]

C: Lets discuss something more specific. *[Say Threshold Not Met marker]*
C: Who is conducting the infiltration? *[Ask another subquestion Q2.3.1]*
U: TF 6-502.

[Compare to the nested expectations]
E2.3.1 6-502 Air Assault Infantry will conduct the infiltration. **0**
[Question not answered]

C: When will the infiltration occur? *[Ask another subquestion Q2.3.2]*
U: 6-502 will LD at 202100may.

[Compare to the nested expectations]
E2.3.2 At night. **0.0312**
[Question not answered]