IMPLEMENTATION OF THE ANTICIPATORY PLANNING SUPPORT SYSTEM

John M. D. Hill
Department of Computer Science
Texas A&M University
College Station, TX 77843, USA
hillj@cs.tamu.edu

John R. Surdu
Information Technology Operations Center
United States Military Academy
West Point, NY 10996, USA
john-surdu@usma.edu

Udo W. Pooch
Department of Computer Science
Texas A&M University
College Station, TX 77843, USA
pooch@cs.tamu.edu

KEYWORDS

Military, Planning Aids, Decision Making

ABSTRACT

A new approach to military operations, called *Anticipatory Planning and Adaptive Execution*, treats planning and execution as a tightly coupled, single process, and replaces reaction to events with anticipation of events. The Anticipatory Planning process accounts for the chaotic nature of warfare in which possibilities appear and disappear. With new information technologies, military planners should be able to plan faster and better. An *Anticipatory Planning Support System* has been designed and implemented to provide an automated mechanism for assisting commanders and their staff in producing plans and evaluating scenarios. The *APSS* provides the human planners with an interactive visual development system to build *Plan Descriptions* around *Nodes* that represent option points in the plan and the *Branches* that represent the transitions between those option points. The *APSS* uses underlying simulations to assist in plan construction and evaluation. For testing or training purposes, an external *Stimulator* uses a controlled *Plan Description* and simulator to produce *Actual States* for use by the *APSS*. The primary goals of this implementation are to provide a common representation of the plan, facilitate the planning process, anticipate the flow of the battle, and provide a means for stimulating planning systems.

BACKGROUND

The traditional Military Decision Making Process (MDMP) focuses on developing a few friendly Courses of Action (COAs) against the "most-likely / most-dangerous" enemy COAs. This results in a very detailed plan that considers only a few branches. There is a well-known axiom that "No plan survives the first shot" - another way of saying that a branch has occurred during execution that was not included in the plan. In response, the commander and the staff transition into reactive mode.

Yet, the military is now capable of producing unprecedented amounts of battlefield information that could be used to better anticipate the flow of the battle. Military planners need a new way to incorporate this continuous feed of battle information into a simultaneous planning and execution mechanism so that they achieve and maintain "option dominance". Brigadier General (ret.) Wass de Czege has proposed a new approach called *Anticipatory Planning and Adaptive Execution* that merges planning and execution and replaces reaction to events with anticipation of events [1].

Hill, Surdu, and Pooch are implementing the anticipatory planning process in the *Anticipatory Planning Support System* (*APSS*) [2]. In this system, the user develops as many branches as reasonably possible in the initial planning process. As the operation progresses the plan is continuously updated based on actual events. Future branches that are known to be invalid are nominated for pruning and new branches are developed – well before they occur in execution. In this way, planning effort is expended on the most likely or most valuable branches.

RELATED WORK

Wilkins and Myers described the CYPRESS system that provides the framework for the creation and control of taskable, reactive agents [3, 4]. Taskable, reactive agents have two main components: an executor and a planner. The executor constantly monitors the world state for situations requiring it to take action. The planner synthesizes sequences of actions that serve as a template for later refinement by the executor. The ACT formalism [4] is the common knowledge representation that provides the means for communication and coordination between the executor and the planner.

Collier described the automated generation of courses of action as an active research area [5]. Fiebig, Hayes, and Schlabach implemented a system for generating many courses of action in the military domain using genetic algorithms[6-8]]. A coarse, low-fidelity simulation evaluated each generation of potential COAs. They also used a scheme to ensure that new courses of action generated through crossover and mutation were in fact different from the existing courses of action. FOX-GA will become part of the Command Post of the Future (CPOF) Advanced Technology Demonstration [9-11].

Kewley combined fuzzy inference systems with genetic algorithms to form a fuzzy-genetic decision optimization (FGDO) system. He applied this system to the battalion-level tactical course of action (COA) development problem [12]. A fairly sophisticated tactical simulation module is used to evaluate the outcome of proposed COAs. The performance of each COA is fed into a fuzzy preference module that produces an overall fitness for the COA that is fed back into a genetic algorithm module.

Porto, et al., demonstrated the feasibility of using evolutionary algorithms to solve platoon-level tactical problems [13]. They developed a modified version of ModSAF (called MEWS) that focused on platoon-level course of action generation in an environment where two competing platoons must encounter each other on the way to their objectives. Different goal parameters can be set, such as the importance of timely arrival at the objective, the importance of survival, or the importance of eliminating enemy tanks. An adaptive algorithm drives the behavior of one or both sides in the conflict, and the evolutionary algorithm compares possible tactics based on the success parameters. Similarly, Mason and Moffat used a number of artificial intelligence techniques to perform command and control functions within a simulation, namely data fusion, decision-making and planning, and plan supervision [14].

Atkins's GRASP planner explicitly represents multiple goals and integrating the planner into an action hierarchy called Hierarchical Agent Control (HAC). HAC handles resource arbitration and failure recovery and provides a general skeleton for controlling agents, management of sensing information, scheduling of actions, message passing, and a forward simulation process to evaluate plans.

Hill and Miller successfully combined software agents, crisp reasoning, and a genetic algorithm to resolve tactical events [15]. Their system verified the applicability of genetic algorithms to generation of options in a course of action. They restricted initial populations to those with a reasonable expectation of success by using a niching strategy based on battlefield function biases.

Surdu, Haines, and Pooch developed a system called OpSim designed to monitor the current operation [16, 17]. The result of that research verified the feasibility of their implementation of Execution Monitors that use simulation to determine the significance of differences between the execution of the operation and the plan. OpSim uses a dynamic hierarchy of rational agents, called Operations Monitors to compare the current situation with the plan. The top-level Operations Monitor informs the decision maker when the success of the plan is at risk.

Gilmer and Sullivan explored the management of multiple outcomes resulting from an event in a stochastic simulation [18] . Rather than allowing trajectories along every possible outcome, they restrict the possible outcomes to a set of representative outcomes with associated probabilities. Their more recent work includes assessment of different implementations, including a discrete event simulation approach, a tail-recursive approach, and a state duplication approach [19]. They have reported positive results in converging towards the set of representative outcomes, but caution that there are still some limitations to the approach [20]. Al-Hassan has investigated the use of measures of effectiveness to prevent the problem of discarding interesting outcomes that have low probabilities [21]. The modified system is designed to be sensitive to loss ratios while determining representative outcomes. Gilmer and Sullivan examine the ability of simulation entities to run their own instance of the simulation (called recursive simulation) to evaluate the implications of possible choices [22]. The data collected from their implementation on the "eaglet" simulation indicates that the quality of decision making by the simulated command elements is improved.

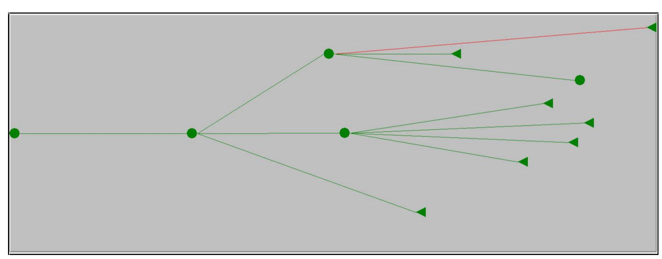

Figure 1: *Nodes* and *Branches* of the *Plan Description*

PROVIDING A COMMON PLAN DESCRIPTION

The *Anticipatory Planning Support System* relies heavily on a common description of the plan. To represent the plan, a *Plan Description* is dynamically built to manage the many tree-like branches that occur in planning and execution of an operation. See Figure 1 for a screenshot of the *Plan Description* display.

The word "common" is used to indicate that every major sub-system in the *APSS* makes use of the same *Plan Description*. The human planner uses the GUI to modify the plan. Execution of the actual operation is compared to the *PD*. The genetic algorithm operates on segments of the plan during re-planning. The simulations process segments of the plan to determine outcomes and provide evaluations.

Nodes retain the *Planned States* at specific points in the plan. These are option points, or situations where the human planner might say, "From here we can do this, or we can do that." In the *Plan Description* display, *Nodes* are shown as circles to indicate that they are expanded, showing their immediately following *Branches*. Alternatively, they are shown as triangles to represent the entire sub-tree from that *Node* and beyond.

Branches use lists of *Tasks* for the *Tactical Entities* that are present at that point in the plan to capture the transition from one option point in the plan to the next option point. They also contain a list of *Status Changes* developed by simulation of the interactions of the *Tactical Entities*. In the *Plan Description* display, lines represent *Branches*.

In the display, time is represented from left to right, with the width of the display representative of the total displayed time. When collapsed *Nodes* are expanded, the display adjusts accordingly. By expanding and collapsing *Nodes* it is possible see the entire plan structure, or to focus on small sections of the plan.

FACILITATING THE PLANNING PROCESS

One of the major goals for the *APSS* was to incorporate a common human planning thought process. This process involves determining a sequence of options based on situations that are likely to occur, and the transitions between those situations. To facilitate this thought process, the *APSS* graphical user interface (GUI) allows for rapid creation and visual control of all the steps.

The first step is to create and organize *Tactical Entities*. See Figure 2 for a depiction of the process. The *Tactical Entities* are built by selecting their attributes, causing the icon representation to change appropriately. Once the *Tactical Entity* is complete it can be dragged and dropped onto the hierarchical *Tactical Entity Tree*. The *Tactical Entities* can be moved around within the tree so long as the new ordering is consistent with certain rules. For example, friendly units must belong to a friendly headquarters and smaller units must be underneath larger headquarters.

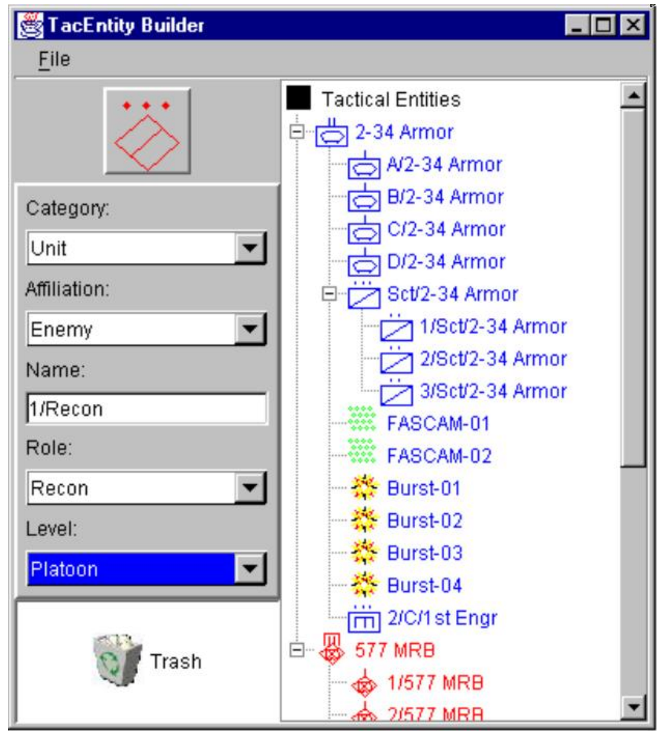

Figure 2: *Tactical Entity* Creation and Organization

The second step is the creation of a terrain model. In *APSS* terrain is implemented using a hexagonal grid. This allows for well-defined terrain effects on *Tactical Entities* and for discrete transitions. See Figure 3 for a *Hex Grid* containing several types of *Tactical Entities*. During construction of the *Hex Grid*, selecting new attributes from a control panel and clicking on the *Hex Cells* applies the changes. However, once the planning process begins, changes are not allowed.

The next step is to build the plan. The plan begins with an empty start *Node* that the user populates by placing *Tactical Entities* on *Hex Cells*. The user then creates new *Branches* and populates them with *Tactical Entity Tasks*. The *Tactical Entities* are dragged and dropped, or given instructions through right-click popup menus, to assign tasks, such as placement, remaining idle, moving, or being removed.

As each task is added the system performs bi-directional checking to ensure the task is valid for that *Tactical Entity* at that point in the plan. It checks backwards to the root *Node* and forwards through all following *Nodes*. Although this may sound computationally expensive, in practice it is not. Checking in either direction has linear time complexity with respect to the number of tasks for that *Tactical Entity*.

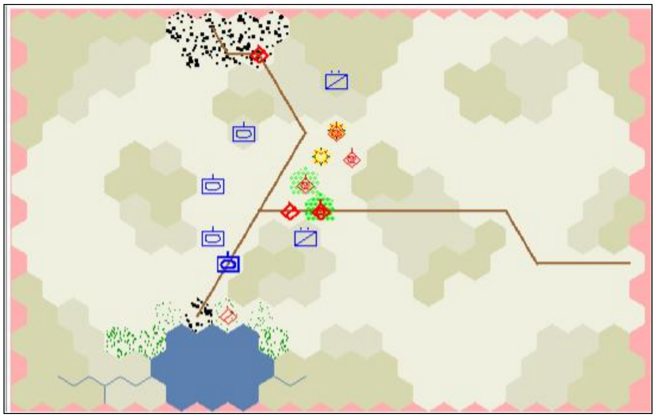

Figure 3: Terrain *Hex Grid* and *Tactical Entities*

In the backwards check it is sufficient to check the most recent non-idle task for this *Tactical Entity* to ensure the new task is consistent. In the forward check, <u>all</u> the tasks in all of the following *Branches* may have to be checked. Regardless of the direction, the system stops as soon as it finds a task that invalidates the proposed task.

As the user assigns *Tactical Entity Tasks* to construct the plan, a discrete event simulation is used in the background to determine the results of entity interaction and ensure the constructed plan is valid. The results are immediately visible in the *Plan Description* display and on the *Hex Grid* display. A discrete event simulation is also used in 'playback' mode to determine and display the actions taken within *Branches*. This is accomplished by building a simulation event list for the display and executing it in accordance with a user-selected time scale.

ANTICIPATING THE FLOW OF BATTLE

Once the *Plan Description* has been built and the operation progresses the *APSS* continuously monitors execution and re-plans ahead of anticipated events. A collaborative multi-agent system dynamically controls monitoring, prioritizes re-planning, and simultaneously builds and prunes the *Plan Description*. See Figure 4 for a depiction of the monitoring and re-planning process.

Execution Monitors are attached to *Nodes*, observe the *Actual State*, and use forward simulation to derive *Anticipated States*. The *Execution Monitor* compares the *Anticipated State* with the *Planned State* at that *Node*, determines the significance of any differences, and recommends that the *Planning Executive* (discussed below) conduct re-planning on a *Node* and/or prune certain *Branches*. In Figure 4 the time used by the *Execution Monitor* to produce its recommendation is represented by the time window labeled ①.

The monitoring and re-planning efforts are prioritized and controlled by a *Planning Executive* that centralizes the assignment of *Execution Monitors* and *Planners* to *Nodes* with the time and system resources available. The relationship of the *Planning Executive* to the *Execution Monitors* and *Planners* is depicted in Figure 5.

The *Planning Executive* maintains a balance between anticipating as many future branches to the plan as possible and constraining the total effort within system and time constraints. The *Planning Executive* examines and prioritizes the recommendations from all of the *Execution Monitors*, and allocates *Planners* to the highest priority *Nodes*. The time window ② represents the amount of time between the *Planning Executive* receiving the recommendation from the *Execution Monitor* and deciding to place a *Planner* on that *Node*. It predicts the amount of time required for re-planning each *Node* and restricts the number of active *Planners* in an attempt to ensure the re-planning will be accomplished in time.

Re-planning is accomplished by *Planners* that generate and evaluate new *Branches*. A *Branch Generator* uses a genetic algorithm and inference mechanisms that consider possible friendly or enemy actions and produces significant, representative, *Branches*. A discrete event simulation is used when Branches are created by the *Branches Generator* to determine a new *Planned State* at the conclusion of the *Branch*. The *Planner* invokes a *Branch Evaluator* to examine a *Branch*, using simulation and inference mechanisms, to determine a viability measure and an outcome confidence. Time window ③ represents this maximum re-planning time available to the *Planner*.

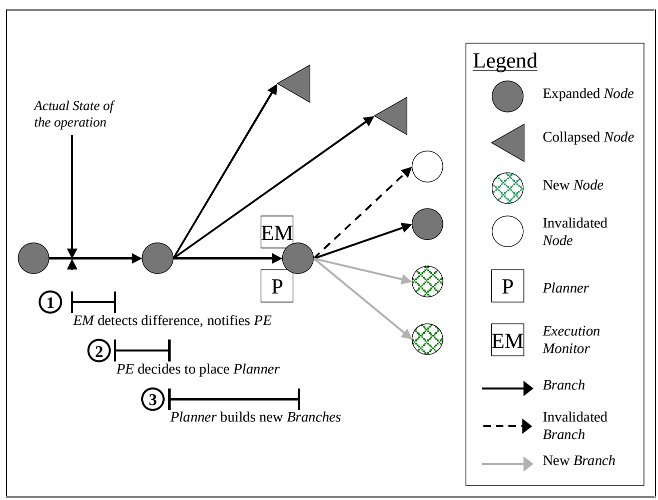

Figure 4: *APSS* Monitoring and Re-planning Process

STIMULATING PLANNING SYSTEMS

To monitor the progress of a real operation it is necessary to keep an *Actual State* of execution. This would be produced by Army Battle Command Systems [23] and represents the objective Common Operational / Tactical Picture [24]. For testing purposes, a *Stimulator* that processes a controlled *Plan Description* through a simulation produces the *Actual State*.

In testing, a *Test Executive* synchronizes the *APSS* and the *Stimulator* by sending control messages and receiving notifications from the two systems. See Figure 5 for a depiction of the test methodology. The *Test Executive* maintains the master clock and the time scale. The *Stimulator* notifies the *Test Executive* when each new *Actual State*, containing a time stamp, is ready. The *Test Executive* sends a message to the *Stimulator* allowing it to post the new *Actual State*. The *Stimulator* replies with a confirmation of the posting, and then the *Test Executive* notifies the *Planning Executive* that a new *Actual State* is available. Meanwhile, the *Planning Executive* uses the time updates to control the flow of monitoring and re-planning.

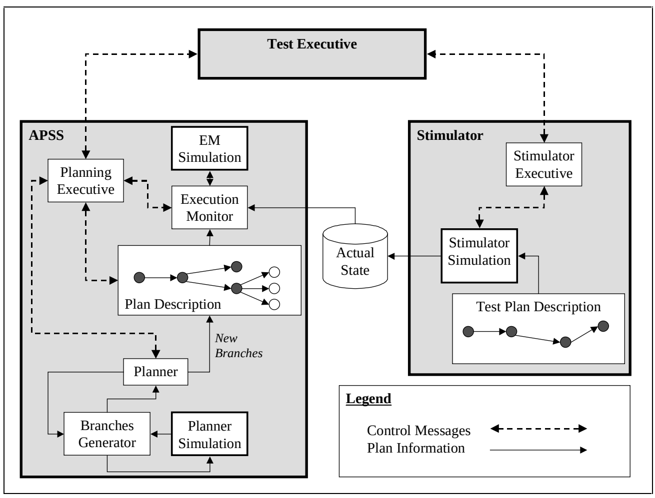

Figure 5: Testing of the *APSS* with a *Stimulator*

The *Stimulator* described above may also be useful for training purposes. In typical tactical training exercises at simulation centers, a large number of personnel are required to manipulate the systems that produce the "real" operation. Using a *Stimulator* based on the *APSS* it might be possible for a much smaller number of people to "drive" the operation. The *Stimulator* already provides a simple external interface – an *Actual State* with a time stamp. It could easily be modified to provide a much more complex interface involving exchange of *branches* or even the entire *Plan Description*.

RESULTS AND CONCLUSIONS

The common *Plan Description* works effectively in every part of the system: plan visualization, task assignment, simulation, plan building and pruning, testing and stimulation.

The *APSS* facilitates the planning process. The human planner can task units, observe interactions, build plans, and consider "what-if scenarios." In future implementations, the system would benefit from more sophisticated terrain representations and *Actual States* tied to Army Battle Command Systems.

The system also adequately anticipates the flow of battle. More testing and refinement is necessary, but preliminary results show that the *APSS* successfully monitors the situation, prunes dead *Branches*, and generates new *Branches* ahead of the real (*Stimulated*) operation.

Finally, the *APSS* is able to serve as a *Stimulator* of planning systems. Future work involves using *APSS* to stimulate planning systems other than itself.

REFERENCES

[1] Wass de Czege, H., Jr., Personal Communication (regarding Anticipatory Planning), October, 1999.

[2] Surdu, J. R., J. M. D. Hill, and U. W. Pooch. 2000. “Anticipatory Planning Support System.” In *Proceedings of the Winter Simulation Conference* (Orlando, Florida, December 10-13). 950-957.

[3] Wilkins, D., and K. L. Myers. 1994. “Integrating Planning and Reactive Control.” In *Proceedings of the Third International Symposium on Artificial Intelligence, Robotics, and Automation for Space* (New Orleans, Louisiana, November 4-6). 161-164.

[4] Wilkins, D. E., and K. L. Myers. 1995. “A Common Knowledge Representation for Plan Generation and Reactive Execution.” *Journal of Logic and Computation*, No. 5(6), December: 731-761.

[5] Collier, K. S. 1999. “Automated Decision Support Systems Enabled by Models and Simulations -- A "Leap-ahead" Technology Recommendation for the US Army After Next Time Frame (2020-2025).” In *Proceedings of the Advanced Simulation Technologies Conference (ASTC 1999): Military, Government and Aerospace (MGA) Simulation Symposium* (San Diego, California, April 11-15). SCS, 3-8.

[6] Fiebig, C., C. Hayes, and J. Schlabach. 1997. "Human-Computer Interaction Issues in a Battlefield Reasoning System." In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics* (Orlando, Florida, October 12-15). 3204-3209.

[7] Hayes, C. C., and J. L. Schlabach. 1998. "FOX-GA: A Planning Support Tool for assisting Military Planners in a Dynamic and Uncertain Environment." Published in *Integrating Planning, Scheduling and Execution in Dynamic and Uncertain Environments*, *Technical Report WS-98-02*, R. Bergmann and A. Kott, eds., AAAI Press, Madison, Wisconsin, 21-26.

[8] Schlabach, J. L., C. C. Hayes, and D. E. Goldberg. 1999. "FOX-GA: A Genetic Algorithm for Generating and Analyzing Battlefield Courses of Action." *Evolutionary Computation*, No. 7(1), Spring: 45-68.

[9] Slife, R., Personal Communication (regarding Research into Course of Action Analysis and Comparison), January 5th, 2000.

[10] DARPA. 2000. "Command Post of the Future (CPOF) Project." Available at http://dtsn.darpa.mil/iso/. Last accessed on January 27, 2001.

[11] SPAWAR. 2001. "Command Post of the Future." Available at http://www-code44.spawar.navy.mil/cpof. Last accessed on January 27, 2001.

[12] Kewley, R. H. 1999. "Automated Tactical Course of Action Development." Technical Report, Operations Research Center, United States Military Academy, West Point, New York.

[13] Porto, V. W., M. Hardt, D. B. Fogel, K. Kreutz-Delgado, and L. J. Fogel. 1999. "Evolving Tactics using Levels of Intelligence in Computer-Generated Forces." In *Proceedings of the Third SPIE Enabling Technologies for Simulation Science Conference* (Orlando, Florida, April 6-8). SPIE, 262-270.

[14] Mason, C., and J. Moffat. 2000. "Representing the C2 Process in Simulations: Modelling the Human Decision-Maker." In *Proceedings of the Winter Simulation Conference (WSC 2000)* (Orlando, Florida, December 10-13). IEEE, Piscataway, New York, 940 - 949.

[15] Hill, J. M. D., M. S. Miller, J. Yen, and U. W. Pooch. 2000. "Tactical Event Resolution Using Software Agents, Crisp Rules, and a Genetic Algorithm." In *Proceedings of the Advanced Simulation Technologies Conference (ASTC 2000): Military, Government, and Aerospace (MGA) Simulation Symposium* (Washington, DC, April 16-20). SCS, 15-21.

[16] Surdu, J. R., G. D. Haines, and U. W. Pooch. 1999. "OpSim: a Purpose-built Distributed Simulation for the Mission Operational Environment." In *Proceedings of the International Conference on Web-Based Modeling and Simulation (WebSim 1999)* (San Francisco, California, January 17-20). 69-74.

[17] Surdu, J. R., and U. W. Pooch. 2000. "Simulation Technologies in the Mission Operational Environment." *Simulation*, No. 74(3), March: 138 - 160.

[18] Gilmer, J. B., Jr., and F. J. Sullivan. 1996. "Combat Simulation Trajectory Management." In *Proceedings of the 1996 Simulation Multiconference: Military, Government, and Aerospace Simulation* (New Orleans, Louisiana, April 8-11). Society for Computer Simulation, 236-241.

[19] Gilmer, J. B., Jr. 1998. "Alternative Implementations of Multitrajectory Simulation." In *Proceedings of the 1998 Winter Simulation Conference* (Washington, DC, December 13-16). 865-872.

[20] Gilmer, J. B., Jr., and F. J. Sullivan. 1999. "Multitrajectory Simulation Performance for Varying Scenario Sizes." In *Proceedings of the 1999 Winter Simulation Conference* (Phoenix, Arizona, December 5-8). 1137-1146.

[21] Al-Hassan, S., J. B. Gilmer, Jr., and F. J. Sullivan. 1997. "A Simulation State Management Technique Sensitive to Measures of Effectiveness." In *Proceedings of the 1997 SCS Simulation MultiConference: Military, Government, and Aerospace Simulation* (Atlanta, Georgia, April 6-10). Society for Computer Simulation, 95-100.

[22] Gilmer, J. B., and F. J. Sullivan. 2000. "Recursive Simulation to Aid Models of Decisionmaking." In *Proceedings of the Winter Simulation Conference (WSC 2000)* (Orlando, Florida, December 10-13). IEEE, Piscataway, NY, 958-963.

[23] U. S. Army. 1998. *Staff Leaders Guide for the Army Battle Command System*, U.S. Government Printing Office, Washington, D.C.

[24] Department of Defense. 1998. "Defense Information Infrastructure Master Plan, Version 7.0." Available at http://www.disa.mil/diimp/diimp-2.html. Last accessed on January 27, 2001.