Proceedings of the 1994 Winter Simulation Conference
ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila

THE CORPS BATTLE SIMULATION: REMODELING THE MODEL FOR NEW MISSIONS

Hugh Henry
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, California 91109, U.S.A
hugh@diane.jpl.nasa.gov

ABSTRACT

During the mid 1980's the U. S. Army established the Corps Battle Simulation (CBS) as a standard tool for training commanders and their staffs. The CBS's architecture is typical of constructive training built during that era. It primarily models high intensity conflict in traditional theaters of operation. The technologies used in its construction reflect the wars then considered likely, the limited capability of uniprocessor computers, and third-generation distributed software environments.

Nevertheless, the Army needs training tools that support emerging missions such as Low Intensity Conflict, multi-factional scenarios, Operations Other Than War, and joint and combined training.

This paper describes the techniques used to transform the Corps Battle Simulation. To meet the needs of the late 1990's the large-scale force-on-force modeling paradigms are being replaced with representations more appropriate to the emerging world environment. The primary technology used to accomplish this transformation is called "Resolution Detailing" wherein new, more detailed and flexible functionality is selectively inserted without disturbing the overall workings of the system. Resolution Detailing permits the Army to specify focused areas for upgrade without perturbing overall system operation.

1 INTRODUCTION

The Corps Battle Simulation is used by the U. S. Army and other organizations to support the training of commanders and their staffs. While primarily oriented toward corps and division-level training the CBS is also used for multi-corps exercises. During command post exercises, the training audience (commanders and their staffs) works from tactical operations centers, often tents or trucks, executing the planning and decision process. Orders are sent to subordinate headquarters for execution. In a CBS-supported exercise, the subordinate headquarters include a computer work station suite which permits entry of the orders into the CBS system. The CBS simulation model acts on the orders and sends results back to the work stations (several hundred work stations may be used in an exercise). The results are then relayed back to the commanders and staffs using standard military procedures. A senior officer serves as exercise director, maintaining control of the computer system and ensuring that exercise objectives are met. At no time does the training audience see the CBS computers. A group of military experts provide the planning, decisions, and orders for the enemy or opposing force. This provides a realistic combat experience for the training audience.

The Jet Propulsion Laboratory delivered the original version of the CBS (then called the Joint Exercise Support System) to U. S. Readiness Command in 1987. Since that time the Laboratory delivered several significantly enhanced releases. In 1989 the U. S. Army acquired sponsorship and the system assumed its current name. Version 1.5, discussed in this paper, was delivered in the spring of 1994.

2 SYSTEM OVERVIEW

From a technical perspective, the Corps Battle Simulation is a system of commercial off-the-shelf computer hardware and custom computer software. The software may be subdivided into the game and its supporting functions. The support includes the user interface, local and wide-area networking, pre and post-processing, and interruption recovery tools. These supporting functions require over a dozen individual computer programs and 308,000 lines of source computer instructions, mostly written in the C programming language.

The remainder of this paper, however, will focus on the CBS game: the simulation model of the real-world battlefield. See Mertens (1993) for additional detail on the CBS infrastructure.

2.1 The Simulation's Model or Game

The game is the model or representation of the real-world battlefield. This includes the simulated environment (terrain and weather), troops, combat and combat support systems, and low-level tactics and doctrine. The model was originally inherited from the Joint Theater Level Simulation. At that time the model consisted of 60,000 lines of SIMSCRIPT II.5 source code. It was designed to model high-intensity conflict using aggregate-level force-on-force techniques. This means that the simulated entities are units (companies, battalions, etc.) rather than individual weapons platforms. Combat is modeled mathematically using Lanchester-type equations as described by Taylor (1981). These equations can be considered differential equations that determine attrition rates in combat. While Lanchester-type equations are widely used in combat modeling they are also subject to a variety of problems.

In a training system a particularly vexing problem is the difficulty in relating combat outcomes to combat situations. The training audience, quite reasonably, wants to know why one unit won and another lost in a given engagement. When the result has been adjudicated by a complex equation non-mathematicians often cannot understand the reasoning. What is worse, even among those comfortable with higher mathematics the validity of Lanchester-type equations as an accurate representation of combat is a matter of continual debate. Lanchester combat has been compared to "two grinding wheels working against each other until one is worn down to nothing." It cannot easily represent the large number of factors that influence combat outcomes.

In an effort to overcome some of the deficiencies of Lanchester theory the Jet Propulsion Laboratory extended the CBS combat model with a rule-based system. The basic combat model's algorithm is written in traditional procedural computer code. The extension uses a rule-based or expert system built with the OPS5 programming language. This system is called Combat Outcomes Based on Rules for Attrition (COBRA) [JPL 91]. It is a separate computer program which interacts, in real time, with the main model. COBRA consists of a set of rules and an inference engine that determines which of those rules are applicable to a given situation.

When a combat engagement takes place the game sends facts about the battle to the COBRA module. The COBRA inference engine applies rules to the facts of the situation and returns modifying factors to the Lanchester attrition algorithm. The COBRA rules are developed by a committee of operations research analysts, computer scientists, and veteran combat-arms officers. The total rule base includes 209 rules. An example of a simple rule is:

* If the terrain is water-based agriculture and the temperature is greater than -5 degrees Celsius the environment favors infantry over armor.

Each time COBRA analyzes a battle the rules applied are recorded. Exercise directors can review this record if questions arise on the outcome of simulated battles. Most users find the rule trace much more understandable than traditional attrition algorithms. Krueger (1992) reports that military experts find the results realistic. After COBRA was fully implemented we found that it facilitated functional extensions to the model. This will be explored in Section 4 below.

3 EVOLUTION OF THE SIMULATION

During the years since the original delivery the CBS game grew from 60,000 lines of source code to 390,000 and from 453 subroutines to 3,100. This growth is primarily due to two factors. The first is the addition of numerous functional enhancements, i.e. extending the model to represent additional facets of the battlefield. The second is the user's desire for increasing detail or resolution on battlefield facets already modeled. A general bias exists that more detailed models are more realistic. If the model is properly constructed this is probably true. Nevertheless, adding detail and new functionality can pose significant challenges for both modelers and software engineers. All changes involve cost, technical risk, potential

difficulty for users, and potential negative impacts on system performance.

The size of the CBS game software imposes a considerable burden on its modification. The structure of the SIMSCRIPT II.5 language dictates that most of the data is global. In other words, any part of the game software can potentially affect any other part. A large potential for unintended side effects exists with any change. The software engineer thus tries to minimize any potential breakage. Additionally, new functionality generally means new code and new data to be processed. This means an additional burden on the computer. It should be noted that the mainframe too has been continually upgraded. The original game ran on a VAX 11/750. Most sites now use the VAX 6420; more than an order of magnitude increase in processing power. Every software change increases the likelihood that the machine will be unable to keep up with the real time pace of the exercise.

The user must also be considered in any changes. The CBS has been in wide use in the Army for several years. A large installed base of users exists in the U.S., Europe, and Asia. Therefore we attempt to minimize any obvious effects of changes. Finally, all changes cost money and generally the bigger the change the more expensive. For all these reasons we attempt to implement the minimal change necessary to meet the Army's training objective. To accomplish this we apply a set of techniques we call Resolution Detailing. Through Resolution Detailing we extract maximum training value from the model while minimizing the costs and risks described above.

4 RESOLUTION DETAILING PSYCHOLOGICAL OPERATIONS

For CBS version 1.5 we were directed to add a Psychological Operations (PSYOP) functionality. Previously there was no PSYOP model in the game. The challenge was to insert a capability that would 1) support the training objectives of teaching commanders and their staffs to effectively use PSYOPs and 2) not perturb the existing functionality in the game. Through consultation with military experts it was determined that leaflets and loudspeakers were the primary psychological weapons to be modeled. This discussion will be limited to leaflets for the sake of brevity.

One approach would have been to create a new entity or type, such as a "leaflet bundle," within the game. This, however, would have required code to specify how the numerous entities in the game could affect the leaflets and how the leaflets would affect them. Instead we took two existing types: artillery ammunition and air-to-surface dumb bombs and created new instances of these types. Large blocks of logic that deal with these types were reused with minimal modification. Data tables describing the weapons set the probabilities of kill or damage to zero. New logic was required to specify how this new, non-lethal weapon influenced the battlefield.

The exercise director can set each unit's susceptibility to psychological operations. When a sufficient quantity of leaflet munitions land within a specified range of a unit that is susceptible, the COBRA module is notified that the unit is suffering the effects of PSYOP. The COBRA rules then modify the unit's morale which in turn modifies its effectiveness in combat operations. The effects of the psychological warfare on the unit disappear after a given time.

5 CONCLUSION

As one of the Army's major training tools, the CBS must change and adapt to the Army's new and continuing roles and missions. At the same time, large software systems exhibit a significant inertia to those who wish to modify them. Such systems are difficult to modify because any change may have surprising and unpredictable side effects. The techniques of Resolution Detailing permit such modification while minimizing the possibility of unwanted effects.

6 ACKNOWLEDGEMENTS

The author would like to thank Major Tim Metivier of the National Simulation Center and Major Larry Harrison of Simulation, Training and Instrumentation Command for their support and technical guidance in the development of CBS version 1.5.

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the United States Army Simulation Training and Instrumentation

Command (STRICOM), Orlando, Florida, through an agreement with the National Aeronautics and Space Administration.

Reference herein to any specific commercial product, process or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

7 REFERENCES

Jet Propulsion Laboratory, 1991. *COBRA User's Guide*, Internal Document D-7856 Rev. A, Pasadena, California.

Krueger, John L., 1992. Pitfalls in Combat Simulations, *Military Review*, June, 1992 pp. 20 - 25.

Mertens, Sherry, 1993. The Corps Battle Simulation for Military Training, *Proceedings of the 1993 Winter Simulation Conference*, ed. G.W. Evans, M. Mollaghasemi, E. C. Russell, W. E. Biles, 1053-1056. Institute of Electric and Electronic Engineers, Piscataway, New Jersey.

Taylor, James G., 1981. *Force-on-Force Attrition Modeling*, Military Operations Research Society of America, Arlington, Virginia.

AUTHOR BIOGRAPHY

HUGH HENRY is supervisor of the Simulation Software Group in the Information Systems Development and Operations Division of the Jet Propulsion Laboratory. He received a B.S. in urban studies from Wright State University in 1981 and a M.S. in computer science from Azusa-Pacific University in 1993. His research interests include the application of software technology and software engineering to simulation development.