Proceedings of the 1995 Winter Simulation Conference
ed. C. Alexopoulos, K. Kang, W. R. Lilegdon, and D. Goldsman

ADVANCED SIMULATION, BATTLE MANAGERS, AND VISUALIZATION

Joseph J. Molitoris
Thomas D. Taylor

Center for Naval Analyses
4401 Ford Avenue
Alexandria, VA 22302, U.S.A.

ABSTRACT

Advanced modeling, visualization, and simulation is of growing utility for the military, e.g., the Navy and Marine Corps. Potential uses are for situational awareness displays, analysis, assessment, doctrine, mission support, rehearsal and replay, tactics, and training. Advanced technology platforms can be assessed using simulation. New sensors and systems can be evaluated. A battle or simulation manager (BM or SM) is a flexible piece of software that commands, controls, and displays the simulation of a group of objects. It is an environment or toolkit for analysis, assessment, training, and wargaming. Current doctrine and tactics can be tested and missions planned with this type of software. The objects (military platforms) propagate and interact in space over the course of time through well-defined object-oriented processes. Features of a good BM include graphical two and three dimensional displays, database access, sensor and weapon models, and networking (via DIS or Distributed Interactive Simulation). A BM should be easy to use, fast, useful to the user, readily available, and widely used. It should be portable cross computer platforms. *Today, there is a wide assortment of BMs, none of which satisfies these criteria.*

1 OVERVIEW

Some of the difficulties with the current generation of software can be fixed with a better design that makes use of many of the parts of advanced modeling and simulation -- visualization, object oriented programming, object oriented databases, and distributed plus parallel processing -- to achieve flexibility, interoperability, and scalability. We present this design, our current investigations of it, and recommendations for future development.

An essential feature of a good battle manager is flexible command and control, which enables the user to easily transition between different control modes. This flexibility is necessary in order to combine live, constructive, and virtual simulations in a seamless way. Hardware limitations (e.g., processing power and network bandwidth) must also be overcome in order for a BM to operate in real time (or faster) at sufficiently high fidelity and with non-trivial multiwarfare scenarios. Faster than real time operation is important for accumulating reasonable statistics. Parallel and distributed processing are ways to achieve the goal of simulating up to a hundred thousand entities (platforms, systems, weapons, etc.). The results of the simulation need to be compared to simple analytic models and military operational data in order for the simulation to be credible and validated. We recommend a commercial off the shelf approach to successfully evolve modeling and simulation -- e.g., battle manager technology -- for military and other use.

2 WHY WE DID THIS RESEARCH

This research effort began because the Fleet Operational Simulation Project (FOSP) at the Center for Naval Analyses (CNA) needed a battle manager as the centerpiece of our military modeling and simulation (M&S) effort (Molitoris et al. 1994). ONR and OPNAV jointly sponsored the FOSP for the past two years. Further, there did not exist at that time (early 1994) any reasonably complete and useful survey or study of simulation managers (SM) or battle managers (BM). Many fragmentary surveys and studies are available (Bailey et al. 1994, Marvel and Watts 1994, Quinn and Feher 1994, Weatherspoon 1992). Research to date is lacking both in completeness (looking at only a few of the many codes) and also in contact with reality (one should see and run the BM/SM to best know its capabilities).

Our research is more complete and grounded in reality. We took a hands-on approach and worked with and modified several BMs: the commercial product STAGE for Simulation Toolkit And Generation Environment (Virtual Prototypes Inc. 1994), the new product TRAXX (Famic Technology Inc. 1994), and the in-house code MWARS for Maneuver Warfare Analytical Research System (Parsons 1994). We also looked at many other BMs (Molitoris 1994) including AWB (Analyst WorkBench), CAAM (Composite Area Analysis Model), EADSIM (Extended Air Defense SIMulation), FLAMES (Force Level Analysis and Mission Effectiveness System), JMASS (Joint Modeling And Simulation System), MARS (Multiwarfare Assessment and Research System), and SAF (SemiAutomated Forces programs such as MODSAF, NAVYSAF, etc.). By our count, there are in excess of fifty BM/SM/war game products out there (Molitoris 1994). Existing products are being upgraded and modified; new BMs are being developed.

In this paper, we address the important questions of what a BM is and what it should be. We believe that the heart of a good BM is flexible command and control. A battle or simulation manager for military multiwarfare use should also use object oriented programming (Taylor 1990), object oriented databases (Elmasri and Navathe 1994), and distributed plus parallel processing (Bauer 1992 and Foster 1995) to be successful.

3 WHAT IS A BATTLE MANAGER?

We now discuss the elements of our simulation architecture (Figure 1).

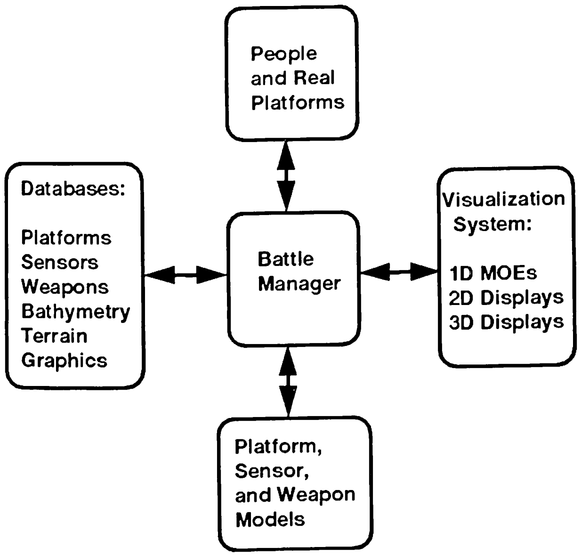

Figure 1: Simulation Architecture

We define a good simulation manager as a flexible piece of software that commands, controls, and displays the simulation of a group of objects using well-defined processes. The objects propagate and interact in space over the course of time. If the objects are military battle-related platforms, then the simulation manager is called a battle manager. The BM is a critical piece of technology that can determine how useful M&S will be to the military. There are many possible features of a battle manager including the reception of events from and broadcast to other battle managers.

Very few people understand what a battle manager is and should be. Some people refer to a battle manager as a computer generated forces (Lee and Fishwick 1994) program or a modeling environment. There is a proliferation of battle managers at numerous laboratories and centers (Bailey et al. 1994 and Molitoris 1994); *none of them has the right set of capabilities for advanced distributed multiwarfare modeling and simulation.*

4 WHAT A BATTLE MANAGER SHOULD BE

What is needed in a battle manager are three things:

1. Ease of use (user and integrator friendliness) including speed of use
2. Useful to the user (e.g., CNA, the Navy, laboratories, warfare centers, and DoD in general)
3. Easily available and widely used.

These are the same criteria for an infrastructure, which also must be inexpensive for each user. Another common criterion is that a good simulation manager serves as the foundation for potentially many uses by diverse users (Department of Defense, Department of Transportation, industry, education or training, etc.). A good simulation manager could thus be a dual use technology.

No battle manager currently satisfies these criteria.

5 FLEXIBLE COMMAND AND CONTROL

A battle manager could satisfy these criteria if it was designed with:

- Flexible command and control
- Common protocol
- Common standards.

This would enable a battle manager to be connective, interactive, and interoperable.

Flexible command and control should be the heart of the system. By this, I mean that the software can easily transition between diverse modes of control.

A flexible battle manager should also be easy to change and maintain. A good battle manager enables the user to maximize the benefits of interactive graphical simulation (Rheingold 1991).

The common protocols and standards are important because of the difficulty with which current simulations communicate and interact with one another. The problem of software portability -- making the battle manager run on multiple platforms -- is secondary in comparision to the national need for standard databases and communication protocols. Two important requirements, which we discuss further later in this paper, are that a BM be interoperable (e.g., via CORBA) and scalable to enable distributed and parallel processing. A further high level national problem is that M&S is stovepiped: most centers and laboratories have their own disconnected simulation efforts.

No battle manager currently is very flexible: most current systems are dumb and inflexible.

6 ARCHITECTURE

The concept of an architecture is important in M&S. However, some care is needed because there are computer architectures (Dowd 1993), software architectures (Buschmann, and Meunier 1995), and simulation architectures. Furthermore, it is important to know in evaluating a system whether this is the current architecture or that planned for years in the future. Many M&S efforts are multi-year multi-million dollar programs with limited currently available products. Software lives in the code that programmers create so that it is always changing.

A high level simulation architecture provides context for system development, integration, and use. We showed the objects of our current system in Figure 1: analysis tools (MOE routines), databases, people, sensor models, visualization tools (graphics), and the battle manager (as the centerpiece). This is our simulation architecture.

7 SOME REQUIREMENTS

One should have the ability to run the BM with no interactions, some interactions, or all interactions; this is important for adjusting how fast the BM executes. With a large number of platforms or replicates, the number of interactions may need to be limited in order to get good performance. Graphics output, the use of script, and data logging will also slow the system down.

One would also like the BM to:

- Perform quickly (using distributed and parallel technology)
- Be interoperable (using OOP or Object Oriented Programming technology to link with other codes)
- Be portable (coded in C/C++/X-Motif, for example)
- Be well written and documented.

The models and methods of control should connect with an ease like that of wall sockets.

No battle manager currently has this flexibility, but we have demonstrated with our research how the job can be done right.

8 ADVANCED SIMULATION: FROM VISUALIZATION TO VIRTUAL REALITY

The various pieces of advanced simulation

- Object oriented programming
- Object oriented databases
- Distributed and parallel processing
- Variable fidelity models and databases
- Visualization
- Knowledge based systems

point the way to a future multiwarfare simulation capability. We have integrated some of the pieces and performed operational UUV (Unmanned Underwater Vehicle) and other simulations at CNA. Our research determines possible requirements for next generation simulation systems and points the way to the effective utilization of advanced simulation in acquisition and assessment.

Visualization (Earnshaw and Watson 1993) in its many forms -- tables, xy plots, 2D contour graphs, 3D perspective pictures -- is common in the hard sciences (Bonasera et al. 1994). But it has only recently matured into an independent field with potential for many applications including military command, control, and intelligence. Visualization is about understanding, not just pretty pictures: the graphics is the medium and the content is what is important.

The right visualization can be very useful in situational awareness displays and other military command post applications. A child can easily recognize the difference between a circular icon and a 3D image: the image conveys more information quicker and makes better use of the eye's high bandwidth. Many of the battlefield problems in warfare relate to information flow and rapidly fusing information into correct timely knowledge in support of appropriate decision making. Lack of the right understanding of a situation can be deadly resulting in accidents, friendly fire, and troop losses. Visualization can thus potentially contribute to reducing the fog of war.

Three dimensional graphics models were purchased by us from a company (Viewpoint Datalabs 1994) and also built in-house, as needed. Both an adequate polygonal count and good texture files are important for a high quality visual presentation. We use a twelve processor SGI Onyx with a Reality Engine^2 for computation and display.

The current state of the art in computing allows us to just begin to address the complexities of visualizing and simulating the real world in real time or faster. The continuing growth of computer technology in speed (processor MHz), power (processor GigaFLOPS), memory (shared or distributed GigaBits), and cost effectiveness will soon allow the visualization of non-trivial scenarios with 10,000 to 100,000 objects.

As a first step, we connected the STAGE battle manager to the 3D Paradigm Vega simulation package (Paradigm Simulation Inc. 1994) to produce meaningful visual results. This connection of 3D graphics to a simulation gives the graphics content; the product is thus not merely a picture show. The combination of data and graphics (from which we have cut rudimentary videos) is an elementary multimedia application.

Virtual reality methods (Rheingold 1991) promise to further enhance the level of interactivity in any graphical simulation. We make use of VR to a limited extent in that we build and use the 3D synthetic environment (terrain and underwater) for our simulations using standard Defense Mapping Agency databases (specifically, Digital Terrain Elevation Database and Digital Bathymetry DataBase). Other databases of known format could also be used. We could also easily attach a boom, head mounted display, or other VR immersion device to our architecture. One interesting further application of VR methods may be found in (Block, and Stytz 1994).

9 COTS APPROACH

We believe that a commercial off the shelf (COTS) approach is best to successfully evolve military M&S to the future. For example, the battle managers STAGE (from Virtual Prototypes Inc. of Montreal, Canada), FLAMES (from Ternion Corp. of Huntsville, AL), and TRAXX (from Famic-Technology of Montreal, Canada) are COTS products. Another evolving commercial tool is the product OMNI from Autometric, Inc. of Colorado Springs, CO.

One problem we encountered is that neither these products nor military laboratory or warfare center battle managers fully satisfied our needs and requirements. (TRAXX simply was not available to us at the time of our initial need and writing this article and we were unaware of the existence of FLAMES or OMNI.) FLAMES and OMNI are the only existing U.S. COTS BM products. STAGE was then a first generation BM (which is now evolving further), whereas TRAXX was an under-development second generation product. Thus, we began evolving elements of a system FLEXX based on experience with STAGE and the in-house product MWARS (Parsons 1994). We did this by integrating available COTS software elements and reusing existing in-house code.

The goal is to be object oriented, parallel, and capable of distributed simulation. We used SGI Indy, Indigo^2, and Onyx computers for the basic hardware. COTS software elements include the Object Store object oriented database system (Object Design 1994), Multigen Inc. 3D model building software (Multigen Inc. 1994), Paradigm Vega 3D display product, and Mak Technologies (of Cambridge, MA) VR-Link distributed simulation software (Mak Technologies 1994).

The GUI and graphical displays are an important part of the architecture where no existing battle manager excels. The graphical displays need to be flexible and allow useful 2D and 3D views of the simulation and MOEs. The presentation of simulation statistics needs to be appropriate to the user's needs (Kuljis 1994). The GUI needs to be intuitive and user friendly.

We initially used UIM/X (User Interface Motif for X Windows) and XMOVE (X Meter Object Visualizing and Editing tool) to investigate a basic GUI and graphical display (Bluestone Inc. 1993 and 1994) without much success. We then explored the use of the free programs Tcl (Tool command language) and Tk (Toolkit for X Windows) for GUI creation with some limited return on our investment.

We also explored the use of application development toolkits like the Galaxy product to develop display windows; this approach looks very promising (Visix Software Inc. 1995). An alternative approach -- in use by the company Famic -- is to develop everything using SGI tools like Inventor and Iconsmith; Famic has developed a Viewer Toolkit Library that may satisfy some of our needs (Famic Technology Inc. 1994). Ternion Corp. has evolved their own GUI and graphical displays (both 2D and 3D) using SGI GL (Graphics Language) and their own toolkit.

10 OBJECT ORIENTED TECHNOLOGY

Object oriented technology -- specifically object oriented analysis (Booch 1994), object oriented design, object oriented databases, and object oriented programming -- is one of the new tools that promises to help in the development and integration of software for advanced modeling and simulation (Bailey 1994, Chervi, and Sautereau 1994, Chang, and Jones 1994).

We give an idea of the object oriented framework a simulation manager should have with a simple scheme. See Figure 2.

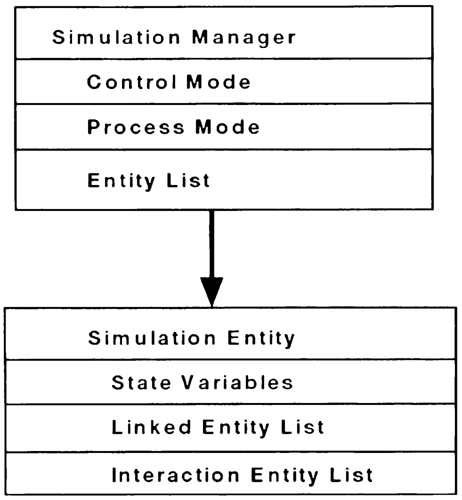

Figure 2: Control Structure

The main object is the simulation manager. Its important control modes have already been discussed above. Note especially that one simulation manager can be controlled by or interact with another. This allows for red/blue evaluations and games. All SMs can be on an equal footing (as they are in DIS) or there can be a master SM that determines ground truth for all the slaves (distributed versus central control). The communication with other SMs can be thus with none of them (stand alone operation), some, or all that are available. Further, one can either run in a demonstration mode with graphical display or a batch Monte Carlo mode to accumulate good statistics. When running in demonstration mode, one wants to be able to stop the simulation, change something, and resume from the break point (what-if analysis).

The types of processes we have in mind are: pre-processing (creation, initialization, replication), run-time processing (interactions, actions, interpretations, propagation), and post-processing (averaging and computing measures of effectiveness). Initialization includes determining links and lists as well as retrieving environmentals and setting up terrain and bathymetry displays. Note also that the simulation manager maintains a list of the entities in the simulation; this entity list could easily be divided up into moving and non-moving platform lists for speeding up interaction calculations.

All of the platforms (e.g., helos, air cushion vehicles, planes, ships, subs, tanks, unmanned vehicles), weapons, and sensors are entity objects. Multiple objects (e.g., MCACs) can be mounted on a mother object (e.g., LHD ship). Multiple sensor objects (e.g., detect and classify sonars) can be attached to an appropriate object (e.g., MCM ship). Figure 3 displays a basic object. Each simulation entity has position, velocity, and acceleration state variables, as noted above, plus other possible variables (rotational angles and angular velocity, for example).

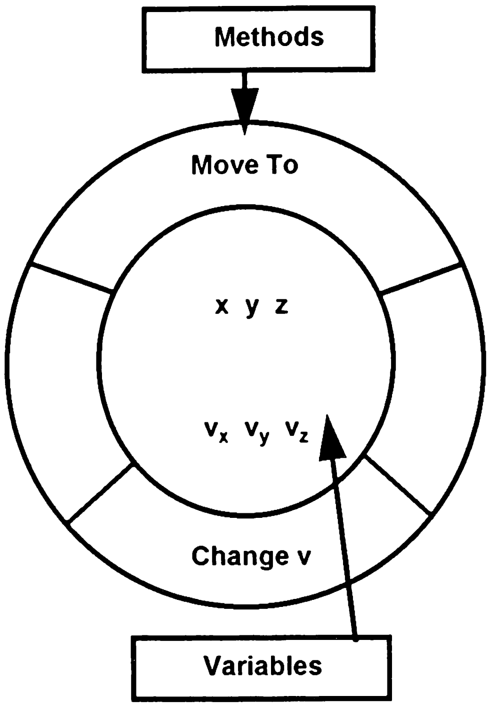

Figure 3: Anatomy of an Object

11 OBJECT ORIENTED PROCESSES

Along with the basic objects, the actions or processes that the objects act according to are very important. Figure 4 shows an example: the MCM process.

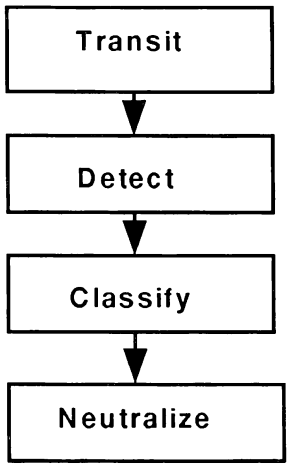

Figure 4: Schematic of MCM Process

The countermine process -- hunting and sweeping -- is of considerable importance for our MCM simulation effort. In hunting, for example, when a mine is detected one may want to change course to avoid it or even stop the ship so that one might deploy an UUV to neutralize the mine.

Modularity or object orientation is also an important property for distributed and parallel computing in order to address both the complexity of a battle manager and inter-processor communication issues.

12 PARALLEL PROCESSING

Parallel processing has been used for decades in high cost R&D (nuclear weapons design, etc.); now it has become affordable and available for more general use. Today, personal computers and workstations are beginning to have more than one processor! In the common serial processing approach, the computer performs tasks sequentially. The tasks can be elements of a simulation process, entire iterations of the simulation, or different algorithmic parts of the simulation. If each of the tasks requires a time dt and there are n tasks, then a total CPU time of n dt is needed in the serial approach.

In the parallel approach, a computer makes use of several processors to perform tasks in parallel as in Figure 5. If there are n processors, then n tasks can theoretically be performed in time dt; this is called the concurrency property (Bauer 1992, and Foster 1995). Parallel processing can thus help fulfill the requirement for a battle manager to run in real time or faster. This is a demanding requirement since multiwarfare simulations -- or even single warfare simulations -- with many platforms, complex databases, and high fidelity sensor models need lots of compute power (on the order of one to ten GigaFLOPS).

The technology of parallel processing, for example going from one to twelve processors, enables a potential twelve fold reduction in computing time. A parallel-capable program uses several collocated processors to run quicker. The use of tens of processors to hundreds or thousands of processors (as in the high performance computing initiative for massively parallel processing) enables even greater reductions in computing time. A desirable BM feature is thus scalability, the resilience of the software to increased numbers of processors on PCs, workstations, etc.

13 DISTRIBUTED PROCESSING

To some degree, distributed computing is like parallel computing. We want a BM to be able to run on and make use of many computers (processors) at one time. However, issues such as reliability, security, and heterogeneity are unique to distributed computing vice parallel computing (Foster 1995).

The locality property is another important issue in both distributed and parallel computing. A collection of nodes (processors) are connected in a network. Each node accesses its own local memory (read/write operations) and sends/receives messages over the network. A BM needs to have more local data accesses that remote accesses if it is to run efficiently and satisfy the locality property.

By distributed processing, we thus mean much more than just the current Distributed Interactive Simulation approach, which is rather limited. A distributed-capable program should be able to use several distributed computers to run quicker. The distributed

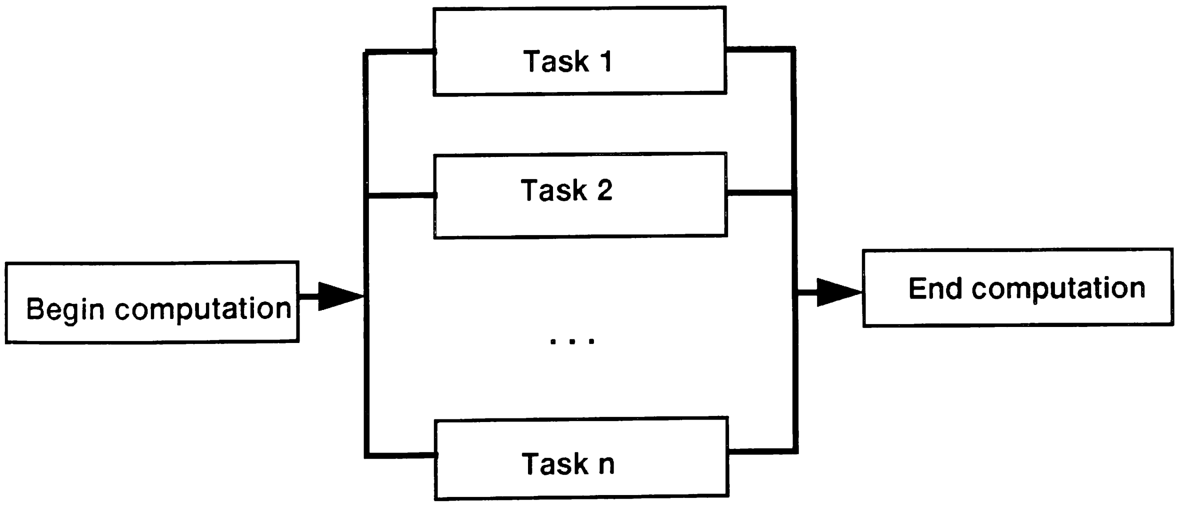

Figure 5: Parallel or Distributed Processing

processing architecture can be thus very much like the parallel processing case (Figure 5) with the distinction that distributed computers are linked by networks and do not share memory and disk resources; they may however share databases and tasks via messages and data exchange over high bandwidth networks..

Another distinct way in which distributed computing can help is thus via accessing distributed databases and models. One wants to have as many useful models and databases available on-line in a simulation facility as possible. However, if the most current or best object is readily available via a network to a center or laboratory, then distributed access is a possibility. The goal is for the user to have timely access to national databases via the evolving Defense Information Infrastructure (DII) so that it does not take weeks to months to get needed data.

The important national networks that are needed for non-trivial distributed simulation are not yet in place; for example, the bandwidth and performance of the current Defense Simulation Internet (DSI) must be improved. Further, the various centers and laboratories need to be connected in a cost-effective way to an appropriate simulation superhighway. DIS and other protocols are still evolving. In the future, we will move beyond DIS/DSI to more capable protocols and networks.

High bandwidth networking technology (ATM for example) needs to be put in place (with standard protocols) and available on a more cost-effective basis for distributed processing to be more successful. Networking and communication (Fitzgerald 1993) is even more of a problem for mobile platforms.

14 NEXT GENERATION BATTLE MANAGERS

The motley zoo of battle managers is in a state of rapid evolution, both the commercial products FLAMES, TRAXX, and STAGE and the numerous government products such as CAAM, EADSIM, JANUS, JMASS, MODSAF, and MARS. Work is also in progress at various program offices, companies, and centers to evolve other systems.

Where all this will go in the next few years is anyone's guess. Coordination and direction from above could help if the best capabilities of each separate BM were combined to have a flexible and powerful system. But the existing competition between BMs can also help to evolve the next generation system. Perhaps there will be a clear winner in the commercial or government arenas. We believe that the architecture is very important and that it must be a flexible one in order to satisfy the widest range of users. Figure 6 repeats our simulation architecture figure (see Figure 1 above) from the point of view of the battle manager interacting with databases, graphics, and object oriented models and processes via diverse interfaces and networks.

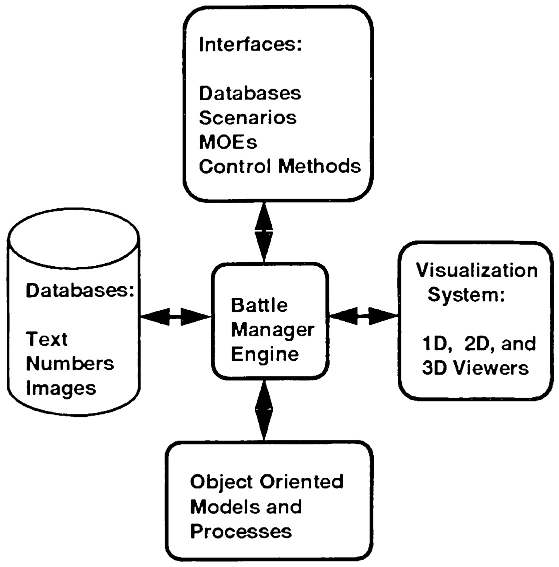

Figure 6: Simulation Architecture.

A viewer is a window to the simulation world for viewing the values of parameters or representations of the simulation and any connected databases. A viewer is a visualization tool. An interface goes beyond a viewer and allows the user to interact with the simulation, to change the value of parameters (speed, heading, etc.) either before or during a simulation.

In Figure 7, we concentrate on a flexible structure built from the ground up on database, graphics, and network abstractions. The foundation libraries are object oriented libraries for all the different software tasks; this enables the reuse of existing in-house, government, and commercial code. This structure enables the battle manager to be interoperable and portable cross platforms (UNIX, WINDOWS, MAC).

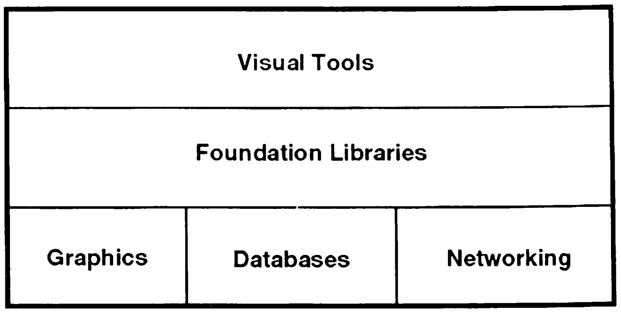

Figure 7: Flexible Structure

REFERENCES

Bailey, M. 1994. Object Oriented Simulation Pictures. In *Proceedings of the Summer Computer Simulation Conference*, 67-76. Society for Computer Simulation, San Diego, CA.

Bailey, S., et al. 1994. Modeling Architectures. Center for Naval Analyses, Alexandria, VA.

Bauer, B. 1992. *Practical Parallel Programming*. New York: Academic Press.

Block, E., and M. Stytz. 1994. Tools for Commander and Staff Training. In *Proceedings of the Military, Government, and Aerospace Simulation Conference*, 43-48. Society for Computer Simulation, San Diego, CA.

Bluestone Inc. 1993. Getting Started with UIM/X Software. Mt. Laurel, NJ.

Bluestone Inc. 1994. XMove Programming Manual. Mt. Laurel, NJ.

Bonasera, A., F. Gulminelli, and J. Molitoris. 1994. The Boltzmann Equation at the Borderline. *Physics Reports* 243: 1-100.

Booch, G. 1994. *Object Oriented Analysis and Design*. New York: Benjamin/Cummings.

Buschmann, F., and R. Meunier. 1995. How Patterns Build Software Systems. *Electronic Design* 43: 90-100.

Chang, W., and L. Jones. 1994. Message Oriented Discrete Event Simulation. *Simulation* 63: 96-104.

Chervi, P., and C. Sautereau. 1994. ESCADRE: A Design Methodology. *Simulation* 62: 161-168.

Dowd, K. 1993. *High Performance Computing*. Sebastopol, CA: O'Reilly & Associates Inc.

Earnshaw, R., and D. Watson, ed. 1993. *Animation and Scientific Visualization*. New York: Academic Press.

Elmasri, R., and S. Navathe. 1994. *Fundamentals of Database Systems*. New York: Benjamin/Cummings.

Fitzgerald, J. 1993. *Business Data Communications*. New York: John Wiley.

Foster, I. 1995. *Designing and Building Parallel Programs*. New York: Addison-Wesley.

Famic Technology Inc. 1994. TRAXX Product Specification. Saint-Laurent, Canada.

Kuljis, J. 1994. User Interfaces and Discrete Event Simulation Models. *Simulation Practice and Theory* 1 (November): 207-221.

Lee, J., and P. Fishwick. 1994. Real-time Simulation Based Planning. *Simulation* 63: 299-315.

Mak Technologies. 1994. Introduction to Mak Technologies VR-Link. Cambridge, MA.

Marvel, O., and K. Watts. 1994. The Use of Simulation in Battle Management. Naval Postgraduate School, Monterey, CA.

Molitoris, J. 1994. Battle Manager Bibliography. Available on request from the author.

Molitoris, J., E. Harder, K. Kropp, and T. Taylor. 1994. A Modern Computer Simulation Facility and Some Applications: Naval Operational Modeling of Mine Countermeasures. Military Operations Research Society Symposium, Colorado Springs, CO.

Multigen Inc. 1994. Multigen Modeler's Guide. San Jose, CA.

Object Design. 1994. Object Store User's Guide. Burlington, MA.

Paradigm Simulation Inc. 1994. Vega Programmer's Guide. Dallas, TX.

Parsons, J. 1994. MWARS Briefing. Center for Naval Analyses, Alexandria, VA.

Quinn, N., and B. Feher. 1994. Simulation of Tactical Decision Making by Warfare Commanders. In *Proceedings of the Military, Government, and Aerospace Simulation Conference*, 23-28. Society for Computer Simulation, San Diego, CA.

Rheingold, H. 1991. *Virtual Reality*. New York: Simon and Schuster.

Taylor, D. 1990. *Object Oriented Technology*. New York: Addison-Wesley.

Viewpoint Datalabs. 1994. Real-time Dataset Catalog. Orem, UT.

Visix Software Inc. 1995. Galaxy Application Environment. Reston, VA.

Virtual Prototypes Inc. 1994. STAGE Product Specification. Montreal, Canada.

Weatherspoon, R. 1992. Modular Solutions for Counterdrug Simulation and Wargaming. In *Proceedings of the 1992 Applied Defense and Military and Government Simulation Conferences*, 92-98. Society for Computer Simulation, San Diego, CA.

AUTHOR BIOGRAPHIES

JOSEPH J. MOLITORIS is a research analyst at the Center for Naval Analyses with research interests in modeling, simulation, computers, communications, and information technology. His academic training was in theoretical nuclear physics.

THOMAS D. TAYLOR is Director of Technology Applications and Development at CNA. His background includes direction of the Naval Technology Office and the Submarine Technology Program at DARPA. He was trained in chemical engineering.