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

AN OPEN SIMULATION ARCHITECTURE FOR FORCE XXI

John A. Hamilton, Jr.

Dept. of Electrical Engineering & Computer Science
United States Military Academy
West Point, New York 10996, U.S.A.

Udo W. Pooch

Department of Computer Science
Texas A&M University
College Station, Texas 77843-3112, U.S.A.

ABSTRACT

Force XXI will be America' s Army in the 21st century. New technology, a post-Cold War world and a declining force structure have made simulation a critical means for defining and implementing Force XXI. Military organizations are hierarchical. In combat, these command levels operate simultaneously with varying levels of coupling. A platoon on platoon operation can be significantly affected by unilateral actions of higher level friendly or enemy headquarters. A high-fidelity simulation must be able to represent the simultaneous actions of several command echelons.

A multi-level simulation approach is presented in which different echelons of command can view objects at varying levels of detail consistently. A task force commander should be able to view the position and movement of a tank platoon on a map or a virtual view of the battlefield as seen from the platoon leader's hatch or real-time live video from an actual tank on the ground. Each view must be logically consistent with each other, so that the mountain on the map sheet affects movement in the same manner as the virtual mountain and the actual mountain seen via live video.

Various combinations of live, virtual and constructive simulations will run concurrently in the same simulation infrastructure. Expert systems will control constructive simulation nodes lacking human players. At any level in the simulation human players and expert system players may be interchanged. The opposing force will be similarly configured. Force on force virtual simulations will be supported in the simulation infrastructure. The resulting open simulation architecture (OSA) is discussed in detail.

1 INTRODUCTION

Event driven simulation is used to produce high resolution models. Probability distribution based simulations run considerably faster with some loss of resolution. An Army Corps of 100,000+ soldiers may have more than two thousand platoon-sized equivalent units. A simulation that represents the entire corps as platoon size elements will be extremely unwieldy. Performance constraints will limit the number of platoon-sized elements that can be represented by a high-fidelity event driven simulation.

The US Army uses live simulation, i.e. soldiers on exercises, vehicle/aircraft simulators or virtual simulation and computer simulations or constructive simulation. A seamless integration of live, virtual and constructive simulations at multiple echelons of command is required. A plug and play design is necessary to allow for both automated players and human in the loop players at various echelons of command for both the friendly forces and the opposing forces.

2 THE SIMULATION ENVIRONMENT

Operations research came into its own during the Second World War. The roots of the American military requirement for operations research can be traced back well before the war. In 1934, then Colonel George C. Marshall supervised the preparation of *Infantry in Battle*, which stated the following on page 1:

> The art of war has no traffic with rules, for the infinitely varied circumstances and conditions of combat never produce exactly the same situation twice. Mission, terrain, weather, dispositions, armament, morale, supply, and comparative strength whose mutations always combine to form a new tactical pattern. Thus, in battle, each situation is unique and must be solved on its own merits (Marshall 1939).

Clausewitz noted that the art of war is based upon "a play of possibilities, probabilities, good and bad luck, which spreads about with all the coarse and fine threads in its web, and makes War of all branches of human activity the most like a gambling game," (Clausewitz 1832). The web of possibilities referred to by Clausewitz is based upon the complex interactions of combat that when unraveled appear to be simple and deterministic. Unraveling those threads is non-trivial because there are so many interdependencies on the battlefield (Hamilton, White and Pooch 1995). Advances in technology have made it much more plausible to model the uncertainties outlined by Marshall. Decomposing the fine threads of combat make it possible to create high fidelity simulations. Such

simulations can serve as an unequaled peacetime as well as a wartime training tool.

Computers, with their capacity to handle large volumes of information, have been a critical aid to the military analysis community. Today, increased processing power can be further augmented by linking powerful heterogeneous computers into distributed computing systems. The Defense Department is working to integrate the varied training and simulation tools of the services through Distributed Interactive Simulation (DIS) (DIS Steering Committee 1993). In the DIS world, simulations are classified in one of three ways:

- •Live: simulations involving soldiers exercising on instrumented ranges.
- •Virtual: aircraft or vehicle simulators.
- •Constructive: automated war games.

The seamless integration of live, constructive and virtual simulation is a major objective in devising common DIS standards.

Distributed systems offer the potential to implement very high fidelity combat models. The same computing power and speed that promises to increase the problem domain bounds that can be represented also threaten to overwhelm a user with details. This problem is not an artificiality imposed by the raw power of computers. Rather it reflects the ability to devise ever higher fidelity models of combat--the most chaotic of all human activity. The US Army is the world's pacesetter for ground combat operations. Capt. J. R. FitzSimonds, USN, writes eloquently on this point:

> The most critical drag on high-tempo system performance is the cognitive limit of the human mind, the rate at which an individual can assimilate information and act. An information-intensive battle space may work to our advantage only if humans can be largely removed from the command loop. The need for speed will likely force today' s hierarchical command structures to become very flat, with automated analysis and decisionmaking largely replacing time-consuming and error-prone human deliberation. More profoundly, technical limitations of communications and data fusion may mean that humans will have to forego a traditional "picture" of the battle space. The question then becomes whether future US military commanders can accept a continuing reduction in their real-time battle information as the price of an increasing pace of activity (FitzSimonds, 1995).

The mobile strike force concept recently tested during the US Army's Prairie Warrior 95 exercise features a flattened hierarchy. While the Army of the 21st Century is likely to feature a flatter hierarchy, span of control limitations will prevent a flat hierarchy. More probable is a dynamic hierarchy that can collapse and expand as needed. Modern military communications have already made this possible. During the 1973 Arab-Israeli War, an Israeli lieutenant and his platoon was personally directed around Suez City by the Israeli theater commander (Dupuy 1978). This kind of zooming in on a critical area by senior commanders will be the norm rather than the exception in 21st Century warfare. This dynamic hierarchy must be represented in any high fidelity simulation.

3 MULTI-LEVEL SIMULATION

A multi-level simulation strategy is necessary to integrate different echelons and services into the same simulation. If a corps commander wants to focus on the progress of a platoon fighting as part of the covering force, he needs to see that part of the simulation in considerably more detail than other parts of the corps battle. However, a corps commander will not always be monitoring/fighting platoon battles. He/she must be able to quickly move back up to the corps level. Combat imperatives based upon the current tactical situation will determine the appropriate level of detail. It is clear that such changes must be accomplished accurately and very quickly.

Scaling multiple levels of command requires methods to deal with the potential combinatorial explosion that representing a division at the vehicle level would impose. A single vehicle object is simple to understand. As vehicles (nodes or objects) are added, the complexity of the interobject communication model grows almost exponentially (actually on the order of n^2 - n where n is the number of objects). An armor company with seventeen tanks has 272 different possible communications paths just within that company.

Fortunately a matrix of tank-to-tank communications would be sparse. Military doctrine dictates that most communications are hierarchical. Rather than simply heuristically limiting communications paths, a more flexible approach would be to apply a multi-level simulation strategy. While virtual (crew simulators) and live (field exercise) simulation participants would continue to be represented by event driven simulation; constructive (computer war game) elements could be represented by process-oriented (distribution based) simulations. Unless a constructive tank is of interest, it could be represented as part of a higher level equivalent object. Constructive units could be represented at the highest possible level of abstraction unless it was desired to "zoom in" on the activities of a particular element. Such higher level objects would still generate and receive the appropriate events and state changes but its intra-object level events would be transparent to the other participants.

To build a simulation infrastructure that is adaptable to service varying requirements in a large domain, multiple levels of abstraction of the base model are needed (Walczak and Fishwick 1988). An object-oriented model is the logical framework for a multilevel implementation. Tradeoffs always exist between

complexity and data sufficiency (Popken and Sinha 1994). Abstract model components have the advantage of great flexibility at the cost of specificity. Concrete implementations can provide more detail but less flexibility. From a software engineering standpoint, it seems clear that flexible abstract components must form the core of the simulation infrastructure. Generic tank objects could then be instantiated as specific, concrete models. Thus the same framework can be reused over and over for actual or planned models. An Israeli Merkava tank may not be available for live testing. An abstract tank object could be instantiated to represent the Merkava. This would allow for interoperability rehearsals between US/Israeli forces on short notice.

For a variety of contingencies, it is not practical to rehearse long-term international interoperability. Being able to instantiate generic equipment objects would make it possible to represent a wide range of unit-types on short notice. A distributed simulation environment would allow for international joint exercises without forces having to leave home station.

An object-oriented model is the necessary framework for a multi-level simulation architecture. Object-oriented technologies have been used previously in simulation research, e.g., in graphical representations and user interfaces, "intelligent objects," parallel/distributed simulation, and simulation software engineering (Bischak and Roberts 1991), but do not yet have widespread application in simulation. Their use in multi-level simulation has been rare.

The fundamental level of simulation is that of an object. The object will have a state that evolves with time or the occurrence of certain events. The simulation must determine and record the evolution of the state. The objects in the system are interrelated, i.e., the outputs of one object will influence the state of other objects. The state of an object can take many forms, depending upon the nature of the real entity it represents. The state may include variables having discrete values (e.g., alive or dead), variables evolving continuously in time (e.g., the location of the entity represented by the object), or statistical parameters (e.g., distribution of the strengths of the platoons in a battalion). Since objects may be composed hierarchically, their state may be some form of aggregation of the state of component objects (e.g., the total ammunition left in a platoon).

An object decomposition of the problem space is an important early step in the creation of a multi-level simulation architecture. The ability to define, incorporate, change, and map objects that represent elements and components at different levels of abstractions is essential. We include the concept of multi-level abstraction where two or more levels of detail may be of interest at any point in time. We distinguish between abstraction and the formation of object classes, both of which are important aspects of developing the open simulation architecture (OSA). Abstraction is the selective examination of certain aspects of a problem. Its goal is to isolate those aspects that are unimportant. The formation of object classes is the identification of classes whose members share some set of properties.

The component objects form the basis for aggregation, inheritance, naming, abstractions, attributes, time management and resolution, and encapsulating state and behavior. The desired level of detail that is appropriate and useful for an OSA requires the capability to move dynamically through the different levels of abstractions, time granularity, and instantiated domain space.

4 MULTI-RESOLUTION SIMULATION

One of the major challenges of building a large multi-level simulation is the requirement for scalability. Computation and communication techniques that work well for small systems often become totally unusable for large systems. Simulations of large numbers of low level individual units (e.g., tanks or foot soldiers) can swamp computational capabilities, while transmission of large numbers of high resolution images can swamp communication channels. We are concerned with identifying relationships between models described at different levels of detail. Traditionally, researchers have been interested in processes deriving more abstract relationships from more detailed ones. However, the multi-resolution simulations we seek, and the ability to dynamically view them at different levels of resolution, requires that we be able to move in the reverse direction as well. That is, we must be able to extend a single abstract relationship into a number of more detailed ones.

Closely related to levels of abstractions are the concepts of aggregation and decomposition. Generally, to move to a higher level of abstraction (less detail) involves aggregation or the representation of several more detailed components by a single equivalent component. Grouping components and aggregating variables is quite a well known procedure. Conversely, to move to a lower level of abstraction (more detail) the model has to be further decomposed to adequately describe the modeled physical entities. This procedure is more difficult and has largely been an unsolved problem. It is a major requirement for our OSA, since without it, the computational cost of achieving high resolutions will be prohibitive.

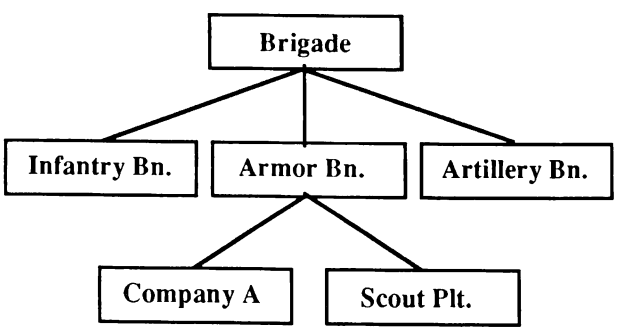

Figure 1: Simple decomposition of a combat brigade

In Figure 1, company A and the scout platoon of the brigade's armor battalion may be the units of interest. The simulation might then only use process-oriented simulation to model the battalions and the brigades, while the units of interest are running a more detailed event-driven simulation.

In process-oriented simulation, the focus is on abstraction with respect to individual model components or processes. Process abstraction can be described as the transformation of one process to another occurring at a different level of abstraction. Moving to different levels of abstraction during the simulation provides the analyst different views of the "running" model. Such access can provide the key to the model's behavior and provides the opportunity to change lower level processes so the impact can be assessed at some higher level. It is closely tied to the multi-resolution issue described above.

The fundamental difficulty that arises from moving down in abstraction to greater levels of detail is the problem of ensuring state, time, and event consistency among all active elements. For example, if one is interested at one point in time in a tank platoon level of information, and, at another time, individual tank information, how are these two kept consistent? One way, of course, is to always carry out the full simulation at the individual tank level, and aggregate information at the platoon level when needed. This solution, however, will require prohibitive amounts of computation when utilized throughout a full simulation. Our approach, which avoids unnecessary computation, is to evolve summary statistical information at a high (e.g., platoon) level, and deaggregate this information into initial conditions for lower levels (e.g., individual tanks) when one wishes to examine the situation in detail. The feasibility of this approach has been demonstrated in the TAMU Traffic Simulation Model (Wall 1993, Vidlak 1993).

The multi-resolution simulation described earlier provides the capability to simulate only at the level needed, suppressing unneeded low level simulations. Similar support is needed to reduce the impact of computational and network bandwidth limitations for various forms of information that must be distributed, e.g., images, maps, graphics. We approach this in two ways.

First, we extend the multi-resolution concept to representation of information, as well as the simulation resolution, and develop techniques to provide only the resolution of information needed for the task at hand. Data at the proper resolution must be delivered to destinations in real-time without causing perceptive distortion to maintain meaningful connectivity between communicating parties. The resolution of information needed by varying levels of command and control provide a natural assist to this process since higher command levels often need higher levels of abstraction in the information they use. Consequently, as the level of information abstraction increases there is typically a decrease in the information resolution.

To accomplish multi-resolution data management, data are divided into multiple resolution versions, which will be achieved by incrementally adding detail to lower resolution data. When an inquiry needs to retrieve data of some resolution, the decomposed information can be transmitted incrementally and reassembled to the required level of resolution. The criteria in determining information resolution increments include the total information demand, available system resources, presence of urgent events, and the computing ability of the receivers. For certain kinds of information, images, advanced compression techniques have yielded compression ratios of 100 to 1 with little loss of resolution, and approaching 200 to 1 with only modest loss of resolution (Lee et al. 1994, Chan et al. 1995). The combination of these techniques is expected to result in acceptable utilization of computational and network resources.

5 OPEN SIMULATION ARCHITECTURE

We combine multi-level simulation and multi-level resolution to form the basis of what we call an Open Simulation Architecture (OSA) (Refer to Figure 2). The OSA will incorporate multi-level abstractions and the ability to navigate through various levels of mixed (DIS-defined: live, virtual, constructive) simulation components in a consistent manner. Multi-level/multi-resolution abstractions provide interfaces to command echelons that provide varying views and details, from the individual component object, such as an instrumented vehicle or soldier, through aggregation to the brigade and/or corps level and beyond. In addition to operating at these different levels of abstractions, it is important that the analysts be able to view/interact with the OSA from the context of each level and time granularity.

Most traditional simulation systems are limited because they are unable to handle models with different degrees of aggregation. Yet, because the appropriate level of aggregation may not be known when the simulation system is built, it is necessary to be able to move dynamically through different levels of aggregation. Current systems are fairly inflexible with respect to changing levels of aggregation. The difficulty arises from the problem of ensuring consistency among all active levels. The ability to define objects that represent aggregates of other objects is essential for the OSA. In addition to the model operating at multiple levels of aggregation, the user must be able to view and interact with the system from the context of each level of aggregation. We explore briefly in the paragraphs below how we can overcome the limitations of traditional methods.

5.1 Interface Specifications

Not only must the objects be derived or integrated in a state consistent manner, the development of clear and precise specification interfaces is at the heart of the object oriented approach. Very precise and well defined interface specifications are needed to co-mingle live, virtual, and synthetic environments--which may exist at differing levels of detail at different nodes in a distributed simulation infrastructure. These then need to communicate via a well defined message protocol. The interface structures must also provide the capability of navigating among various process-oriented simulation layers and event-driven simulation layers.

The ability to mix and match different forms of objects representing the same real entity (i.e., constructive, live or virtual) is dependent upon the definition of appropriate *classes* of objects having the same interface. While the definition of these classes is ultimately in the application domain, the concepts and a basic set of class definitions are essential elements of the Multi-level, multi-resolution distributed simulation (MMDS) hierarchy.

Finally, all interfaces must be backward compatible with DIS. It must be recognized that DIS standards are likely to be modified; considerable care must be exercised in designing interfaces for known present requirements as well as accommodating future DIS standard revisions.

5.2 Timing and Synchronization

Time management is essential to the envisioned OSA. Without near real-time performance, seamless integration of the virtual and constructive simulation with the live simulation will not be practical. You cannot halt a moving tank platoon while you wait for the rest of the simulation to catch up. Thus, real-time communication is central to the success of distributed simulation. Real-time communication deals with transmitting delay-sensitive messages in a distributed system. Minimizing the delays of messages communicated among the nodes will reduce the impact of roll backs and hence improve the scalability of the system. Further, in a hybrid simulation system (i.e., one that consists not only of simulated devices but also actual ones) message delays must be carefully controlled in order to achieve desired (e.g., synchronized) effects. Thus, a successful distributed simulation system must have the support of real-time communication. Our objective in this part of the project is to develop real-time communication technology that can ensure the satisfaction of timing requirements of message transmission in the OSA.

We should first establish a framework which allows us to analyze the delays in the message transmission. Based on the framework, we should address resulting issues in order to provide feasible solutions for distributed simulation applications. Thus, the actual performance of the proposed methods must be evaluated, based on the following measurements.

- **Schedulability**: This is a direct measure on the capability of meeting message delay requirements. There are two possible ways to measure it:
  - The worst case achievable utilization. This is a threshold utilization below which the delay requirements are always met.
  - The probability of meeting delay requirements for a given load. The higher the measure, the better the performance is in each case.
- **Complexity of schedulability testing**: Sometimes, it is necessary to know if a particular delay requirement can be met. This is done by schedulability testing. The less the complexity, the easier to be used.
- **Buffer requirement**: A message will be lost and never be delivered if a buffer overflows. Our previous analysis on some networks showed that different scheduling methods may result in different buffer requirements. An upper bound of buffer size must be derived for each scheduling method proposed.
- **Stability**: This reflects the system's sensitivity to change in configuration. One would prefer that a small change in the system configuration (e.g., a

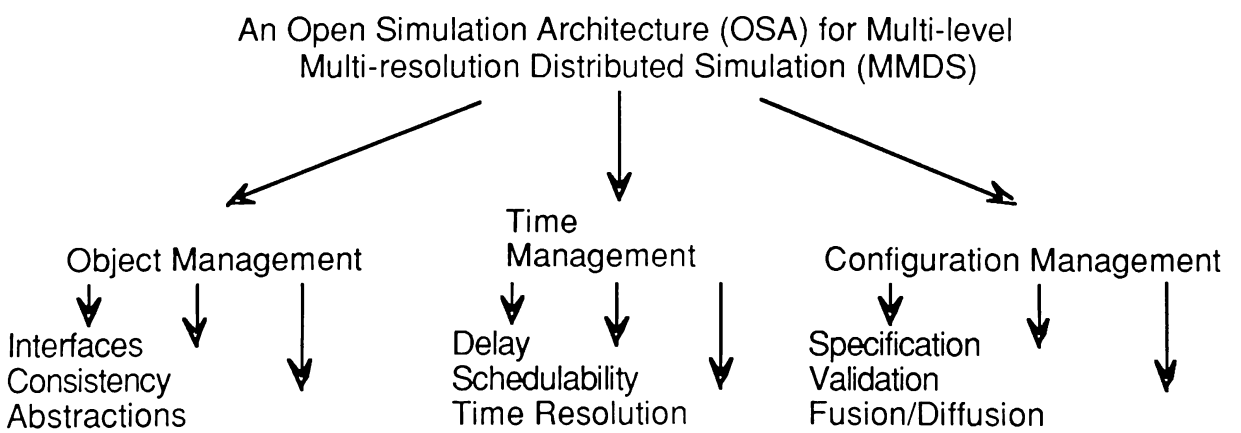

Figure 2: Functionality of the OSA

slight increase on a node) has minimum impact on the system's capability of meeting message delay requirements.

We propose to obtain these measurements by rigorous mathematical analysis. One might argue that values of some measures could be obtained experimentally. However, there are two problems with the experimental approach here: 1) Usually these kinds of experiments are tedious and costly in terms of both time and resources; and 2) the measured values obtained are only correct for the limited cases and may not apply to the general case of simulated system. We thus believe an analytical approach is necessary.

5.3 Composition Rules - The Building Codes of the Architecture

In general, the OSA itself will not be a fixed entity, but rather evolve with changes in technology and requirements. The process by which the evolution occurs is at least as important as the architecture itself. The need to identify objects and object classes to facilitate information-sharing at the applications level was stated earlier, and is central to the development and evolution of the OSA. It is necessary to define what information is to be shared, among what set of applications, and by what set of users. The tool designer must also provide application-level interoperability across the domains of interest as a necessary pre-cursor, i.e., the applications across which interoperability is desired must be designed to use common data definitions and representations.

5.4 Validation

The validity of any simulation depends on the accuracy of the model(s) representing the real system. The model must be sufficiently detailed to provide the analyst with information about the aspects of the system's performance that are of primary interest. The best method of validating a simulation is to judge its performance. If the inferences drawn from the analysis of the simulation's output allow correct conclusions to be drawn about the system or the situation being simulated, then the simulation can be assumed to be valid for that particular situation. However, using such tests for a validation procedure has some drawbacks. Courses of action not taken or decisions not made by the simulation could provide some added insight into the real validity of the simulation and the model it is built on. Thus, it may not be sufficient to assess the validity of the simulation only from observations of the simulation's performance.

A number of methods for assessing the validity of a simulation use techniques other than direct observations of its performance. These methods, just like the observation technique, do not guarantee that the simulation is valid, but they do provide a basis for assuming that the simulation is valid. The first of these methods deals with validating the design of the model. This form of validation is simply a checking problem in which the design of the model is verified at different stages of the development. The process of modeling a system is broken into two phases, the conceptual and the implementation phase.

In the conceptual phase the logical flow of the system being modeled is determined, and the relationships between the various subsystems are formulated. The factors likely to influence the performance of the model are isolated and tentatively selected for inclusion. The model is then traced in reverse order. This technique is somewhat analogous to verifying the accuracy of an arithmetic result by applying the inverse operation to the result. The implementation phase of the model includes selecting and quantifying procedures for the model, coding the model, and actually using the model. Here it can be determined how accurately the model represents the real system. The easiest way to assess the validity of a model is to compare simulated results with the behavior of a system whose performance characteristics are known. Other bases for comparison of the simulated results include theoretical predictions, and historical data.

The main purpose of MMDS, as with any simulation, is to represent a specific real world system. Whatever level of abstraction chosen should produce output that is statistically equivalent to another level. Based upon the implementation of the different abstract levels, it would seem to be an intractable exercise to mathematically (or rigorously) prove each mode's equivalence to each other. Rather emphasis should be placed on using statistical tests to establish equivalencies of the output data. Thus, validation of the ability to handle different abstraction levels will be centered on establishing an equivalence of simulations featuring various resolution modes.

A further validation test, besides those of historical data, and statistical equivalence, will be to compare the MMDS output with those of more traditional existing simulations (each of which has been separately validated), and in the case of a virtual-live-synthetic simulations, to each other.

Two other validation procedures should be utilized, internal validity testing and variable-parameter validity testing. The internal validity tests consist of performing several simulations using the same model and input parameters, and then comparing the output to detect variability. If the variability of results is high, the model will probably be of little value as a predictor, since it will be difficult to assess whether changes in the output are due to changes in input parameter settings, the model's inherent variability, or a combination of the two. The limiting effect of internal variability on the usefulness of the model can also be viewed in light of the real system's possible behavior under similar circumstances. It is unlikely that a real system of interest, when presented with identical operating conditions, will produce radically different results.

The variable-parameter validity tests consist of varying parameters and variables to determine their effect on the simulation and the subsequent output. If the impact of certain variables or parameters is large (statistically tested and determined) compared with the initial estimate of their impact, the validity of the model is questioned.

Other additional techniques include face validity, hypothetical validity, and event or time series validity. Face validity is measured by evaluation of output results by individuals familiar with the real system. Hypothetical validity, rarely properly investigated, is a test of negation. Event or time series validity is used to determine if the simulation predicts observable events, even patterns, or the variations in the output variables.

6 CONCLUSIONS

Seamlessly integrating live, virtual and constructive simulation requires an object decomposition of the problem space with well defined interfaces. At a minimum, all interfaces must be backward compatible with DIS. It must be recognized that the DIS standards are likely to be modified so considerable care must be exercised in designing interfaces for known present requirements as well as accommodating future DIS standard revisions. Strict adherence to interface standards is critical to allow for custom tailoring of the information infrastructure.

As the Army transitions to Force XXI, there will be several generations of technology represented in the operational units. Increased emphasis on joint and coalition warfare will add additional systems and levels of sophistication to an already heterogeneous environment. In order to operate in this fast moving environment gateways must be constructed that can integrate new systems into the simulation infrastructure.

An emerging technology supported by the OSA is the integration of stove-piped databases and data repositories into integrated information systems (Miller 1994). The implementation of simulation interface gateway objects (SIGOs) will facilitate access to already existing databases and provide integration capability across numerous platforms. It is feasible to consider the implementation of a virtual meta-database in which distributed data are seamlessly integrated via the direct translation capability of the SIGOs.

The simulation infrastructure will support the integration of the acquisition process into the simulation process. By modeling current and evolving threat arrays, it can be determined if current capabilities are sufficient and what deficiencies require priority of effort. General strategy questions such as force mix could be addressed and then specific design issues such as precise system capabilities i.e. 140mm main gun versus 152mm missile launcher as main armament for example. Finally, the simulation infrastructure could support fine tuning efforts such as what is the optimal number of this new system in a unit? Questions such as these are both system specific and project lifecycle stage dependent (McHaney 1991).

Expert system technologies play a critical role in supporting the "plug and play" strategy of interchangeable human and machine participants. After developing an appropriate expert system engine, variants to model, US, Allied and Threat doctrine would be developed. Once the doctrinaire expert system player is fielded, deviations from expected doctrine and performance can be developed. It is not unreasonable that when good intelligence exists, specific threat commanders may be modeled.

The rapid evolution from a bipolar to a multipolar world environment as well as the ever increasing pace of technology make the US Army's transition to Force XXI a complex effort to hit a rapidly moving target. The transition to Force XXI will be incremental. Doctrine, weapons systems and force structure are progressing at different rates. An open simulation architecture is required to support the evolving structure. Multi-level, multi-resolution distributed simulation is a means for American commanders to continue to control the tempo of combat operations and to avoid being controlled by information-driven decision cycles.

REFERENCES

Bischak, D. and Roberts, S. 1991. Object-oriented simulation, *Proceedings of the 1991 Winter Simulation Conference*, 191-203. Institute of Electrical and Electronics Engineers, Phoenix, AZ.

Clausewitz, C. von, 1988. *On War*, London: Penguin Books.

Chan, A., Chui, C., Lemoigne, J., Lee, H., Liu, J. C. and El-Ghazawi, T. 1995. On the performance impact of data placement for wavelet decomposition of two dimensional image data on SIMD machines. To appear in *Proceedings of the Frontier 95 Conference on Massively Parallel Processing.*

DIS Steering Committee. 1993. "The DIS Vision, A Map to the Future of Distributed Simulation," Comment Draft, University of Central Florida/Institute for Simulation and Training, Orlando, Florida.

Dupuy, T. N. 1978. *Elusive Victory The Arab-Israeli Wars 1947-1974*, New York: Harper & Row.

FitzSimonds, J. R. 1995. The coming military revolution: opportunities and risks. *Parameters xxv, 2*:30-36.

Hamilton, J. A., Jr., White, G.B. and Pooch, U.W. 1995. Towards a concurrent theory of combat: a parallel processing approach, to appear in *Military Review lxxv, 6.*

Lee, H. J., Liu, J. C., Chan, A. K., and Chui, C. K. 1994. Parallel implementation of wavelet decomposition/reconstruction algorithms. In *SPIE*

Wavelet Application Conference, 248-259.

Marshall, G. C. 1939. *Infantry in Battle*, Washington: The Infantry Journal Inc.

McHaney, R. 1991. *Computer Simulation, A Practical Perspective*, San Diego: Academic Press.

Miller, J. A., Potter, W. D., Kochut, K. J. and Ramesh, D. 1994. Object-oriented simulation languages and environments. To appear in an IEEE Press Monograph, *Object-Oriented Simulation*.

Popken, D. A. and Sinha, A. P. 1994. Object-oriented frameworks for multi-level simulation modeling. To appear in an IEEE Press Monograph, *Object-Oriented Simulation*.

Wall, J. A. 1993. Multilevel abstraction of discrete models in an interactive object-oriented simulation environment. Ph.D. Dissertation, Department of Computer Science, Texas A&M University, College Station, Texas.

Walczak, S. and Fishwick, P. 1988. A centralized methodology for multilevel abstraction in simulation. *Simuletter 19*, 2:25-31.

Vidlak, M. D., 1993. User-object interfaces in an object-oriented discrete event simulation, Ph.D. Dissertation, Department of Computer Science, Texas A&M University, College Station, Texas.

AUTHOR BIOGRAPHIES

JOHN A. (DREW) HAMILTON, JR., Major, US Army, is currently pursuing a Ph.D. in Computer Science at Texas A&M University enroute to the Department of Electrical Engineering and Computer Science at the U. S. Military Academy, West Point, New York. He was most recently Chief, Officer Training Division at the Army Computer Science School, Fort Gordon. Major Hamilton has a B.A. in Journalism from Texas Tech University, where he was commissioned in Field Artillery, an M.S. in Systems Management from the University of Southern California and an M.S. in Computer Science from Vanderbilt University.

UDO W. POOCH, Ph.D., P.E., received his Ph.D. in Theoretical Physics from the University of Notre Dame and is the E-Systems Professor of Computer Science at Texas A&M University. Dr. Pooch is the author of numerous articles and books, and most recently authored with Lt.Col. James A. Wall, *Discrete-Event Simulation* published by CRC Press. Dr. Pooch is a very active researcher supervising projects in network simulation, network security and fault-tolerant distributed environments.