Proceedings of the 1998 Winter Simulation Conference
D.J. Medeiros, E.F. Watson, J.S. Carson and M.S. Manivannan, eds.

ESSENTIAL TECHNIQUES FOR MILITARY MODELING & SIMULATION

Roger D. Smith

STAC Inc.
Orlando, FL 32765, U.S.A.

ABSTRACT

This tutorial will identify and explore the essential techniques necessary for modern military training simulations. It will provide a brief historical introduction followed by discussions of system architecture; simulation interoperability; event and time management; distributed simulation; and verification, validation, and accreditation. This will be followed by fundamental principles in modeling and specific military modeling domains.

The growth in government sponsored simulation programs has drawn engineers and scientists from other fields. These practitioners bring valuable skills, but lack an appreciation for the historical and technical foundations of simulation. The tutorial will familiarize the audience with important areas and give them an appreciation for the complexity of developing large simulations. We suggest that a need exists for academic and commercial courses that focus on this topic. This tutorial may serve as a template for one such course.

1 MILITARY DOMAIN

The military has a long and rich history of using models and simulation. The US military alone spends hundreds of millions of dollars acquiring, designing, fielding, and operating simulation systems. These systems have been categorized by the Department of Defense into training, analysis, and acquisition applications. A wide variety of training is conducted through the use of virtual, constructive, and live simulations (Davis 1995). Virtual training simulations are those in which the trainee is immersed in a virtual world where physical actions such as driving a vehicle or firing a weapon have a direct visible on the synthetic world they are in. Constructive simulations are widely known as wargames. Tactical and strategic decisions are reflected in the movement of military icons on a map, testing the commander and staff's ability to use their forces effectively. Live simulations are the application of real equipment in mock combat scenarios or firing ranges. These allow pilots, tank drivers, and other soldiers to practice the physical activities of war with their real equipment.

Analytical simulations are used to study problems like force composition, weapons effectiveness, and logistics issues. This community is strongly influenced by the science of operations research and may produce simulations very similar to those used for constructive level training. Analytical simulations usually differ in that they do not focus on interactive exchanges with people during a simulation run. This allows them to execute much faster or slower than real time without adversely impacting a human operator (Law 1991).

The military acquisition community uses models to identify shortfalls in its ability to perform specific missions or meet certain threats. These models identify weaknesses in our military forces without the necessity of testing them in war. This community also uses detailed engineering level models to conduct studies of the design of weapons under acquisition.

1.1 History

Military simulations have arrived at their current state of sophistication and application through a long history of experimentation and evolution. We can identify the existence of models of warfare as far back as 5000 years ago as discussed in Perla 1990. Historical records indicate that the Chinese developed a wargame called Wei-Hai around 3000BC. No diagrams or artifacts of this game have survived, but descriptions lead us to believe that it was similar to the modern game of Go. Players used colored stones on a grid system to control as much space on the board as possible. The modern game of Go emerged around 2200BC.

Chaturanga emerged in India around 500AD accommodating two or four players on a checkered board. Each was equipped with four pawns, a king, elephant, horse, and chariot. The objective of this game was to capture the enemy's pieces rather than to control area. The modern game of Chess evolved from Chaturanga around 1400AD in Southern Europe.

Examples of the use of sand tables and miniature replicas can be found among the Roman legions around 30AD. This form of training can be seen right up to the present with the use of these items to train soldiers in the military academies and schools. Only the advent of computer simulations has begun to replace these apparatus.

The modern era of wargames began in 1664 with the development of Koenigspiel (the "King's Game") by the German Christopher Weikhmann. This game consisted of a checkered board with 30 pieces representing military ranks that included the King, Marshall, Colonel, and others down to Private. Additional developments followed through the 17th and 18th centuries, these included War Chess, and Kriegsspiels. Each added detail and more intricate techniques for operations of the game. Kriegsspiels, developed by Baron von Reisswitz in 1811, used contoured terrain, porcelain soldiers, and the new concept of a starting scenario with a stated military objective.

During the twentieth century we have experienced the evolution of wargaming into a scientific application of techniques from operations research, analytical game theory, Monte Carlo techniques, the Lagrange-Multiplier Method, mathematical programming, and systems analysis. These games and computer systems incorporate more reasoned mathematic techniques than were feasible under manual operation (Davis 1995).

1.2 Interoperability

Simulations have traditionally been independent, stand-alone systems that address specific problems and adhere to a unique architecture established by the designer. This approach has persisted from the earliest games through the most modern computer simulations. Around 1988 the military began to explore the possibility of linking multiple interactive training simulations to allow them to interoperate with one another during execution.

In 1988 the Defense Advanced Research Projects Agency (DARPA) initiated a program called Simulator Networking (SIMNET) to create multiple tank simulators that could be joined over a network such that each could detect, engage, and destroy the others (Miller 1995). This program resulted in the establishment of important principles for simulation interaction and the creation of a network messaging protocol to exchange essential data. SIMNET was the forerunner of the Distributed Interactive Simulation (DIS) protocols. DIS attempted to generalize the SIMNET technology so that it could be applied to a wider variety of combat vehicle simulators such as trucks, helicopters, fighters, ships, and soldiers.

At the same time, members of the constructive training community were developing methods for linking simulations for higher level combat events. The Distributed Wargaming System fielded at the German ACE-89 exercise demonstrated the feasibility of tracking military units in other simulations and engaging them effectively and accurately. This experiment lead to the development of the Aggregate Level Simulation Protocol (ALSP) to demonstrate interoperable training at the staff level. ALSP linked seven existing simulations from each military service by providing both the network messages and software services for insuring consistency and causality between the simulations (Wilson 1994).

The Defense Modeling and Simulation Office (DMSO) is developing the High Level Architecture (HLA) to replace both DIS and ALSP (DMSO 1997). Those methods have proven to be very system specific and do not provide a general interoperability solution that can support future simulation systems and missions. The HLA defines: 1) rules for simulation interaction and for the behavior of a family of simulations; 2) Object Model Templates for expressing the military systems and activities that are represented in any simulation system; and 3) an interface specification to support interoperability between multiple simulations. The Runtime Infrastructure (RTI) is a software package that manages the interactions between distributed simulations according to that interface specification.

1.3 Verification, Validation, & Accreditation.

Simulations are creations from the minds of human designers. Though every effort is made to insure accuracy, compromises are always made and mistakes are inevitable. It is essential that all simulations be tested to establish their accuracy and appropriateness for specific problems. This process is known as verification, validation, and accreditation. These are applied to a simulation development cycle that assumes that the real world system to be replicated is identified and a conceptual model of it is defined. This conceptual model is then encoded as computer software. These three items form the points of a triangle where VV&A is used to insure that the transformation from one point to the next is accurately done (Figure 1) (Sargent 1987).

Validation is the process of determining the extent to which a conceptual model is an accurate representation of that portion of the real world that is important to the model sponsors. Essential aspects of the real world must be captured in the conceptual model that represents the problem to be addressed. In paraphrase, validation is often described as answering the question, "Are we building the right product?"

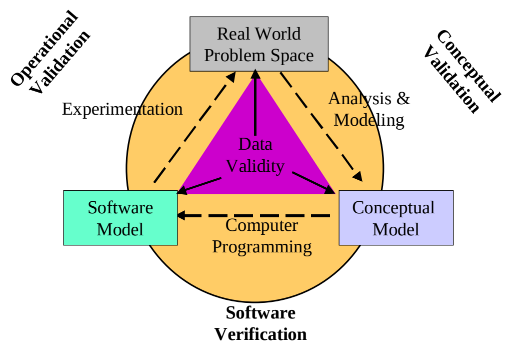

Figure 1: Verification and Validation of Simulation

Verification is the process of determining that the software product is an accurate implementation of the conceptual model as it was designed. This process insures that the software performs the operations as they were described in the conceptual model. Practitioners also attempt to identify the degree of control that the developing agency has over changes to the software. The intent is to verify that the current software is correct, but also to provide a level of assurance that it will remain correct in the future. Verification is often described as answering the question, “Are we building the product right?”

Finally, accreditation is an official determination that the simulation is acceptable for some specified purpose. No simulation is a universal solution to all problems in a domain. Each addresses a specific class of problems and may only be valid under the conditions found in those problems. Accreditation defines the set of problems for which a simulation is a good and useful model.

2 INFRASTRUCTURE

Within each simulation there is an infrastructure that supports the operation of the system, but is itself largely domain independent. An infrastructure can support many different simulations and is a potential source of software reuse.

2.1 System Architectures

When every simulation was custom crafted for a specific application there was no need, nor opportunity, for emphasis on an underlying architecture to support the extension of the system to future problems. Neither was any thought given to the reuse of the architecture by other simulation developers. As simulation science and simulation products matured, it became common to design a simulation such that certain operations could be encapsulated as libraries and used by many different customers. These libraries contained routines for generating random numbers, formatting specific reports, performing complex mathematical and statistical operations, and managing simulation execution. The evolution of commercial vendors to sell these libraries encouraged developers to design their simulations to take advantage of these products. This was the beginning of a widespread reusable architecture for simulations (Law 1991).

Within some military simulation projects a common structure began to emerge and repeat itself (Figure 2). (Smith 1995) This “architecture” was focused on the functional nature of the missions to which the simulation was put. This centered on a simulation engine that performs both execution management and modeling functions. Simulation input data is created by a Scenario Generator. Simulation output data is analyzed by an After Action Review system. A Controller Interface is used to manage the starting, execution, and stopping of the simulation. A Training Interface supports interactive participation by users. Finally, a Network Interface allows communication between simulations operating on different computers. This allows interoperability between heterogeneous simulations and the distributed execution of a single simulation system.

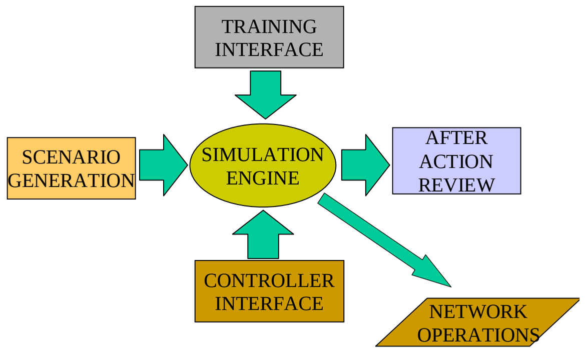

Figure 2: Functional Components of Military Simulations

Recently, object oriented architectures that provide greater interoperability and efficiency of execution have begun to emerge. These architectures promise an infrastructure for simulations that may be reused by multiple projects. If successful, this approach will allow developers to create a complete working simulation system simply by adding detailed models to the provided infrastructure. This can potentially eliminate as much as 90% of the time and cost of creating a simulation system.

The most ambitious and widely watched architecture of this new form is the Joint Simulation System (JSIMS). This project is attempting to unify the next generation of staff training simulations for the Army, Air Force, Navy, Marine Corps, and Tactical and National Intelligence Communities. JSIMS will provide a layered architecture with object oriented software frameworks supporting model specification (Figure 3) (Powell 1996). The architecture also provides a platform independent software product by creating a System Abstraction Layer between the simulation software and the operating system of the computer. The Object Services layer allows the infrastructure to efficiently distribute simulation objects, manage the progression of time, and store historical data through mechanisms invisible to the developers of the models. The Support Services provide object-oriented frameworks that are foundation classes for each type of object that can be represented. These also define the interactions that can take place between simulated objects. The framework object classes are extended to create specific models for each unique piece of equipment. This extension specializes both the characteristics of the object and the interactions it can have with other objects. This layer includes translation mechanisms that allow a simulation to exchange data with a wide variety of external systems - primarily simulations and military command and control computers. Specific models and tools form the Application layer atop the architecture.

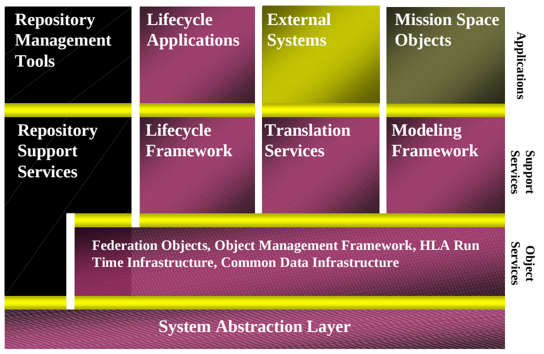

Figure 3: JSIMS Architecture

The JSIMS architecture is the most advanced available in the military simulation community. It is currently under development and details on the value it can provide are not yet available.

2.2 Event Management

Simulations are dynamic representations of systems. The execution of events that allow the simulation to portray the state of the real system at many points in time differentiates a simulation from a static model. These events are scheduled upon instantiation of the scenario and throughout the execution of the simulation. Since events are such an integral part of the simulation, it is important that they be managed accurately and efficiently. Events are usually organized into some form of list and stored through a variety of computer structures. Storage may be in the form of an ordered array, linked list, tree, or other structure. Whatever the form, each event contains information about the operation to be performed, the trigger for its execution, and the identity of the objects that it will operate upon.

A simulation may be structured such that events are handled by executive software which has the ability to contact each object and apply the event to it. The structure may also allow the event to belong directly to an object. Executive management simplifies the application of events that require multiple object interactions. Direct object management of events allows greater self-containment, independence, and reusability of each object. However, it also requires more inter-object communication to correctly apply complex events.

2.3 Time Management.

Since simulations represent the real world, they usually contain some representation of time. In most cases, time is the variable that orders and separates the execution of all events. When a functional simulation operates on a single computer system, the management of time is relatively simple. The simulation may choose to move forward in defined discrete time steps, or according to the times of events being executed. A “time stepped” simulation usually contains a mechanism for both time progression and event management that allows the simulation to set the time and execute any event prior to that time. New events are caused at determined points in the future. An “event stepped” simulation does not contain mechanisms for generating regular time steps. Instead, it chooses to represent only those instances in time at which events actually occur. The time of the simulation jumps from one event time to the next. All activity between these times is represented as a duration over which the object state changes.

With the advent of networked and parallel computer equipment, simulations were developed to take advantage of this hardware. As a result, it became necessary to synchronize time across multiple software applications. This synchronization insures that events happen in an order that preserves their causal relationships. Because of delays in network message delivery it is possible for events to arrive at an object in the wrong order. Since the object can not determine whether event messages are enroute, it does not know whether the most current event in queue is the next in the execution sequence. To address this problem, techniques for both conservative and optimistic synchronization of time across processors were developed.

Conservative synchronization provides a mechanism in which all objects are held in strict lock-step progression into the future (Fujimoto 1990). This is accomplished through queues that hold the latest messages from each simulation on the network. Each simulation must consult these queues to determine the time within other simulations. Each simulation is allowed to process all events up to a time at which all events prior to it have been calculated and distributed. The synchronization software determines this time and has the assurance of each simulation that no events will be applied in the period prior to this. Though a simulation may have no future events to inform the network about, it is required to send a "null" message that identifies a time prior to which it will generate no events. This promise prevents deadlock of simulations that are waiting for others to provide an event message.

Optimistic synchronization seeks to maximize the use of computer assets available to the simulation, and perhaps to finish execution faster than conservative synchronization will allow (Fujimoto 1990). Under this time management technique each simulation is allowed to process all events available with no consideration for the time at which other simulations are operating. However, the system is required to process all events in the correct order, including those that arrive late or out of order. A simulation may race into the future but subsequently receive an event message that happened in the past. When this occurs, the simulation is required to "rollback" all events until it can insert the new event in its proper place and re-execute all of them in order. This synchronization mechanism creates a distributed system in which each computer is racing independently into the future and is periodically interrupted by the necessity to go back and redo some of its work. The premise of this approach is that, in spite of rollbacks, the entire distributed simulation will complete its mission faster than it would have under conservative synchronization. Since training simulations can not require humans to follow this same repetitive experience of time, they can progress with Global Virtual Time (GVT). All events prior to this time are guaranteed to be in the past of all simulations on the network and none of these events are subject to rollback. This provides a foundation for the interactive user to experience a harmonic simulation time in spite of the rollbacks happening in front of GVT.

2.4 Issues in Parallel and Distributed Simulation

Both parallel and distributed simulations free the system from the limits of a single computer system and the necessities for co-location of all participants and sub-systems. This also creates a unique variety of problems that must be addressed to ensure causality, efficiency, and accuracy. Some of these, such as the time management problem, have already been addressed in this tutorial. But there are many others, only a few of which will be described here.

In the interoperability section we discussed the search for a common protocol that can support multiple models of the real world. Since each simulation represents the real world in a slightly different manner it is very difficult to create a standard protocol that is useful for joining all of them into a common execution. One of the strengths of the High Level Architecture is that it recognizes this fact and attempts to provide services that are useful for many different protocols. Within military simulation circles the concept of a "reference federation" is evolving. This attempts to categorize simulations into groups for which a common protocol is feasible, but also identifies boundaries across which no single protocol is likely to suffice.

All simulations are subject to the efficiencies of the computers and networks upon which they reside. Networks currently provide sufficient bandwidth for simulation messages in small scenarios. They do not appear to be able to deliver all of the messages necessary for large scenarios involving tens or hundreds of thousands of objects. This requires that the simulation itself be designed within the current limitations of the hardware. Though this solution is very realistic, it creates a system that shows its age as computers evolve. Older simulations appear to be inadequate for current problems, when in truth they were the best solution possible at the time they were built.

3 MODELING

Though a military simulation is a complete system fitted for use in a larger world, the core of the system are the models which represent the existence and activities of the real world. This core is an area in which a great deal of experience and creativity is required to develop good representations. It is very difficult to arrive at a set of models for both the existence and activities of many objects that are appropriately balanced to address a particular problem. Decisions in every part of model design may effect the representations and operations of other models. Experienced modelers are very familiar with this effect and approach the design and development of a new model with a broad perspective. It is important to see conflicts as soon as possible to allow time, money, and man-power to correct them. Many problems survive the development and fielding of a simulation because they are either undetected, or detected too late to be remedied.

3.1 Fundamental Principles of Modeling

Even though every simulation is unique there are some principles which seem to apply universally to the activity of creating a model of the real world. The principles described here were derived from the experiences of several practitioners and certainly can not be the sum total of principles that exist.

The golden rule of modeling is that no model, no matter how accurate, has any inherent value of its own. The value of every model is based entirely upon the degree to which it solves someone's real world problem. Accuracy and fidelity are driven by the problem that the model is supposed to solve. A beautiful, elegant, exact model of a problem has no value if it is a model of the wrong problem (Smith 1998).

All of the intricacies and details involved in building a model often conspire to lead the modeler away from the intended problem and toward an adjacent or related problem. Though experience is probably the best defense against this habit, really understanding the problem from the user's perspective is essential. If the user or modeler has an incomplete or inaccurate understanding of the problem it is unlikely that the resulting model will address that problem correctly.

We have reached a period in which many models exist for the systems we are representing. Modelers should learn from the past by studying these models and determining the good points that apply to the new problem. Older models and modelers may not have solved problems of the complexity of the current system, but have valuable lessons to teach. The adage "those who are ignorant of history are doomed to repeat it" applies here. At the very least, the old models can be instructive in what not to do on new models.

Complex systems often require more detail than can be pictured mentally or uncovered through the design process. It is always valuable to build a model of the model – a prototype. These uncover subtle problems and provide a tool for experimenting with new ideas. A prototype can be an invaluable tool for communicating with the users of the system as well as clarifying areas that are vaguely understood.

Credibility or validation is not a totally objective determination. Each user or problem owner expects to see certain characteristics of the problem in the model. It is important that the model address these "hot buttons" in a clear and communicable manner. If the model falls short on these subjective criteria it will be very difficult for the user to accept the validity of more complex representations within it.

All models require some set of data upon which to operate. Data about all aspects of real systems is not currently, nor likely ever to be, available. Consideration must be given when designing the model to the availability of data to drive it. Even the data that is available is often incomplete, duplicitive, and conflicting. In this situation it is important to approach the modeling process fully prepared for these facts, but willing to accept a model under these limitations.

Finally, constructing a model is an activity subject to the universal constraints of time, money, and quality. The model will be finished when one of these resources is expended.

3.2 Physical Modeling

The military mission is usually focused on very physical operations and accomplishments. Therefore, most military simulations prominently feature the existence and interactions of physical objects. These objects include vehicles, people, and machinery involved in the activities of moving, perceiving other objects, and interacting with them (often quite violently). Military models have often been described as representing the process of "move-look-shoot" (JPL 1991). This basic sequence of events is reflected in the architectures of functional models that explicitly focus on these activities. More recent military models include intelligence dissemination and processing, logistics operations, communications, command and control, and other supporting activities.

Movement is governed by the need to accurately position units and vehicles through time. The basic equation RATE*TIME = DISTANCE is the beginning of many models. This is modified by information about the terrain, the enemy presence it is experiencing, and the level of damage previously done to the vehicles. Movement can also be effected by the need to maintain some formation among multiple vehicles and the urgency of the mission at hand. It is up to the modeler to determine which factors are necessary for each model.

"Looking", or sensor detection, includes characteristics of the sensor, the target, and the environment through which the detection is performed. A sensor usually has some effective range and field of regard. Within the area defined by these variables, some algorithm must be used to determine the level of detection achieved. The sensor may indicate the presence of an object, its location to some degree of accuracy, classification of the object, recognition of the type of object, or clear determination of the true identification of the object. Physical objects must include details that allow them to perform their primary function, but must also describe the object such that it can serve as a target for sensor systems. Information like the radar cross section, presented area, infrared signature, and physical dimensions may be necessary to support sensor modeling.

Engagement and attrition models represent the military penchant for violent interaction with opposing objects. These algorithms capture the effects of weapons on other objects. The application of these algorithms again requires that each object be viewed as a target for other systems, both sensors and weapons. The simplest, and most prominent, engagement modeling involves a set of tables that define the effectiveness of each weapon against each target. These tables may contain a scoring system for degrading the target or probabilities that a specific type of kill has occurred. Field tests have indicated that the most common types of "kills" are mobility, firepower, and catastrophic (M-Kill, F-Kill, and K-Kill respectively). These categories are often adhered to in engagement models. When attrition must be determined at a higher level of abstraction than individual weapon on target it is common to use some form of differential equation to apply the force effectiveness of each side to the other. The famous Lanchester equations are one instance of this method, as are Epstein equations.

Algorithms must be developed for a much larger set of objects and interactions than those provided here. The potential number and variety are almost uncountable. The descriptions of movement, detection, and engagement are provided because of their nearly universal presence in military modeling.

3.3 Behavioral Modeling

Because of its complexity, behavioral modeling has traditionally been very basic. The goal has been to provide military vehicles and units with the ability to react to basic events in the absence of human intervention. These models allowed aircraft on patrol to “decide” to return to base when getting low on fuel, rather than continuing until the aircraft falls to the ground. Ground units respond to enemy attacks by focusing firepower on the aggressor rather than blindly continuing their preprogrammed mission. Algorithms like these have been the extent of behavioral modeling for many years. However, more recent models have attempted to provide more reasoning capabilities to simulated objects. Most notable among these systems have been the Semi-Automated Forces (SAF) or Computer Generated Forces (CGF) systems that are used to stimulate virtual training audiences. These allow one operator to play the part of many vehicles or several platoons with the aid of embedded behavioral models.

The approach taken by most of these models is to replicate the product of human decision making, rather than the process. Since we do not completely understand the inner workings of the human mind, it is much more feasible to gather information about human reaction to certain situations than it is to represent the process of thinking about that information. However, research in the area of intelligent agents is leading to models of independent, emergent behavior derived from the interactions of multiple stimuli on an object.

Current systems make use of the following technologies from the artificial intelligence field to model human decision making: finite state machines, means-ends analysis, constraint satisfaction, expert systems, knowledge based systems, and traditional planning. Evaluations have been done on the applicability of Petri nets, Markov chains, case based reasoning, fuzzy logic, neural networks, genetic algorithms, and adaptive behavior. Each of these techniques has strengths and weaknesses for military decision making. Researchers familiar with both the simulation and AI fields are developing techniques specifically designed for this problem.

3.4 Environmental Modeling

The environment in which objects exist and operate has important impacts upon the outcomes of every operation. Some models represent the environment explicitly, others integrate its effects into the object models and interactions. In either case, it is necessary to understand the effects of this medium on the objects represented in the simulation. Though simulations exist in which the environment is the sole objective of the model, we treat it as a medium supporting other activities.

When environmental effects are included within the physical models described above it is often because the data describing those interactions was collected under specific conditions. Therefore, the model already accounts for one form of environment. Rather than extracting these effects from the collected data, the modeler may choose to match the simulated environment with one of the conditions under which the data was collected.

When the environment is represented independently and explicitly it is necessary to collect and manage a large volume of data. This data may include characteristics of the terrain surface, natural and cultural features, atmosphere, sea surface, sub-surface, and ocean floor. The representation of radio and acoustic energy, chemical and biological agents, and nuclear effects are also considered part of the environment since these create a medium within which the objects must operate. Characteristics in each of these categories must be collected or synthesized. This data may be very voluminous given the large areas over which military activity can take place.

3.5 Multi-Resolution Modeling

All models pose a multi-resolution problem. Each object is portrayed at a level appropriate for its interaction in the simulation. There is no universal set of levels that allow objects to interact without some degree of discontinuity. As the military has developed distributed, interoperable simulations this problem has grown in importance. Since different simulations do not represent objects and interactions in the same manner, achieving interoperability between them requires solving some form of multi-resolution problem. Some models represent a missile as a force that can be applied over some range and have a defined effect. Other models represent that same missile as a complex system in which the thrust motors, fuel volume, sensor seeker, warhead, and flight surfaces all play a part. Achieving interoperability requires supplementing the lower resolution model with more detail, eliminating detail from the higher resolution model, or performing some combination of both operations.

The classic constructive-virtual integration problem is one form of multi-resolution modeling. A virtual simulation may place each vehicle at a unique location with a specific orientation. A constructive simulation may aggregate information about dozens or hundreds of vehicles and place a single icon marker on the battlefield. If these two models are to interact in any meaningful way it will be necessary to generate data from one that can operate within the world view of the other, or to establish some middle ground that can support both views.

Different techniques and experiments have evolved to address these problems but none have been able to provide a general solution. Each solution appears to be specifically tailored for a known set of simulations. The success of multi-resolution modeling has many of the characteristics of the interoperability problem. It may be possible to create techniques that apply to a specific class of models that use similar representations of the world, but it is not likely that any one technique will suffice for all varieties of multi-resolution modeling that will be attempted. Standardization within each class of simulation would be a great aid in applying multi-resolution techniques to simulation systems (Smith 1998).

4 CONCLUSION

This tutorial has attempted to describe the techniques and knowledge base that are important for those who develop military simulations. There is currently no formally defined curriculum for learning the simulation art and science. Increased government funding for simulation projects and their growing presence in the commercial market makes the need for such a curriculum more evident each year. Practitioners in this field are currently crafted from the raw material provided by Engineering Schools, Colleges of Arts & Science, Business Colleges, and Military Institutions. This practice results in a very uneven education among practitioners and necessitates a great deal of on-the-job-training. Academic and commercial education could improve this situation through the organization of material into formal degree programs as well as a series of professional education courses.

This tutorial may outline the format for some part of such an education program. It is our hope that this will stimulate thought, conversation, and action toward the production of well-prepared simulation scientists.

REFERENCES

Davis, Paul K. 1995. Distributed Interactive Simulation in the Evolution of DoD Warfare Modeling and Simulation. *Proceedings of the IEEE.* Vol 83, No 8.

Defense Modeling and Simulation Office. 1997. Department of Defense High Level Architecture for Modeling and Simulation. DMSO. Alexandria, VA.

Fujimoto, Richard M. 1990. Parallel Discrete Event Simulation. *Communications of the ACM.* Association of Computing Machinery. New York, NY.

Jet Propulsion Laboratory. 1991. Corps Battle Simulation - Analysts Guide (3 volumes). California Institute of Technology. Pasadena, California.

Law, Averill M. and Kelton, W. David. 1991. *Simulation Modeling and Analysis*. McGraw-Hill. New York, NY.

Miller, Duncan C. 1995. SIMNET: The Advent of Simulator Networking. *Proceedings of the IEEE.* Vol 83, No 8.

Perla, Peter P. 1990. *The Art of Wargaming*. Naval Institute Press. Annapolis, Maryland.

Powell, Edward T. 1996. “The JSIMS Architecture”. *Summary Report on the 15th Workshop on Interoperability of Simulation Interactive Simulations.* Institute for Simulation and Training.

Sargent, R.G. 1987. “An Overview of Verification and Validation of Simulation Models”. *Proceedings of the 1987 Winter Simulation Conference*. Society for Computer Simulation.

Smith, Roger D. 1995. “Military Training via Wargaming Simulations”, *IEEE Potentials*, October/November.

Smith, Roger D. 1996. *Proceedings of the Electronic Conference on Interoperability in Training Simulation.* http://www.scs.org/confernc/elecsim/elecsim.html.

Smith, Roger D. 1998. *Military Simulation Techniques & Technologies*. Distributed Simulation Technology.

Wilson, Annette L. and Weatherly, Richard M. 1994. The Aggregate Level Simulation Protocol: An Evolving System. *Proceedings of the 1994 Winter Simulation Conference*. Orlando, Florida.

AUTHOR BIOGRAPHY

ROGER D. SMITH is the Technical Director for STAC Inc. and an Adjunct Professor at the Florida Institute of Technology. He is actively involved in designing, developing, and fielding constructive and virtual simulations. He is the Chairman for the ACM Special Interest Group on Simulation and a member of the editorial board *of ACM Transactions on Modeling and Computer Simulation*.