DEVELOPING SYNCHRONIZED PLAYER MODELS FOR EMBEDDED TRAINING

Vanna McHale and Wesley Braudaway Ph.D.
Science Applications International Corporation (SAIC)
Orlando, Florida

Abstract

The Synchronized Player Models (SPM) project supports the U.S. Army Inter-Vehicle Embedded Simulation Technology (INVEST) Science & Technology Objective (STO) Program [1]. The overall goal of the SPM project is to reduce the network bandwidth required to maintain synchronization between a Live vehicle, a Modular Semi-Automated Forces (ModSAF) player model simulation and its associated clone models in separate simulation environments. The SPM project conducted a series of experiments in order to determine the feasibility of the SPM objective. The first experiment, reported in this paper, focused on the ability to have computer-generated forces operate identically in separate simulation environments without requiring network communication. To obtain this level of synchronization it is necessary to have a *repeatable* ModSAF that provides simulation events (e.g., vehicle location events, firing events, damage events) that occur at the same simulation time in each run of the same scenario.

This paper discusses the use of repeatability to support synchronized embedded simulation and focuses on the modifications required to produce a deterministic, repeatable ModSAF. Experiments were conducted to test and demonstrate the repeatable ModSAF and are illustrated in this paper. These ModSAF modifications, that were developed in support of SPM, were the basis for developing the repeatability mode currently supported in the ModSAF version 4.0 baseline.

Authors Biography

Vanna McHale is a member of the Advanced Simulation Research Team within SAIC's Orlando Operation and a scientist on the Synchronized Player Models project. Ms. McHale received her B.S. in Computer Science from the University of West Florida and has been actively involved in the M&S community since 1992.

Wesley Braudaway, Ph.D. is a member and technical lead of the Advanced Simulation Research Team within SAIC Orlando's Operation. Dr. Braudaway was the Principal Investigator for the Synchronized Player Models project. He was also the System Architect for CCTT SAF and has been involved in several Computer Generated Forces related research and development projects. Dr. Braudaway received his Ph.D. from Rutgers University's Computer Science Department and has been actively involved in the M&S community since 1991.

DEVELOPING SYNCHRONIZED PLAYER MODELS FOR EMBEDDED TRAINING

Vanna McHale and Wesley Braudaway Ph.D.
Science Applications International Corporation (SAIC)
Orlando, Florida

1. INTRODUCTION

Simulation technology advances can be leveraged into a form suitable for embedding into ground vehicles for training, mission planning, and other operational uses. Embedded Training (ET) is a capability designed into or added onto operational hardware and software systems that enables it to provide the simulation cues necessary to train crewmembers. This on-board technology will allow mission rehearsal and sustainment training to occur whether the soldiers are at home stations or deployed.

As part of this simulation environment, collective operation requires the synchronization of multiple embedded simulations. Using today's distributed interactive simulation technology, the volume of data transfer required to support embedded training at the unit and battalion level is a significant obstacle because of the network and communication limitations of the fielded systems. Typically, the fielded systems rely on wireless communication, which provide very low bandwidth for simulation use. The Simulation, Training and Instrumentation Command (STRICOM) is conducting the Inter-Vehicle Embedded Simulation Technology (INVEST) Science and Technology Objective (STO) to address the technologies required to provide this ET capability [1].

Providing deterministic Computer Generated Forces (CGF) as part of the simulation environment solves part of the ET problem by removing the need for communication to synchronize the computer-generated models. Suppose each simulation environment has its own deterministic CGF to simulate all computer-generated models. The simulation environments will produce exactly the same simulation event sequence for the same scenario without requiring any coordination. If the input to these CGFs is synchronized then there is no need to synchronize the computer-generated parts of the simulation environments as done today using the Distributed Interactive Simulation (DIS) protocol.

The SPM project chose as its simulation platform the Modular Semi-Automated Forces (ModSAF) system. This paper describes the necessary modifications to implement a deterministic or *repeatable* ModSAF. This paper is organized to describe the synchronization challenge for embedded simulation, the solution to synchronization using a repeatable CGF, the effort required to make ModSAF repeatable, and the remaining effort required to complete the synchronization of multiple embedded simulations.

2. PROBLEM DEFINITION

The INVEST objective is to provide a simulation environment for both individual vehicle operations and multiple vehicles that interoperate within a collective simulation. In the collective mode, the simulation environments (one for each live vehicle) must be synchronized to present an identical synthetic situation to each vehicle concurrently.

There are two types of synchronization required for this modeling (see Figure 1).

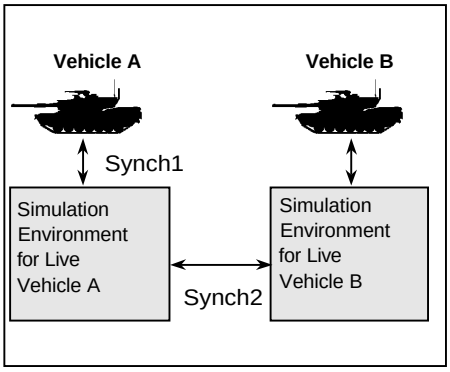

Figure 1: SPM Synchronization

This first synchronization activity occurs between the live Vehicle A and its simulation environment. The interaction of the vehicle and its simulation environment is achieved by a direct coupling of the vehicle's vision blocks and controls to the simulation environment. A virtual model in the simulation environment representing the live vehicle, called the player model, replicates the behaviors of an actual vehicle under the command and control of its crew. Any interaction between the live vehicle and simulated entities is achieved as a side effect of the interaction between the player model for the vehicle and the simulated entities within the simulation environment.

The second synchronization occurs between Vehicle A's simulation environment and Vehicle B's simulation environment. Contained in the simulation environment are CGF (vehicles, units, munitions, etc.) and a player model representing each live vehicle participating in the collective simulation. The synchronization activity makes each environment identical as defined by state data and events affecting each virtual entity regardless of whether they are computer-generated or player models. For example, in addition to replicating live vehicle A's simulation environment, a "clone" of that player model must exist within vehicle B's simulation environment that replicates that same behavior. This synchronization is implemented today using the DIS protocol and communication channels requiring very high bandwidths. However, in meeting the objectives of INVEST STO, high bandwidth and physical connectivity are not feasible alternatives.

3. SYNCHRONIZATION THROUGH REPEATABILITY

The CGF of an embedded simulation can be synchronized by ensuring that each simulation environment is started at the same time and that the computer-generated models process the same events with respect to time in each simulation environment. Assuming that all other aspects of the simulation environments are synchronized, synchronization of the computer-generated models is achieved using a repeatable implementation of the CGF in each environment. Each CGF replicates the exact simulation of all computer-generated models in each simulation environment.

A CGF implementation is *repeatable* if the simulation events (e.g., vehicle location events, firing events, damage events) occur at the same simulation time in each run of the same scenario (where "scenario" is defined as a set of initial conditions and external events). By satisfying this requirement, the CGF implementation within each embedded simulation environment will result in a synchronized simulation environment as long as the initial conditions are the same, the simulation time is synchronized and the external events are synchronized. No additional communication will be necessary to synchronize the CGF models in each simulation environment.

4. IMPLEMENTING ModSAF REPEATABILITY

Although CGF implementations, such as ModSAF, are not typically designed to be repeatable, they can be modified to provide a repeatable mode of execution. Repeatability can be achieved by modifying the scheduling of simulation events, the generation of stochastic events, and the elimination or control of distributed events.

4.1 Event Scheduling

Many CGF implementations, including ModSAF, are event simulations that are managed to ensure a perceived real-time performance. They are designed to emulate a real vehicle by implementing a model that performs as close to the vehicle's actual real-time performance as possible.

A model's behavior is implemented as discrete event changes maintained in an event queue, such as changing its location over time, firing weapons, or taking damage. Each event in the queue is executed relative to a simulation time that is also updated periodically with respect to real-time. An attempt is made to maintain a simulation time equivalent to real-time so that the model's performance appears to correctly emulate an actual vehicle's real-time performance. This differs slightly from discrete event simulations where events in the queue are executed at an explicit, predetermined simulation time.

While the generation and execution of simulation events are deterministic and repeatable, the synchronization of simulation time to real-time is

not repeatable. As the operating system interrupts the simulation to make system service calls at different times and for different periods from simulation run to simulation run, the execution of events with respect to real-time (relative to the start of the simulation) will also vary. Because the discrete events are a sampling of continuous real-time and therefore an approximation of real-time, the outcome of an event may vary if a different sampling occurs between the runs. For example, consider the same movement event for a vehicle in two different simulation runs (as shown in Figure 2).

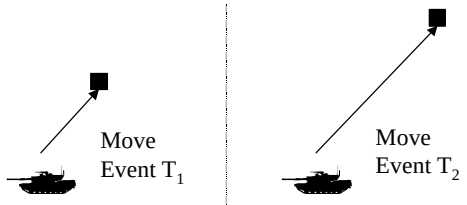

Figure 2: Same Discrete Event in different SRs

Because real-time advanced further in SR2 (possibly due to a longer system interrupt), the vehicle traveled farther during this event in SR2 than in SR1 in order to maintain the real-time approximation. Suppose that this update in both runs placed the vehicle within line of sight of an opposing vehicle and that opposing vehicle reacted immediately by firing its weapon. Because this observation and reaction occurred at different simulation times within the two simulation runs, the executions are no longer identical (i.e., not repeated).

Because of the great dependency between events, as a single event between the simulation runs diverges with respect to time, the differences between the runs will cascade until there are significant variations in the simulation outcomes of each.

To provide a repeatable mode for ModSAF, its event scheduling mechanism was modified. These modifications included changes to the clock and scheduler implementations, and the scheduling of behavior models.

4.1.1 Clock Implementation

To support real-time simulation, the ModSAF real-time clock and its simulation clock were tightly interconnected. As stated earlier, when running in a real-time mode, the simulation clock is dependent on the real-time clock and is advanced with respect to the real-time clock. However, what was unusual about the ModSAF implementation was that the procedure to advance the simulation clock also always advanced the real-time clock. The ModSAF clock implementation was modified to remove this reverse dependency.

In ModSAF's repeatable mode, ModSAF's real-time clock continues to be based upon the system clock. However, the simulation clock was modified to advance to the next event's time on the event queue after processing each event. The continuous frame update rate for the simulation clock was disabled, essentially severing the simulation clock to real-time clock dependency.

4.1.2 Scheduler Implementation

A thorough review of the ModSAF scheduler was conducted to determine areas resulting in random scheduling and/or invoking of events and function calls. ModSAF's scheduler implements four queues: a high priority real-time queue, a periodic real-time queue, a deferred real-time queue, and a simulation time queue. All four queues utilize the real-time clock to invoke events on the queues. The simulation queue implementation was altered to be event driven and based upon the simulation clock rather than the real-time clock to produce a repeatable ModSAF mode. Therefore, when a simulation queue's event is invoked in this mode, the simulation clock is advanced to the time of the next event on the simulation queue.

4.1.3 Scheduling Behavior Models

To utilize the modified, event-driven simulation clock, the scheduling of behavior and physical model update functions was moved from the real-time queue to the simulation queue. All non-simulation functions such as user interface and network functions remained on the real-time queues.

The simulation processing of each vehicle occurs in that vehicle's "tick" or update function. All vehicle events are generated as part of this tick

and therefore, have the biggest influence on repeatable performance. Moving this function and its associated unit behaviors to the simulation queue will provide the required ModSAF repeatability by removing them from the influences of the real-time clock. No other modification to the models or the modeling architecture was required.

4.2 Stochastic Events

Many CGF systems, including ModSAF, include an implementation of stochastic events. For example, given a vehicle is hit by munitions, it may lose its ability to move, it may lose its ability to shoot, or it may be totally destroyed depending of some statistical representation of the chances for each alternative. These events are implemented using algorithms that are based on a number taken from a random number sequence. A repeatable random number sequence is easily implemented by using the same random number seed between simulation runs.

ModSAF's random number seed was not initialized in one central location and was corrected as part of this effort. ModSAF was modified to provide initialization within the random number generator library to provide a constant random number seed for ModSAF's repeatable mode.

4.3 Distributed Events

The distribution of simulation events on a network also impacts repeatability since it is difficult to guarantee the execution time of delivered events between simulation executions. This is due to network latency, variable order of network packets, and the variability in times at which the operating system services its network operations. A repeatable CGF could be achieved either by not allowing distributed simulation events or by explicitly controlling the network events. Controlled network events can be achieved using a reliable network mechanism and some reliable time management capability that alleviates the problem of network latency and system network management.

In ModSAF's repeatable mode, this problem is avoided by only providing repeatability in a non-networked mode.

5. MODSAF REPEATABILITY EXPERIMENT

ModSAF's repeatability was confirmed by experimentation with several scenarios to demonstrate that ModSAF's repeatable mode generates a duplicate simulation outcome for the same scenario. For a particular scenario, two separate ModSAF executions were run to collect data. The data was collected to confirm that identical location update, fire, and damage events occurred at exactly the same simulation time in successive runs. Several other scenarios were used to determine the success of the repeatable ModSAF implementation relative to a variety of behaviors. Data collection and analysis was performed for the following areas:

Scheduler Analysis to determine which function calls were placed on the scheduler, the number of calls made to the functions, and the cumulative processing time.

Queue Scheduling to identify the function scheduling sequence for the deferred, periodic and high priority real-time queues and scheduled simulation time queue.

Random Number Generator to analyze the random number generator activity, to determine its calling functions and to determine the calling event simulation time.

Vehicle Specific Data to gather the vehicle's position (x and y coordinates) and to obtain its damage assessment with respect to simulation time.

Using this data collection, graphs were created to represent and assess ModSAF's repeatable performance. The first graph (see Figure 3) compares the position variation for the same vehicle between two runs of the non-repeatable ModSAF. Position variation is defined as the distance between two instances of the same vehicle at the same simulation times between the two runs of the same scenario. This graph shows the cascading effect of event differences over time between two runs.

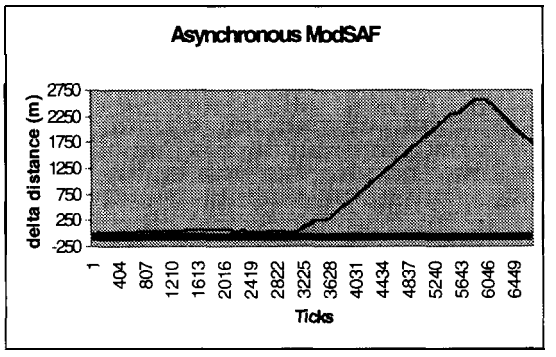

Figure 3: Vehicle Location Difference, Non-Repeatable

The second graph (see Figure 4) shows the same change in position for the same vehicle from one run to another using ModSAF's repeatable mode. This graph illustrates that the vehicle's position throughout the entire scenario remains consistent from run to run (i.e, the distance between the vehicles instances within each run is always zero).

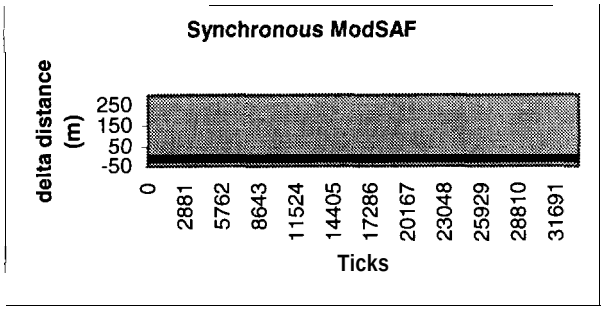

Figure 4: Vehicle Location Difference, Repeatable

In addition to location data, damage assessment data was also analyzed to determine repeatability of fire and detonation events and overall vehicle damage. The third graph (see Figure 5) shows the damage assessment data collected for all eleven vehicles within the scenario. Each run is illustrated as a separate series to show that within the non-repeatable ModSAF, no two events happen in the same manner at the same simulation time (time events in the graph).

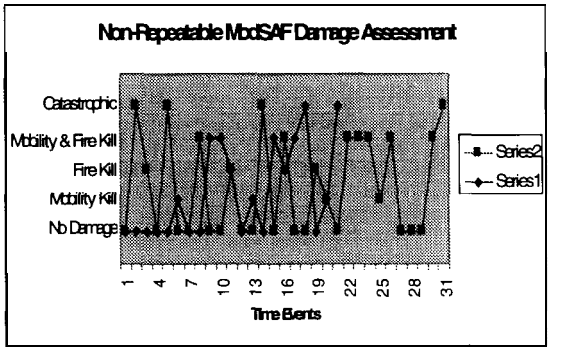

Figure 5: ModSAF Damage Assessment, Non-Repeatable

The final graph (see Figure 6) represents the damage assessment data using the ModSAF repeatable mode. In this figure, the two runs are individually graphed, showing that run 1 is entirely overlaid with the data from run 2. This illustrates that over the entire scenario, all vehicles received the same damage at the same scenario time, from one run to another.

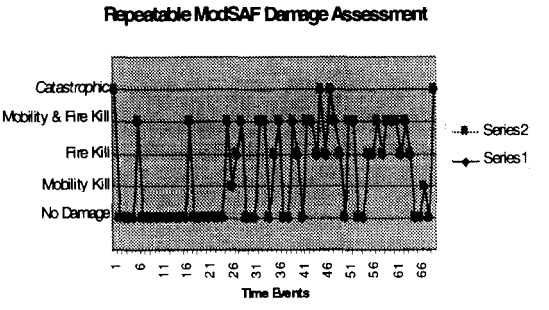

Figure 6: ModSAF Damage Assessment, Repeatable

Numerous other scenarios were executed to test the robustness of the ModSAF repeatability mode. These scenarios incorporated a variety of additional behaviors and obstacles. In each case data collected about the simulation events was consistent from one run to the other, ensuring that repeatability was attained within ModSAF.

6. FURTHER SPM EFFORT

The remaining SPM project goals [2] address the synchronization between a live vehicle and its player model, and the communication of behavior control updates. Current SPM activities are

focusing on continued experimentation to refine the understanding of the player model control interface, behavior parameterization, and overall bandwidth issues. Additional experiments are being developed to investigate the capability to synchronize the simulated models using the state changes of vehicle behavior parameters rather than vehicle location updates. It is believed that this method of synchronization combined with a repeatable CGF will substantially reduce the bandwidth required to synchronize a distributed simulation of player models.

In future phases, additional experiments will be performed to identify and implement the control between ModSAF and the behavior control interface based upon the INVEST STO operations identified by the INVEST Architecture Working Group.

The ongoing SPM experimentation will also consider alternative behavior control strategies within ModSAF to control the synchronization of player model and its associated clone models.

The results of current and future experiments will influence the final SPM architecture and implementation. It is intended that the SPM prototype will be integrated into a developed ET architecture whose objective is to demonstrate the viability of the embedded simulation approach.

7. CONCLUSIONS

The objective of the SPM architecture is to provide Computer Generated Forces that will interact with live vehicles through use of a player/clone model approach. This objective must be achieved while using a lower bandwidth than the current DIS protocol approach for implementing simulation environment synchronization. To achieve this objective, it is believed that dead reckoning at the behavior abstraction level and reducing the requirement for communication to achieve synchronization will reduce the bandwidth [2].

Through the use of a repeatable CGF, SPM achieves synchronization of the computer-generated models in different simulation environments without resorting to DIS protocol. This approach assumes that the scenario (inputs and distributed events) implemented in each environment is synchronized. A repeatable CGF is achieved by scheduling the simulation events independent of real-time, generating stochastic events from a fixed random number sequence, and either eliminating or control the processing of distributed events. The results of this effort were fully integrated into ModSAF version 4.0 to provide a ModSAF with a repeatable mode.

8. ACKNOWLEDGEMENT

The authors would like to acknowledge the efforts of Deanna Nocera, Gene McCulley and Eddie Cason for their contribution to this analysis and repeatable ModSAF development.

The authors would also like to acknowledge the efforts of Derrick Franceschini and Gene McCulley of the Advanced Distributed Simulation Technology (ADST - II) program for their complete integration and testing of the repeatable mode into the Army's ModSAF version 4.0 baseline.

The work presented in this paper was sponsored by STRICOM under contract N61339-97-C-0040.

9. REFERENCES

[1] Bahr, H.A., "Embedded Simulation for Ground Vehicles," Spring 97 Simulation Interoperability Standards Workshop Proceedings, Institute for Simulation & Training, Orlando, FL, March 1997.

[2] Braudaway, W., and Nocera, D.L., "Synchronized Player Models for Embedded Training," Spring 98 Simulation Interoperability Standards Workshop Proceedings, Institute for Simulation & Training, Orlando, FL, March 1998.