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

THE AGGREGATE LEVEL SIMULATION PROTOCOL: AN EVOLVING SYSTEM

Annette L. Wilson
Richard M. Weatherly

The MITRE Corporation
7525 Colshire Drive
McLean, Virginia 22102, U.S.A.

ABSTRACT

The Aggregate Level Simulation Protocol (ALSP) concept was initiated by ARPA in January 1990, the first laboratory demonstration took place in January 1991, and the first fielding in support of a major military exercise took place in July 1992. Since then, the ALSP confederation of models has grown from the original two members to six. In support of this growing confederation, the ALSP Infrastructure Software (AIS) has evolved from its fundamental functionality to the current focus on improved confederation management and performance. This paper describes the evolution of the AIS from the initial prototype to the present, emphasizing the discovery of new requirements and how they were accommodated.

1 INTRODUCTION

The ALSP concept was initiated in January 1990 when ARPA sponsored MITRE to analyze the distributed wargaming process with the goal of generalizing and systematizing the design of constructive simulation interfaces. MITRE approached the task by analyzing aggregate-level simulations in use at the Warrior Preparation Center (WPC) in Einsiedlerhof, Germany; studying the results of ARPA's first wargame distribution efforts at the ACE '89 exercise; and by analyzing the successful SIMNET, now DIS, distributed interface scheme. The results of this investigation were the initial set of ALSP requirements and a set of promising principles which could be applied to the distributed aggregate-level simulations interface problem (Seidel 1993).

The ALSP project did not set out to design a complete, distributed, constructive-simulation interface in the laboratory. Rather ALSP sought to evolve to such an interface through the discovery and generalization of problems encountered while building a confederation of cooperating, combat, training simulations. ALSP has successfully balanced the flexibility demands of rapid prototyping with the stability requirements of a fielded system. The AIS has supported major combat training exercises for two of its four years while continuing an aggressive functional evolution fueled by this real world experience.

2 LABORATORY EXPERIMENTS

After analysis of the ACE '89 experience, it was decided that a laboratory prototype using existing simulations should be built to demonstrate the concept. This was both a technical and political decision: results derived of real systems have greater credibility in the wargaming community and are more likely to address practical concerns than those derived of pure academic research. The Warrior Preparation Center provided the Ground Warfare Simulation (GRWSIM) for use in these experiments.

A major requirement revealed in the ACE '89 exercise was the need to properly coordinate the advance of simulation time between elements of the distributed GRWSIM simulation. Many anomalies observed in ACE '89 were traceable to simulation events being processed in an order that was more influenced by the speed of computers and communication links than by the order the events occurred in simulation time. To alleviate these anomalies, ALSP's first objective was to provide distributed time synchronization services.

To expedite the pace of experimentation, an existing tool, the Simulation Control Executive (SCE), was used to provide conservative time synchronization. The SCE is infrastructure software designed to facilitate the development of simulations in the Ada programming language (Maginnis 1988). The SCE passes event messages between and schedules the execution of user supplied simulation elements. To take advantage of the SCE's capabilities while maintaining the integrity of the actual wargame software, ambassador processes were designed. These ambassador processes looked to the SCE like normal simulation elements. The SCE was programmed to deliver all message received from one ambassador to all other ambassadors. The ambassador processes (shown as the circles with T's in Figure 1) did not actually represent part of a simulation (often referred to as an actor when participating in an ALSP confederation) but acted as message gateways and execution throttles. The ambassador would block the actor's attempt to advance simulation time until the SCE determined all other actors were in the proper state and transport messages between the machine hosting the SCE and the actor's host. The isolation of the SCE and the ambassador processes from the activities above them revealed the need for a layered protocol within ALSP.

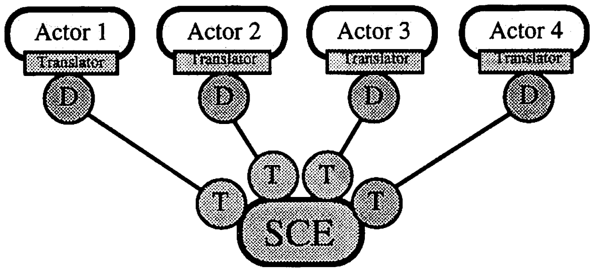

Figure 1: Initial ALSP Architecture Utilizing the Simulation Control Executive

The other end of the connection between the actor and the SCE-based synchronization and message distribution service is a process (shown as the circles with D's in Figure 1) that manages the inter-host communication on one side and provides a procedural interface to the actor on the other. As shown in the figure above, the portion of the actor that has been modified to translate internal data representations into confederation representations and vice versa is referred to as the translator. As experimentation continued, the need for more than simple time management services became clear. The ALSP object and object interaction model were becoming richer and, as a result, the need for additional common services was growing. All these services assumed a confederation of time synchronized models. These new data management services formed a natural layer above the time management services.

With the emergence of the data management services and the desire to move away from a centralized time management system, the ALSP architecture was modified in January 1991 as depicted in Figure 2 below. In the intermediate ALSP architecture, the time management services were distributed as identical processes in the ALSP Time Manager (ATM). The ALSP Broadcast Emulator (ABE) provided the data replication services available in the SCE for local-area networks while waiting for the reliable multicast services to be provided by the Defense Simulation Internet (DSI) for wide-area networks. The ALSP Data Manager (ADM) continued to expand incorporating services such as managing a distributed object database across the confederation of actors.

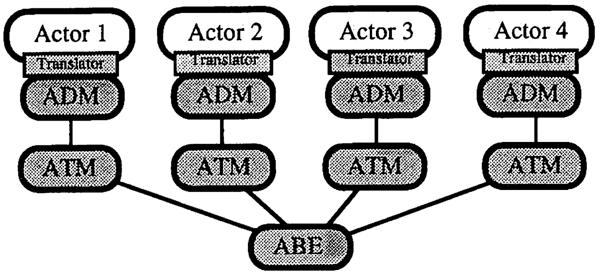

Figure 2: Intermediate ALSP Architecture with Distributed Time and Data Management

This architecture was used in a laboratory demonstration linking the Corps Battle Simulation (CBS) and the Air Warfare Simulation (AWSIM) in September 1991 for ARPA and members of the combat training community. An air-ground interface was demonstrated that proved the essential features of time synchronization, coordination of access to common simulated objects, and the interaction of objects controlled in separate simulations.

3 INITIAL FIELDING

The success of the laboratory demonstrations prompted USAREUR to request the use of ALSP in REFORGER '92. To ensure that the software was ready to support a major exercise, a functional test was scheduled for early March 1992: this left less than 6 months to turn the prototype into exercise quality software. To further complicate matters, it was determined that the procedural interface to the ALSP software with its callback routines caused problems when used with certain simulation languages.

The next step in the evolution of the ALSP architecture was to remove the procedural interface between the ADM and the actor and replace it with a message based interface. The ADM was joined with the ATM to form a new process, the ALSP Common Module (ACM). The ACM and the ABE became known as the ALSP Infrastructure Software (AIS). The new AIS design provided for the most flexibility since it no longer restricted the languages or operating systems of the confederated simulations. This first version of the

AIS ran under the VAX VMS operating system and could communicate with the actors using either DECnet or TCP/IP based communications. Figure 3 below shows a confederation of four actors using the AIS software.

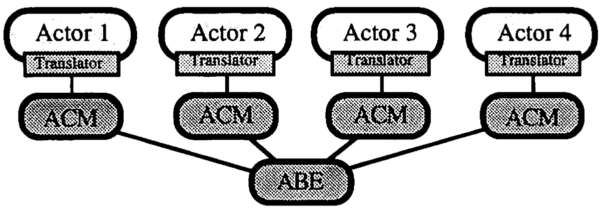

Figure 3: Current ALSP Architecture

The functional test held at the WPC in March 1992 demonstrated the flexibility of the new architecture. In addition to the two simulations (AWSIM and CBS), a graphics display program (the GIAC) participated in the confederation. The two simulations ran on VMS machines and communicated with their ACMs using DECnet, while the graphics program ran on a DECstation workstation and communicated with its ACM using TCP/IP. Two configurations were run during the test. In one configuration, the entire confederation was run at the WPC. In the second configuration, both AWSIM and GIAC were run at the WPC while CBS was run at 5th Corps in Frankfurt. In the first configuration, the AIS was communicating through a single ABE using DECnet. In the second configuration, ABEs were run at each site and communicated with each other using TCP/IP over the DSI with the ACMs using DECnet to communicate with the ABE located at their site.

During this test, two additional operational requirements for the ALSP software were recognized. The new requirements arising from the test were 1) the need to simultaneously save the state of all confederation components and 2) the need to provide multiple displays for each ALSP process. Although each actor could save its state independently, it was discovered that in order for the confederation to return to a consistent state, the saves of all actors and their ACM states needed to be coordinated. A manual procedure was put in place for the remainder of the functional test with the mandate to automate this procedure in the future. The configuration with the confederation distributed between two sites demonstrated the need for multiple displays on the AIS. In addition to the AIS controller at the WPC, the CBS controller in Frankfurt wanted a view of the current state of the confederation. This was not possible with the original version of the AIS.

The number of changes required of the translators and the AIS coupled with the desire to use the confederation to support Central Fortress in July precipitated a second test, the Follow-On Joint Test (FOJT), in May. Both the confederation save mechanism and the ALSP Control Terminal (ACT) which allows for multiple displays of each AIS process were designed and implemented in the two months between the functional test and the FOJT. The confederation consisting of AWSIM and CBS was successfully tested at the FOJT. This confederation was used to support three exercises (Central Fortress, Ulchi Focus Lens (UFL), and Reforger) during the first year.

4 MANAGEMENT TOOLS

The experience at the first functional test indicated the importance of management tools. As the confederation has grown and matured, so have the management tools available to the confederation manager, the person who provides primary control for all ALSP components. The ACT, developed as a result of the original functional test, provided the means for people at multiple locations to simultaneously view the state of the confederation. To have a complete picture of the confederation, the confederation manager was required to have an ACT for each AIS component running. The Confederation Monitor Screen, which allows the manager to receive a summary of the entire confederation on a single screen, was the next step in management tools. To control the AIS components, the confederation manager was required to have an ACT for each AIS component. Allowing the manager to control all AIS components from a single application, the Confederation Management Tool (CMT) is the latest development in management tools.

4.1 ALSP Control Terminal

The ACT is used to monitor and control the ALSP software. The ACT may run on the same computer as or a different computer than the AIS component (ACM or ABE) which it is controlling or monitoring. The ACT sends commands and receives state information from the component through a DECnet or TCP/IP connection. As shown in Figure 4 below, up to four ACTs may be connected to any component.

The ACT-based design of the AIS control has several advantages. First, by designing the operator interface to the ABE and ACM as separate processes, the actual ABE and ACM processes are freed from ties to the display device. Thus, if the display device were to go down (someone accidentally unplugged the terminal), the ABE or ACM would continue to function. Second, the AIS may be monitored from several locations. Each location may select from several operational views of the confederation provided by the ACT. Finally, the protocol between the ACT and component has matured so that other processes, such as the CMT, may attach to an ACM or ABE as though they were an ACT.

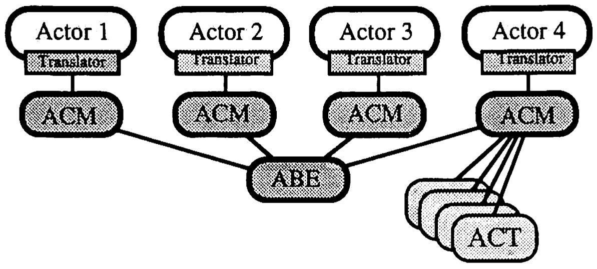

Figure 4: Four ALSP Control Terminals on a Single ACM

The ACT provides several operational views of the confederation depending on the component to which it is attached. The Confederation Monitor Screen, discussed in the following section, is available when attached to either an ACM or ABE. The ACM Monitor Screen and Actor Monitor Screen are available when monitoring an ACM, while the ABE Monitor Screen is available when monitoring an ABE. The ACM Monitor Screen displays detailed information about the state of the ACM internal processes, the Chandy-Misra queues, and other status and performance information required to diagnose and repair faults. The Actor Monitor Screen indicates the ACM's perception of the actor's state including the number of ghosted and owned objects tabulated by object class and information about the actor's progress in both simulation and real time. The ABE Monitor Screen displays the types of messages sent by source and the number of messages held for each destination ACM or ABE.

4.2 Confederation Monitor Screen

The Confederation Monitor Screen, which can be displayed on any ACT, was the first step in providing a single screen to monitor the entire confederation. This screen grew from the operational need for the confederation manager to tell at a glance that the confederation was advancing normally. Should there be a problem, the Confederation Monitor Screen also provides some indication of where the problem may be found.

The Confederation Monitor Screen is composed using status messages which are periodically broadcast by all components of the confederation. ACM status messages contain information about the actor's state, while ABE status messages contain the list of ACMs attached to it. Each component uses the status message transmission time and content to build a table of system-wide communications activity and performance. Placing the status messages in the regular message stream allows the system to compute a latency measurement for the communications path. However, since the messages are routed through the ABE and the confederation can be composed of a number of linked ABEs, failure of a single link can cause a loss of visibility into a significant portion of the confederation.

4.3 Confederation Management Tool

The Confederation Management Tool grew from the need to control the confederation from a single application. Until the development of the CMT, the confederation manager had to maintain an ACT for each AIS component. As the confederation grew this task became increasingly difficult. For a confederation of six actors connected over a wide area network through two ABEs (as might be found in a 1994 Confederation), the manager must monitor eight ACTs. It is virtually impossible to view eight ACTs simultaneously from a single workstation.

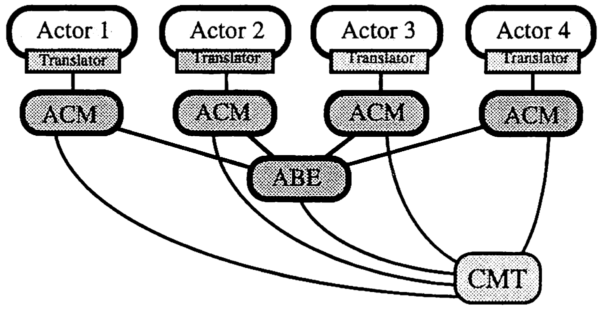

Figure 5: The Confederation Management Tool

Whereas the Confederation Monitor Screen receives its information from the normal message stream through the ABE, the CMT monitors the confederation through direct connections to each component, as shown in Figure 5 above. The CMT has the ability to issue a single command and have that command sent to each component in the confederation (e.g. Restore from save) or to function as an ACT for any component to which it is attached. When the CMT is functioning as an ACT, all of the usual ACT monitor screens and commands are available. The CMT thus provides the capability to control the entire confederation from a single point. The CMT also provides four views of the confederation: the Temporal Monitor Screen, the Object Monitor Screen, the Communications Monitor Screen, and the File Monitor Screen.

The Temporal Monitor Screen provides a summary of the information available on the Actor Monitor Screen of each individual ACM ACT and that provided by any

Confederation Monitor Screen. This includes, the amount of real time since each actor advanced simulation time, the length of time since each ACM received a message from its actor, a running average of the ratio of real-time to simulation-time, a percentage measure of each actor's utilization, and other time related information. The Temporal Monitor Screen allows the manager to determine at a glance whether the confederation is moving forward at the desired rate and, if it is not, an indication of which actor is responsible for the delay.

The Object Monitor Screen displays the number of owned and ghosted objects, by class, for each member of the confederation. Object ownership information for a single actor was previously available at each individual ACT Actor Monitor Screen. Now it can be viewed for the entire confederation on a single screen. To provide a uniform basis for comparison, the CMT forms the union of all the object class hierarchies used in the confederation. This unified class tree allows a meaningful juxtaposition of individual actor's object counts.

The Communications Monitor Screen displays the status of communications links in the confederation. For each link an item is displayed on the screen which shows the computers and components associated with that link. Alarm conditions are added to the display as they arise; they are: excessive buffer utilization on either end of the link, insufficient timeliness of link status information, loss of link connectivity, excessive link latency, and existence of conflicting connection parameters (communications protocol or host location).

The File Monitor Screen displays the status of all log and statistics files. For each open file, the amount of disk space remaining on the device is displayed. Additionally, the number of messages buffered for log files is displayed. The top half of the screen displays the status of ACM files while the bottom half displays the status of ABE files.

5 PERFORMANCE ENHANCEMENTS

As the size and capability of the confederation has increased, the number of messages that must be processed and distributed has also grown. To deal with this increased load, several performance enhancements have been made to the AIS. These enhancements can be grouped into three categories: 1) reductions in the number of messages that are transmitted, 2) reductions in the number of unnecessary messages that an actor must receive, and 3) optimizations to the code which analyzes and composes the messages.

5.1 Traffic Reduction

In the original design of the AIS, the ABE sent all messages to all ACMs. The ACMs then filtered the message stream based on criteria proved by the actor. While this approach had been sufficient, it was clear that as the number of actors in the confederation increased, the strategy of sending all information to all actors would fail in the face of an N^2 growth in demand for communications and processing. To address growth questions associated with the assumptions of reliable, order preserving multicast provided by the ABE, MITRE undertook the Event Distribution Protocol (EDP) research project in 1993 (Weatherly 1993).

The project focused on determining a more efficient means for distributing messages. It seemed that by making the ABE smarter about where it sent messages, the problem could be solved. The ABE would have to expend more computational resources to look inside each message to determine where it needed to be sent, but this would pay for itself by reducing the number of I/O operations the ABE must perform.

The result of the EDP project was a mechanism to distribute the object classes and attributes which were needed by each actor. This allows the ABE to decide, based on the class of an object, whether or not an ACM should receive the updates on that object. Further savings are achieved by sending the actor only the portion of the object update that is relevant to it. The mechanism developed as part of the EDP has been incorporated into the AIS and is in use in the 1994 Confederation.

5.2 Improving Actor Performance

One of the concerns of the ALSP design is limiting the overhead burden on an actor associated with participation in the confederation. Two sources of unnecessary overhead are calculations required of an actor which would not be required if the object were simulated within the actor and identification of irrelevant messages. Examples of mechanisms which have been incorporated into the AIS to reduce each case are a dead reckoning mechanism and interaction filtering, respectively.

A dead reckoning mechanism was introduced into the ALSP confederation prior to the first functional test. The need for dead reckoning was recognized during early experiments with CBS. During these experiments, it was realized that CBS must recompute radar detections whenever it receives an update message for an aircraft. With dead reckoning in place, CBS can predict the location of the aircraft in the future and calculate radar detections based on these predictions. Only when the course of the aircraft changes does CBS need to recompute the radar detections, thus, increasing simulation performance.

Until the 1994 Confederation, each actor received every message specifying interactions between objects. It was then up to the actor to decide whether the interactions were relevant to it. With the increased size and functionality of the confederation, actors were required to discard more and more irrelevant interaction messages. The AIS has been modified so that actors may specify the types of interactions (e.g. air-to-ground) they are to receive in a manner similar to the one used to specify the class and attributes of objects. EDP techniques are applied to distribute the interaction criteria within the AIS software. Thus, unnecessary interaction messages are eliminated from the confederation at the earliest possible point. As an example of the savings achieved by this mechanism, consider an air-to-air engagement interaction of interest only to AWSIM and RESA. Using the original methodology, the interaction message would be sent from the initiating actor (say AWSIM) to its ACM, from the ACM to the ABE, from the ABE to the five other ACMs (such as might be found in a 1994 confederation), and finally from the five ACMs to their respective actors. Using the new methodology, the initiating actor (AWSIM) would send the message to its ACM, the ACM to the ABE, the ABE to the RESA ACM and finally to RESA which is the only actor interested in air-to-air engagements. Though not all message traffic is reduced so dramatically, this example demonstrates the potential for traffic reduction and elimination of messages irrelevant to an actor.

5.3 AIS Code Optimizations

As the AIS has matured and functionally stabilized, efforts have been made to increase its performance. The most frequently used portions of the software have been identified and optimized. As would be expected with a message-based system, the most frequent activity of the AIS is the composition, parsing, and analysis of the messages. These routines were the first analyzed and modified to achieve maximum efficiency. Since then other areas have been identified and modified to achieve better performance. One such modification is the mechanism used to perform ACM saves. The original save mechanism created a file that was editable and could be used for diagnostic purposes. The new mechanism, which no longer creates editable files, decreases the ACM state save time by more than 60 percent in many cases. A small, off-line tool was introduced into the AIS which converts ACM save files to a human readable form should an operator wish to view them As a result of changes like these, a set of software has been developed with significantly more functionality than the early versions while achieving the same performance with a larger confederation producing more messages.

6 CURRENT CHALLENGES

The ALSP confederation has been able to successfully meet the needs of the combat training community by providing the flexible infrastructure necessary to accommodate changing requirements. Confederation growth and exercise experience continue to influence the evolution of the AIS. Current challenges to ALSP confederations are reductions in the time and expense of the validation process, increased understanding of inter-simulation and inter-object behavior, and further reductions in operator workload. A brief description of initiatives in these areas follows.

During the Prairie Warrior '94 exercise, inconsistencies were detected in the distributed object database. These inconsistencies were traced to communications problems and procedural errors. A new mechanism is under development that will continuously detect, report, and correct most attribute-ownership database inconsistencies as they occur.

While the basic ALSP time management services guarantee the best possible conservative time synchronization, the instantaneous temporal relationship between individual simulations and their objects is not always clear to a human analyst. New reports are being added to the AIS and a new tool is being constructed that will help clarify these relationships.

The ability to remotely start all AIS components is under investigation for incorporation into the 1996 AIS. As the confederation has grown, the task of manually starting each component has become tedious and error-prone. A utility which could automatically start each component on the appropriate machine with the desired protocol will ease the confederation manager's job. This capability will most likely be added to the CMT.

An important confederation design consideration is the producer-consumer relationship between simulations. Each simulation expects to find certain classes of objects and object interactions in the confederation. It also expects to contribute certain objects and interactions to the confederation. Based on parameters supplied to the AIS by each simulation, a tool is under construction that will check for object and interaction requirements that cannot be met by the confederation. It will also check for the generation of objects and interactions for which there is no requirement.

Also under investigation for inclusion in the 1996 AIS is a system that would dynamically evaluate the relationship between confederation objects. This system, listening to all inter-simulation message traffic, would function like an actor in the confederation. The system would expose the evolving relationship between objects in terms easily understood by simulation participants, aid in the correctness verification of object interactions, and produce a condensed chronicle of the simulated situation for after-action analysis.

REFERENCES

Maginnis, F. X., et al. 1988. A Parallel Simulation Control Executive for Strategic Defense Battle Management Simulations. In *Proceedings of the Military Computing Conference '88 (Anaheim, CA May 1988).* Military Computing Institute, Los Altos California.

Seidel, D. 1993. Aggregate Level Simulation Protocol (ALSP) Program Status and History. MTR-93W0000079. The MITRE Corporation, McLean, Virginia.

Weatherly, R., A. Wilson, and S. Griffin. 1993. Event Distribution Protocol Quarterly Progress Report. MPR-93W000064V4. The MITRE Corporation, McLean, Virginia.

AUTHOR BIOGRAPHIES

ANNETTE L. WILSON is a Member of the Technical Staff for the MITRE Corporation. She received her B.S. in computer science *Magna Cum Laude* from Texas A&M in 1986. She participated in the design and development of the production ALSP and EDP software systems. She is currently leading development efforts for the 1995 releases of the ALSP software.

RICHARD M. WEATHERLY is a Principal Engineer for the MITRE Corporation in their C^3 Modeling and Simulation Center. He received his Ph.D. in Electrical Engineering from Clemson University in 1984. He is an original member of the ALSP team and the designer of the ALSP Infrastructure Software.