The Road to Successful Joint Experimentation Starts at the Data Collection Trail

Robert J. Graebener, Gregory Rafuse, Robert Miller & Ke-Thia Yao
M&S Team, Experimentation Engineering Department, J9 USJFCOM
Suffolk, Virginia
rgraeben@ida.org, grafuse@alionscience.com, rmiller@alionscience.com & kyao@isi.edu

ABSTRACT

Joint Forces Command has made great strides formulating the roadmap for conducting joint experiments. However, success for the Command will be measured by its ability to present quantifiable results to support transformational findings. The Services have considerable experience documenting requirements and articulating needs based on quantifiable results. Weapons systems, sensors and related procurement developments lend themselves to statistical testing (primarily through repetitive constructive simulation runs and live tests). The nature of joint experimentation relies on a discovery-type of approach when dealing with 2015 (or later) weapons, decision support systems and identifying the best methods for their utilization. The strengths in using human in the loop (HITL) immersion within distributed virtual simulations (e.g., Joint Semi-Automated Forces (JSAF)), requires innovative approaches to data collection and analysis. The correct approach will provide creditable and quantifiable results to strengthen the Commander, Joint Forces Command's rationale for transformation within DOD. This paper addresses methods for achieving more creditable and quantifiable data support. The first section provides a short description of the spiral development and data integration processes. The second section describes the flexible data collection toolkit used in the initial verification of entity behaviors and performance and then used to extract and display the data generated from the simulations. Finally, the third section describes a distributed framework for scaling the logger and analysis tools to handle very large data sets--in the terabyte range--for meeting the Joint Forces Command, Joint Experimentation Directorate's need for a Distributed Continuous Experimentation Environment capable of providing quantifiable results.

ABOUT THE AUTHORS

Bob Graebener retired after 25 years in the military in 1997. His last active duty assignment was as the Chief of Modeling and Simulation at USACOM's Joint Training, Analysis and Simulation Center. Mr. Graebener is currently a Research Staff Member and team lead with the Joint Semi-Automated Forces (JSAF) program at IDA. In that role, he has participated in several joint experiments sponsored by USJFCOM. He is currently working towards a doctoral degree in Systems Engineering from GWU.

Gregory Rafuse is a data collection analyst and developer and is currently the lead developer for the data collection toolkit. He is a Software Engineer with Alion Science and Technology. Mr. Rafuse has previously served seven years with the US Army as a Field Artillery Crewman. He also possesses an AAS in Computer Information Systems (CIS) from McLennan Community College and is pursuing a BS in CIS from Strayer University.

Robert Miller is a Senior Software Engineer with Alion Science and Technology. He brings over 11 years of experience to the current effort of designing, coding, and testing software for the Future After Action Review System. He holds a Bachelors Degree in Engineering from The Cooper Union School of Engineering and a Masters Degree in Computer Science from the City University of New York.

Ke-Thia Yao is a research scientist in the Distributed Scalable Systems Division of the University of Southern California Information Sciences Institute. Currently, he is working on the JESPP project, which has the goal of supporting very large-scale distributed military simulation involving millions of entities. Within the JESPP project he is developing a suite of monitoring/logging/analysis tools to help users better understand the computational and behavioral properties of large-scale simulations. He received his B.S. degree in EECS from UC Berkeley, and his M.S. and Ph.D. degrees in Computer Science from Rutgers University. For his Ph.D. thesis he implemented a spatial and physical reasoning system that automatically generated grids for novel geometries for computational fluid dynamics simulators.

The Road to Successful Joint Experimentation Starts at the Data Collection Trail

Robert J. Graebener, Gregory Rafuse, Robert Miller & Ke-Thia Yao
M&S Team, Experimentation Engineering Department, J9 USJFCOM
Suffolk, Virginia
rgraeben@ida.org, grafuse@alionscience.com, rmiller@alionscience.com & kyao@isi.edu

Joint Forces Command (USJFCOM) has made great strides over the past few years in formulating the roadmap and processes necessary to conduct joint experiments. The roadmap covers initial concept design through presentation of results to the Office of the Secretary of Defense. The Joint Requirement Oversight Council (JROC) is the responsible agent for determining which recommendations are taken for action, and more importantly, which are funded. The Commander, USJFCOM's objective is to "provide actionable recommendations from experimentation results to senior leaders to inform options for future force investments" (USJFCOM, 2003 p.3). USJFCOM's effectiveness in this environment will be measured by its continuing ability to present quantifiable results to support joint transformational findings.

Today the modeling and simulation (M&S) and operations research communities are faced with ever increasing challenges to meet the demand for creditable and quantifiable results. An initiative, currently underway at USJFCOM, is attempting to leap ahead of the demand by the creative integration of processes, commercial off the shelf (COTS) products and scalable parallel processors (SPP). This paper will address a method for providing more quantifiable and accurate data generated within large supercomputers running human in the loop (HITL) virtual simulations and federations of simulations used in support of future joint experimentation.[1,2]

BACKGROUND

Leveraging simulation to support joint experimentation has been the centerpiece strategy for USJFCOM because it provides a capability to rapidly prototype futuristic concepts. The M&S toolkit includes constructive as well as virtual simulations and, as the recently concluded Millennium Challenge 2002 joint experiment has demonstrated, live simulations will be integrated when needed (USJFCOM, 2002). Constructive simulations normally provide faster-than-real-time capabilities and are excellent when one requires statistical results to prove or disprove experimental hypotheses. Virtual simulations offer an environment that allows real people to be immersed within the futuristic environment, an excellent medium for evaluating decision-making processes.

Providing Immersive Synthetic Environments

Developing collaborative planning and decision support systems designed to establish user situational awareness present an additional challenge because of the reliance on human subjects as integral components of the command and control system. Immersion of humans within virtual simulations, such as Joint Semi-Automated Forces (JSAF), requires an integrated data generation and collection approach to achieve quantifiable results. Although HITL experiments offer a great potential for exploring complex issues they are a greater challenge to the data collection and analysis team and analysts charged with providing quantifiable results that will survive JROC scrutiny. Being able to provide usable results is also challenged when "setting the initial conditions" requires millions of entities to generate realistic levels of civilian traffic, before one even begins to assess future sensor or weapons systems operating in the urban environment. Federating simulations compounds the challenge, as interactions between simulations have to be checked for accuracy.[3] When sensor systems and radar systems are added to the mix, one can imagine the large quantity of generated data that must be logged, mined and provided to the analyst as quickly and accurately as possible. The enormity of this task was not lost on

[1] The authors caution the reader not to assume that there is only one solution to this challenge. Frequently a number of tools are required to bring distinct, quantifiable results to the senior decision maker.

[2] Federation. A named set of interacting federates, a common federation object model (FOM), and supporting Runtime Infrastructure (RTI), that are used as a whole to achieve some specific objective (Department of Defense, 1998).

[3] Federate. A member of a High Level Architecture (HLA) Federation. All applications participating in a Federation are called Federates (Department of Defense, 1998).

USJFCOM as decisions made in 2002 and executed in 2003 will be realized when the Joint Urban Operations HITL series of experiments in USJFCOM's Distributed Continuous Experimentation Environment gets underway in 2004.

Analysis vs. Discovery

Before describing the joint experimentation data collection strategy, a short description of the difference between analysis and discovery experimentation is warranted. Analytical data is largely derived from statistical testing, applying controls over independent and dependent variables so as to isolate the cause/effect relationships. Constructive modeling has been a major "toolset" in the analysis arena, primarily through its ability to run faster-than-real-time (therefore providing multiple runs to support statistical inquiries). Analysis techniques have been very effective when the problem can be framed with an expected outcome.

Discovery-type events, more often than not, rely on a progressive understanding of what is unfolding, thus requiring flexibility in tool design to explore new avenues as they present themselves. Although hypotheses exist, the number of dependent variables, human interactions, and complexities preclude adherence to rigorous statistical methods of analysis.

THE FUTURE AFTER ACTION REVIEW SYSTEM (FAARS)

The key ingredients to the FAARS are the process, tools and design of logger protocols that operate on scalable parallel processors. The remainder of the paper addresses these three areas. The first section provides a description of the spiral development and data selection and collection processes. The second section describes the flexible data collection toolkit used in the initial verification of entity behaviors and performance and then used to extract and display the data generated from the simulations. Finally, the third section describes a distributed framework for scaling the logger and analysis tools to handle very large data sets--in the terabyte range.

THE PROCESS

Once the concept designers and operations researchers have settled on the experimental concept, collaboration with the M&S community should closely follow. A process for successfully designing the simulation to support specific measures of performance (MOPs) is a recipe for success. Data collection development, when dealing with multiple simulations federated across a wide area network requires a commitment from all functional areas, not just the data collection team. Dependencies are created since the need for specific interactions related to the MOPs must be generated by the entities within the federation or simulation to answer the operations research questions. Figure 1 provides a pictorial of what is involved in the spiral development process (in this case for software).

.

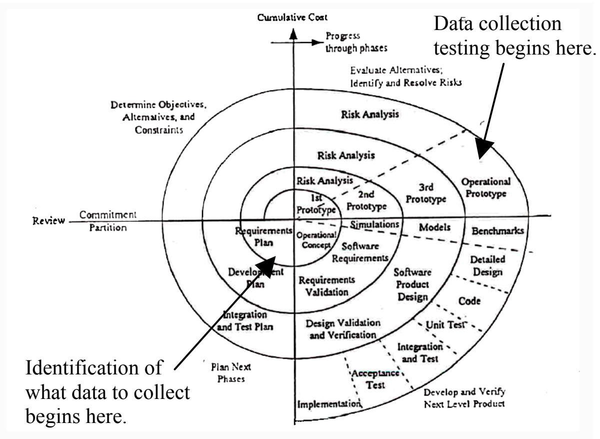


Figure 1. Spiral Development Process (Boehm, 1988)

The following checklist will bring value to the process and establish user confidence in the data and simulations generating the data. The steps described below were designed for a joint experiment supported by the HITL virtual simulation, JSAF (Graebener & Kasputis, 2000).

1. What question(s) does the experiment address?
   -What hypothesis is being explored?
   -Steps include "drilling down" or decomposing the question into sub-elements.

2. What Data will answer the question?
   -Sub-elements are further decomposed into metrics that will support the answer in a manner favorable to quantification.
   -This process begins a spiral crosswalk between the data analyst and the data collection developer to ensure what is being asked for is provided.

3. How is the Data Generated?
   -Further detail generated during the spiral crosswalk identifies how the data is created and by what means the data will be collected.
   -In most cases there will be several methods used to collect the necessary data, while automated data generated from the simulation might be the goal, observers monitoring the test participants could also generate (and collect) data.

4. How is the Data Transmitted/Received?

Table 1 shows the three tests used to explore the question above.[4] Each test has a set of conditions on the first line and then a solution or range of solutions.

Table 1. Data Transmission/Reception Matrix

Test #	Does Entity Transmit?	Does Entity Receive?
1	If answer is: Yes	& if answer is: Yes
	Then: Test to ensure enumerations are accurately sent/received.	
2	If answer is: No	& if answer is: No
	1: Then: Determine if data element can be derived indirectly from other interactions, or: 2: Create a new (experimental) interaction, or: 3: Determine if data element can be generated by non-simulation means?	
3	If answer is: Yes	& if answer is: No
	or	
	If answer is: No	& if answer is: Yes
	1: Then: check to see if the interaction field is empty, and; 2: Fill in the field with appropriate entry, and; 3: Ensure enumerations are accurately sent/received.	

5. When do you need the data and in what format do you need the data products (see Table 2)?
- -Identify the data display interval or timings (left-most column).
- -The periodicity is negotiable between the data analyst and the data collection developers. In some cases performance issues could impact on how often/how quickly one receives the generated data.
- -The format (top row) relates to the following headings:
  - (1) Raw data.
  - (2) Grouped.
  - (3) Combined.
  - (4) Disparate.
  - (5) Automatic Visual (Graphs).
  - (6) Manual Display.
  - (7) COTS Readable (e.g., SPSS).
  - (8) Final Form/Report Ready.
  - (9) Other.

[4] If a gateway is required (e.g., HLA to/from DIS (Distributed Interactive Simulation) further work must be done to ensure an accurate mapping across the gateway is established and tested.

Table 2. Data Format and Timing Checklist

Format	1	2	3	4	5	6	7	8	9
Timing									
Immediate/near-real-time									
Overnight									
End-of-Trial									
End-of-Exp't									
Other									

6. What types of data integration are envisioned for post-experiment processing?
- -By Time Segment.
- -By Mission/Task.
- -By Force Level.
- -By Geographic Region/Area.
- -Other.

7. What is the cost associated to achieve steps 4 thru 6?
- -In processing time?
- -In developer time and availability?
- -In bandwidth size?
- -In relation to other competing interactions?

THE TOOLS

This section describes a framework for using COTS and modified commercial software for logging simulation events and creating analysis tools. The rationale for moving to a new approach in toolkit design is discussed as well as a description of one of the tools currently in use.

New/Innovative Approaches to Data Collection

In previous joint experiment events, the data collection and analysis tools have been custom written applications that collected, processed and performed general analysis activities on a very specific and limited set of data. Previous solutions did not include a level of robustness and reusability necessary to meet future requirements. Because of this, a new approach was created to meet these anticipated demands.

The FAARS toolkit has been designed around a flexible, COTS-based solution minimizing the amount of specific software to be written. The philosophy behind using COTS software over proprietary custom software systems has been to allow concentration of most of the development efforts into providing highly flexible and responsive data extraction methods to support the analysis tools and data displays for the end-users. The FAARS toolkit provides near-real-time event information and a post-event data analysis. The near-real-time tools supports the verification and validation processes used for quality control of the simulation or federation generated data along with specific, predefined statistical reports and summaries. The post-event tools have been designed to perform analyses in support of the MOPs and measures of effectiveness (MOEs).

FAARS Toolkit Composition

The FAARS toolkit currently is comprised of the following commercial software applications:[5]

Data Collection:
1. hlaResults v2.0.2 (2002)
2. Microsoft Access2000 (1999)

Data Presentation:
1. Near-Real-Time
    a. Apache v1.3.27 (2003)
    b. PHP v4.3.2 (2003)
    c. ChartDirector v3.0 (2003)
2. Post-Event
    a. Microsoft Excel2000 (1999)
    b. Microsoft Visual C++ v6.0 (2002)
    c. MySQL v4.0.11 (2003)
    d. ChartDirector v3.0 (2003)

The core of the data collection is the hlaResults tool. This package provides for the logging of both HLA and DIS-based simulation data. The tool works by intercepting RTI-transmitted information amongst a given federation and storing the collected information in a Microsoft Access2000 database. Figure 2 indicates the flow of data within the FAARS toolkit environment.

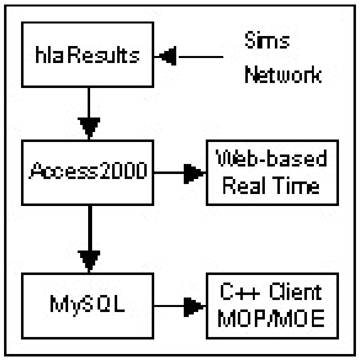


Figure 2. Data Flow and Capture

The near-real-time solution is based on the industry-proven Apache web server, the PHP (Preprocessor Hypertext Preprocessor) scripting language and the Chart Director graphing package. The web server/scripting language combination was primarily chosen to facilitate an independent, cross-browser, operating system solution for providing information in as-close-to-real-time as possible. The web server scripts connect to the hlaResults-generated database using industry standard ODBC (Open Database Connectivity) and SQL (Structured Query Language) commands. The returned information is then processed within the script and presented to an end user via a standard web browser.

The post-event solution is based on a custom-built C++ client interface to access/view data being stored in a relational database (MySQL) using industry standard ODBC and SQL commands. The C++ client interface then processes the returned data and presents the resulting information via a series of either Microsoft Excel2000 spreadsheets, natively or via ChartDirector graphics.

Data Relationships

Storing HLA-generated data in a relational database requires an appreciation for the differences in the way these two technologies handle information. A database, being an information storage and retrieval system, is geared towards eliminating data redundancy. An HLA federation is optimized for data exchange, and therefore is not subject to the limitations of a database. These two goals, though not completely in contradiction, need to be reconciled. Inasmuch as the data is being generated in an HLA-oriented format, it becomes necessary for the database to accommodate the federation (and not the other way around). This "accommodation" essentially amounts to a recognition of the relationships between data attributes in different objects. Such relationships, though not enforced by the HLA federation, must be respected. The FAARS tool development team needed to become highly conversant with the FOM, the Object Model Template (OMT), Federation Execution Document (FED) (a federation agreement between various simulations and the current RTI). From an understanding of how the various parts work together in forming the federation the team determined how the various fields were related and how to "force" relationships between collected data.

To force relationships within data where none are previously defined requires knowledge of the data and the data format. FOM-related data is transmitted as either an interaction between class objects or as a class object status update. Also, there are various linkages used to convey information about a simulation transaction that occurs within the simulation.

[5] COTS application providers are listed in Reference Section.

Understanding hlaResults Collected Data

The hlaResults data collection software package utilizes the OMT, FED and RTI components to generate a "collector." As part of the collector creation process, a pseudo-schema for an Access2000 database is created. This database contains tables representing each interaction or object class as defined by the three components. Each table contains fields representing the "payload" for a specific update of the interaction or class object. When the database is created, the hlaResults-generated schema does not introduce nor define any specific relationships between the various interactions and class objects. Therefore, relationships have to be imposed for the data to be truly relational. It is necessary to decide which fields within the interaction and class object table share common information. Once this is done, a database schema is dynamically generated within the FAARS toolkit and applied to both the Access2000 database (used as the immediate store) and the MySQL database (used as the final "rollup" data store).

Lessons Learned: Evolving Methods and Tools

Flexibility was a going-in design objective for this program. One of the initial decisions was to select Access2000 as the intermediate data store since it was already designed as HLA Results primary database. The team's ability to rapidly prototype the FAARS during early development was facilitated by this decision. However, as the spiral development process matured, new requirements surfaced for increasing the size of collect data storage capability for reasons explained below. The emerging result is a continued improvement to the capabilities of FAARS as it is adaptable to a variety of new COTS database products.

A technical limitation in Access2000 is a 2 GB limit for database sizes. Based on previous usage during test events, a typical HLA federation with 35,000 entities will cause an hlaResults-created Access2000 database to reach this threshold within 2 to 2.5 hrs of event runtime. A typical simulation day, designated by the experiment technical lead, consists of 6 to 8 hrs of continuous runtime, requiring three or more databases to be created. A technique for selecting and processing relevant data from each database (and taking into account overlaps within the data) has been developed by inserting the data into a MySQL database for a "roll-up" into one total event period, to preclude retrieving data separately from each Access2000 database. Currently, two hlaResults collectors are used during experiments in overlapping periods so that there is continuous coverage of simulation data being recorded. Figure 3 shows the method for employing collectors.

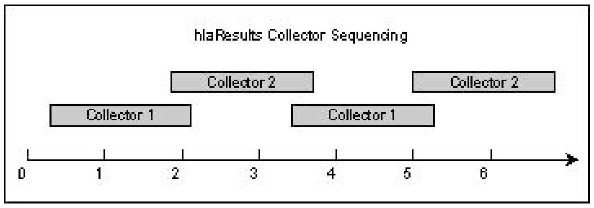


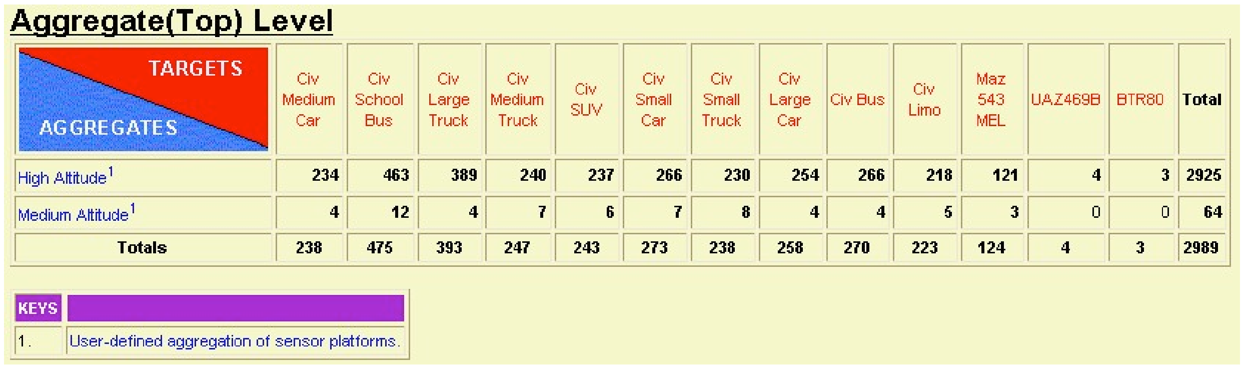
Figure 3. Collector Sequencing over Time

Processing Collected Data

After a simulation run is completed, the collected database segments are processed into the single MySQL database in the following sequence: First, a database schema is applied to the MySQL database that will represent the "rollup" of all of the data collected. Second, the Access2000 database segments are put into sequential order by their unique file names and internal file creation timestamps. Third, each segment is filtered to build a list of relevant tables from which data is to be extracted. Fourth, the first Access2000 database is inserted directly into the MySQL database. Fifth, subsequent Access2000 databases are processed to ignore overlapping data by examining the difference in timestamps between where the previous segment ends and the next segment begins. Also, internal record timestamps are adjusted with an offset so that individual record timestamps represent time since beginning of simulation run and not just for the individual database segment. Finally, summarization information is extracted and stored into new tables to facilitate speed in reviewing common information. Those reports that are processing intensive will be generated and saved for post-event review.

Post-Event Data Processing

Once the simulation data has been processed and inserted into the MySQL database, the MOP/MOE tools are applied to the completed database to provide predefined statistics for the event period. In conjunction with these predefined reports, additional reports and queries can be rapidly created based on additional feedback and desires of the analyst. A sample of predefined reports available for the end user includes: Killer/Victim scoreboard, Entity Lifecycle and Lifecycle Details, and "String" analysis charts. Other MOP/MOE components are developed and included with the toolkit based on input from the data collection and analysis plan designed by the joint experimentation users and analysts.



Aggregate(Top) Level

TARGETS / AGGREGATES	Civ Medium Car	Civ School Bus	Civ Large Truck	Civ Medium Truck	Civ SUV	Civ Small Car	Civ Small Truck	Civ Large Car	Civ Bus	Civ Limo	Maz 543 MEL	UAZ469B	BTR80	Total
High Altitude[1]	234	463	389	240	237	266	230	254	266	218	121	4	3	2925
Medium Altitude[1]	4	12	4	7	6	7	8	4	4	5	3	0	0	64
Totals	238	475	393	247	243	273	238	258	270	223	124	4	3	2989

KEYS	
1.	User-defined aggregation of sensor platforms.

Figure 4. Sensor/Target Scoreboard Example

A Near-Real-Time Tool: Sensor/Target Scoreboard

An example of one of the tools used for near-real-time analysis is the Sensor/Target Scoreboard. The scoreboard is designed to provide insight into the category of class object types being detected, the specific sensor platforms and sensor modes. One possible use of the Sensor/Target scoreboard is to provide summary reports of sensor activity for use during typical daily event briefings.

The scoreboard is divided into four layers of detail, with aggregated sensor platforms versus class object types summaries at the upper-most level. Figure 4 shows a sample Sensor/Target Scoreboard at the aggregate level. The next level of detail (accessed by clicking on a value in the table) displays the total count of individual class objects of a specific type detected by individual platforms and by specific sensor modes. Subsequently, the next level of detail (by clicking on a value in the displayed table) provides a list of specific detected class objects as indicated by the selected sensor platform and sensor mode combination from the previous table. The final level shows the finite information concerning a specific detected class object. The information contained in this report includes details of the sensing platform, the time of the detection(s), the class object detected, location of the detected class object, velocity of the class object (if moving), and detected appearance bit mask value.

SPP & LOGGING DATA

Harnessing SPP for Logging and Analysis

This section describes a distributed framework for scaling the logging and analysis tools to handle very large data sets--in the terabyte range--for meeting J9 needs for the Distributed Continuous Experimentation Environment. This tool is part of USC ISI's Joint Experimentation on Scalable Parallel Processors (JESPP) project. One of the goals of JESPP is to enable modeling of futuristic sensor capabilities, weapon systems and complex battlefield environments by providing the underlying computational and network infrastructure.

Logging and analysis desired properties

Listed below are some of the desired properties for the logger and analyzer framework. Several are inherently contradictory, so design decisions have to be made to balance them.

Scalability
The framework must be able to log and to process very large volumes of data. Based on past experiences from a Future Combat Systems (FCS) experiment, a partial logging of simulation events for selective data analysis requires about 5 kB/hr of data for each entity. Scaling up to the desired million entity SPP runs would require handling ~5 GB/hr. Logging all the simulation events for full playback or comprehensive data analysis would require over 200 kB/hr of data for each entity. For 1 million entities, the data rate is ~200 GB/hr. Five terabytes of data would be collected over the course of a 36 hr experiment.

Minimize resource contention
The simulation and the logger are running simultaneously, both requiring computing resources in terms of CPU, memory, disk access and network bandwidth. The logger should avoid competing against the simulations for resources.

Extensibility
The framework must facilitate the adding of new functional capabilities to handle novel types of analysis. As previously stated, goals and evaluation criteria should be defined well in advance of the event. However, it is not always the case and unanticipated questions do frequently arise. The framework must be

flexible enough to incorporate additional functionality without extensive modification.

Responsiveness
The framework must be able to respond to analysis queries of interest within a prescribed time frame.

Reusability
The framework must be able to work with different simulations and possibly with different HLA runtime infrastructure implementations. Initially, this work focuses on using JSAF and RTI-s (Calvin, et al., 1997).

Partitioning the data

One major design decision for a logger implementation is to determine where to store the logged data. Previous logger implementations for HLA simulations have utilized centralized data storage schemes (e.g., hlaResults) to store events on a centralized relational database. Typically, a centralized logger would self-register as an active listener interested in all federation events. The HLA runtime interface will then forward all events generated by each simulation federate to the logger. However, such centralized storage schemes are difficult to scale up as the number of simulation federates and number of entities increase.

Distributed storage schemes can overcome storage bottlenecks. Instead of funneling all the data to a single computer node, the data is partitioned and saved on multiple nodes. Then the question becomes how to partition the data. Possible partitioning schemes include partitioning by event type, by geography, by time, and by communication connection topology.

There are advantages and disadvantages for accessing data from each partitioning scheme depending on the type of analysis being conducted. For example, in order to compute a Killer/Victim scoreboard, the analyzer needs to access all the Damage Assessment Events. If the data were partitioned by event type, then all the Damage Assessment Events would be on the same node. The analyzer only needs to perform local queries to compute the Killer/Victim scoreboard. However, if analysis involves examining interactions among multiple types of entities, then the analyzer may need to join multiple remote data stores. In this latter scenario partition by geography may be a better choice.

If the desired analysis type is known, the logger should choose the data-partitioning scheme that minimizes remote data access. These partitioning schemes can be viewed as ways of pre-fetching data to support faster computation during analysis. Extra network bandwidth is utilized to aggregate data during collection, in order to improve responsiveness by minimizing communication during analysis. See Figure 5 for three representative types of data aggregation schemes.

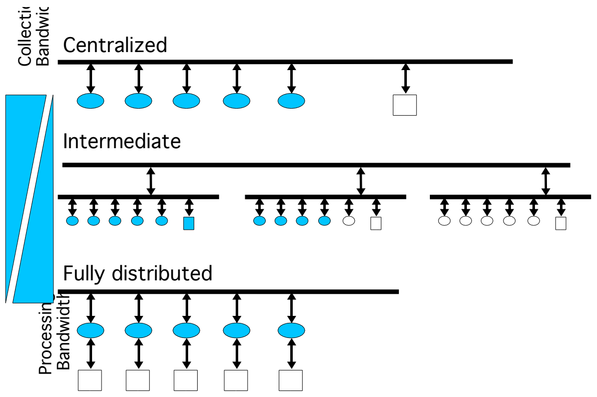


Figure 5. Data partitioning schemes based on communication connection topology.[6]

Frequently the analysis type is not known beforehand and arbitrarily choosing a partitioning scheme wastes valuable bandwidth during the actual simulation run. Also, sometimes multiple analysis types are desired and implementing multiple partitioning schemes wastes even more bandwidth. Resource contention between logger and simulation is unavoidable, when possible the logger should delay resource intensive operations until after the simulation runs. In addition overly aggressive data aggregation may result in underutilization of SPP computing resources. In the extreme centralized scheme only the CPU resource from one node is available for processing.

One way to minimize imposing additional communication load during collection is to partition the data according to the communication connection topology. For example, one type of connection topology used for the SPP is the three-level tree. The simulation federates sit on the leaves of the tree, while the RTI router nodes sit on the internal nodes of the tree. The simulation federates can only communicate with each other through the routers. The logger can be deployed on the routers to passively watch and

[6] Ovals are simulations and squares are the loggers. The centralized scheme expends network resources during collection, but it requires no network communication during processing. The fully distributed scheme is exactly the opposite, which requires no network resource during collection, but expends network resources during processing. Of the three schemes, the fully distributed scheme has the best potential for utilizing the distributed computing resources of the SPP.

collection events as they move through the router. The advantage of passive loggers is that they impose no additional load on the network, since they do not subscribe to interest channels (which forces additional events to be sent). However, the passive loggers attached to routers will fail to capture all the events on RTI implementations (e.g., RTI-s) that perform source-side squelching. If no simulation federate declares an interest to the events then source-side squelching RTI will drop the simulation events before they are sent to the router. To capture all possible events passive loggers attach themselves to the simulation federates themselves before the RTI has a chance to drop the events. Otherwise, the loggers have to become active and explicitly register interest to the squelched events.

Implementation

Based on discussions with potential users and simulation developers, the team came up with three implementation requirements:

- The logger must capture all simulation events. HITL events are very difficult to repeat and the users were concerned that passive data collection would leave gaps.
- The analyzer must be extensible. The user is sometime vague about final analysis products/results, so the analyzer must be flexible enough to handle a variety of analysis types.
- The logger/analyzer must minimize resource contention with the simulation. The simulation developers expressed a strong reluctance to give up simulation performance.

For the initial iteration the team plans to use a fully distributed data storage implementation. Also, to further minimize resource contention the plan is to insert very lightweight probes into the simulations to gather the needed events. Under an HLA simulation environment, federates communicate with each other using the RTI. Conceptually one can place a wrapper around the RTI to intercept all events sent through the RTI. Before forwarding these events to the RTI layer, the wrapper uses non-blocking interprocess communication (IPC) to send the contents of these to separate local logger processors. Since the local logger processor resides on the same node, no network bandwidth is consumed (see Figure 6).

The existence of the wrapper is hidden from the federates, so no modification to simulation federates is needed. Many current RTI implementations, such as RTI-s, RTI-NG and MÄK, provide Interceptors that facilitate wrapper implementation. In addition RTI-s provide dynamic Interceptor loading capabilities, so even recompilations are not needed.

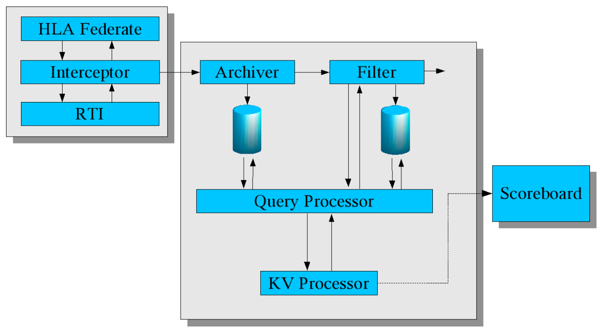


Figure 6. Node level view of logger

The local logger process provides a framework for multiple components to register and receive the intercepted events. For example, note the Archiver and Filter components in Figure 6. The Archiver component faithfully stores all intercepted events to disk. The Filter can be programmed to gather event subsets. Both the Archiver and Filter data stores are accessed through the Query Processor, which exposes an SQL-like API. The actual format of the data stores is exposed. However, for the initial implementation the plan is to store the data in text format. Based on past experiences with logging for network packet analysis, the team has found text format to provide a good first order implementation. The logged data is easy to perform a sanity check on through visual inspection, and easily manipulated with regular Unix tools. The query performance in most cases is acceptable.

For the local data stores the team has also considered using full-fledged relational databases, which provide the flexibility to handle general SQL queries. However, for the initial prototype the decision was not to use relational databases due to concerns that they may not be able to keep up with the volume of data during insertion. Also, logistically it is difficult to set up hundreds of databases (one for each node for each simulation run). In most cases nodes on cluster machines are reserved for fixed time periods. After the time periods expire the local disks are typically wiped clean.

Each Query Processor only provides a partial view of the simulation, since it can only access the data from the local data store. The SPP analyzer provides a framework for integrating multiple local stores to offer the appearance of a centralized data store to application analysis programs (e.g., Killer/Victim scoreboard, etc.). The scoreboard application sends a SQL query to the root Aggregator (See Figure 7). This query is replicated and forwarded to lower level Aggregators until it reaches the Query Processors at the leaf nodes. Then, the Aggregators merge the results of the query, and send the tables back up the tree. For the initial implementation, the plan is to implement query replication and simple result table merging/sorting operations within the Aggregators. In the future, there are plans to add metadata descriptions to local data stores to enable more advanced processing within the Aggregators.

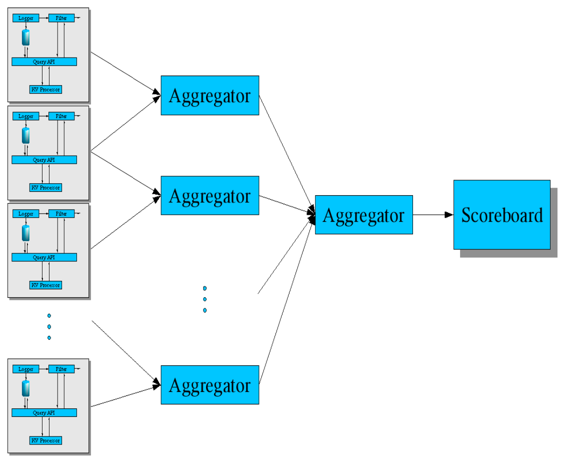


Figure 7. System level view of the logger across the nodes.

CONCLUSIONS

Being able to support senior decision makers with quantitative results does not have to be an insurmountable obstacle when using HITL simulations such as JSAF. Experience being gained at USJFCOM in harnessing scalable parallel processors to do data logging across the M&S federation will provide the technological base to better support the future needs of joint experimentation. Spiral processes and data collection tools round out the knowledge set that will enable quantitative data to be verified, validated and employed in the Department of Defense's quest to transform the military. This exciting area of simulation work is just a segment of the overall support found in joint experimentation, but a critical player, as the prudent application of public funds needs to be as effective as possible.

ACKNOWLEDGEMENTS

The authors wish to thank the JSAF M&S and SPP team members involved in this innovative undertaking. Data collection and analysis is a critical piece of the puzzle, but the people are the key to creative advancement.

REFERENCES

Apache HTTP Server. vers.1.3.27 [Computer Software]. Retrieved May 29, 2002 from http://www.apache.org.

Boehm, Barry W. (1988). A Spiral Model of Software Development & Enhancement. *Computer 21*(5), 61-72.

Calvin, J., Chiang, C., McGarry, S., Rak, S., Van Hook, D., & Salisbury, M. (1997). Design, Implementation, and Performance of the STOW RTI Prototype (RTI-s), *Simulation Interoperability Workshop*, 97S-SIW-019, Mar 1997.

ChartDirector. vers.3.0. [Computer Software]. Retrieved May 12, 2002 from http://www.advsofteng.com.

Department of Defense. (1998). *DOD Modeling and Simulation Glossary*, 1998 [Manual 5000.59-M]. Retrieved November 6, 2002 from https://www.dmso.mil/public/dod/policy.

Graebener, R.J., Kasputis, S. (2000). *The Evolving Role of Analysis in Complex Decision-Making*, Institute for Defense Analyses Document D-2415, (2000).

HlaResults. vers.2.0.2. [Computer Software]. Retrieved November, 18, 2002 from http://www.virtc.com.

MySQL. vers.4.0.11. [Computer Software]. Retrieved May 12, 2002 from http://www.mysql.com.

PHP. vers.4.3.2. [Computer Software]. Retrieved June 10, 2002 from http://www.php.net.

USJFCOM (2002). *Millennium Challenge 02 & the Joint Experimentation Federation*. October, 2003.

USJFCOM (2003). *Campaign Plan 2003-2009 Information Briefing*. March 13, 2003.