

Simulation Interoperability using the Synthetic Common Operating Environment in Simulation Based Acquisition

Karin Larsen, Ford Brockman, Chris Burns, Olga Nnedu, Cliff Sons
Quality Research, Inc.
4901 D Corporate Drive
Huntsville, AL 35805
(256)842-0026
e-mail: larsen@rdbewss.redstone.army.mil

Laurie Fraser
AMCOM
Lab Manager, Advanced Prototyping, Engineering and Experimentation Lab
Redstone Arsenal, AL 35898
(256)842-0942
e-mail: laurie@rdbewss.redstone.army.mil

Dr. Bill Hopkinson
Science Applications International Corporation
12479 Research Parkway
Orlando, FL 32826-3248
(407)306-4740
e-mail: William.C.Hopkinson@cpmx.saic.com

Keywords:

Computer Generated Forces, Simulation Interoperability, Reconfigurable Prototyping, Simulation Based Acquisition, Composable Behavior Technology

Abstract: This paper describes the Synthetic Common Operating Environment (SCOPE) software system that enables Simulation Based Acquisition (SBA) for future combat systems. The paper discusses how the SCOPE system allows the user to reconfigure currently existing entities, prototype new entities, and develop conceptual components and attach them to entities. Additionally, the user can build Computer Generated Forces, plan experiments, attach composable behaviors and analyze these new entities in large-scale battlefield simulations. The SCOPE system also provides the user the capability to collect experiment data for Run time and After-Action-Review on either a Distributed Interactive Simulation (DIS) network or a High Level Architecture (HLA) network. This paper will also discuss how easily the SCOPE software system can be configured to interoperate with other simulations, such as the Interactive Distributed Engineering Evaluation and Analysis System (IDEEAS) and One Semi-Automated Forces Test Bed (OTB) to take full advantage of SCOPE capabilities.

1.0 Introduction

The Synthetic Common Operating Environment (SCOPE) was originally developed as a framework to leverage and interoperate with evolving DOD and Army simulation architectures, models, and constructs. A few simple prototypes of software components were developed under the direction of the Mounted Maneuver Battlespace Lab (MMBL) under an initial project called Simulation Army After Next and Battle Command Future Environment for Reengineering (SABER). The current SCOPE effort is

a continuation of the SABER work under the direction of the MMBL. The enhanced SCOPE framework provides a robust suite of enhanced tools for developing composable entity behaviors, designing and testing experimental weapon systems, creating battle force units, and developing scenarios for both the OTB and IDEEAS simulations. Future plans include support for additional simulations.

The SCOPE tools can be used to plan a battle in a one-on-many environment, or folded into a large-

scale battle with other experimental systems and group organizations. This framework provides the capability to both inject experimental weapon systems or sensors into a scenario to perform “what if” experimentation in an integrated virtual battle space and the ability to provide real-time and post-processing support for After Action Reviews (AAR) and studies. SCOE consists of 4 primary components: Virtual Workbench (VW), Scenario Mission Planner (SMP), Composable Behavior Tool (CBT) and Data Collection and Analysis Tool (DCAT). Figure 1 depicts the SCOE components graphically illustrating how each component interacts.

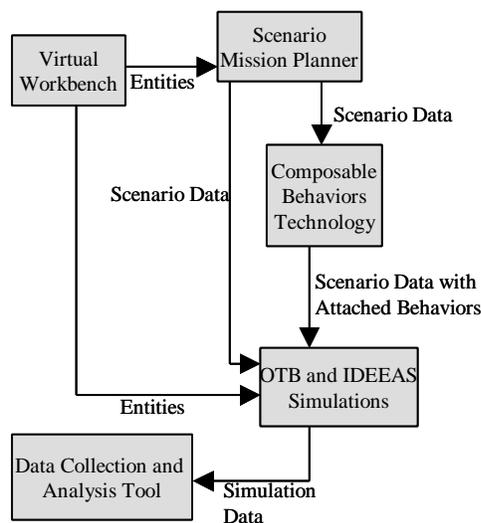


Figure 1 SCOE System Components

2.0 Background of Simulations

There were two simulations selected for the initial SCOE effort, OTB and IDEEAS. These models were chosen because of the interoperability of their terrain databases and because of the contractors experience level with these simulations. The strengths and weaknesses of each model will be described as well as a description of how each fits into the overall SCOE system.

2.1 OTB

The OTB system provides the capability to create and control entities within a simulated battlefield. The goal of OTB is to replicate the outward behavior of simulated units, their component vehicles and the

weapons systems to a level of realism sufficient for training and combat development. OTB supports an extensive list of entities including fixed and rotary wing aircrafts, ground vehicles, dismounted infantry, and specialized models such as howitzers, mortars, minefields, and environmental effects. Simulated entities can behave autonomously; that is, they can move, fire, sense, communicate, and react without operator intervention. These entities can interact with each other as well as with manned simulators, over a DIS network.

2.2 IDEEAS

IDEEAS is a simulation developed to enable rapid modifications to equipment, terrain, weather, and C3 functionality. The simulation performs as a stochastically guided, front-loaded system, absent of operator control inputs during exercise execution in either standalone or DIS-integrated operations. It is a fully accredited, high fidelity, engineering level, force-on-force simulation that combines weapon and sensor system characteristics, battlefield environment models, and attack/defend tactics to assess organization and weapon system effectiveness in a realistic battlefield environment. In interactive mode, terrain representation appears as 3D battlefield imagery, portraying roads, rivers, obstacles, and man-made features as well as tactical vehicle icons. Terrain elevation profiles along single selected axes can be displayed simultaneously in a superimposed window.

2.3 Interoperability of Simulations

SCOE is designed to allow a military analyst to take advantage of the strengths of each of these models as well as allowing them to participate in analyses to give a more realistic view of the battlefield. SCOE provides a GUI based interface to the underlying IDEEAS and OTB data, providing a data level bridge between the two models (See Figure 2). This data bridge provided for the user allows data to be edited from a centralized, single location, as opposed to being dispersed across many different files. The data is controlled using a database and input files are generated electronically as needed. This minimizes typographical errors and configuration management problems associated with maintaining multiple input file sets. Since the database is designed to be generic, SCOE can easily be adapted to support text data for most simulations.

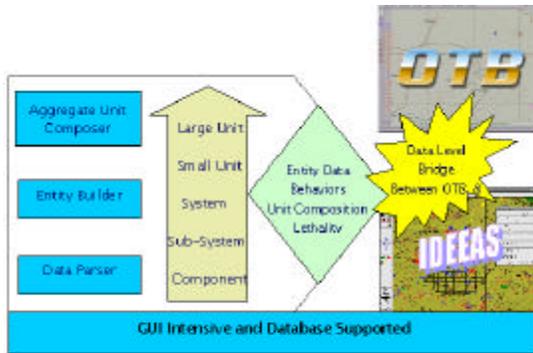


Figure 2 Interoperability of simulations through the Parser, Virtual Workbench, and Unit Builder

For SCOE, both IDEEAS and OTB are used concurrently to provide the best analysis for the minimum price. One of the strengths of the IDEEAS simulation is that it can handle large counts of entities on a single system. OTB can only manipulate a comparably smaller number of entities. However, it provides a very strong GUI for run-time support, allowing adjustment to scenarios during experiments.

SCOE is designed to take advantage of both these capabilities. Entities and units are created and ported to both simulations. OTB is used to create the specific detailed pieces of the scenario. The analyst develops one scenario for OTB with a small entity count. These entities are then given composable behaviors. Then, in order to provide a realistic battlefield, OTB is used to generate a larger scenario, which is translated to IDEEAS to round out the field. Then, both scenarios can be played at one time each affecting the other to determine the outcome of the battle.

All of the specific detailed pieces of the scenario can be run on independent OTB systems, providing run-time operator interface and greater detailed analysis. The remaining entities are transferred to the single IDEEAS platform. This minimizes the hardware and time required for the analysis process and provides a much more realistic picture of the entities and their interaction in the battlefield. All of these capabilities make SCOE the ideal set of tools to use for Simulation Based Acquisition.

3.0 Virtual Workbench

The Virtual Workbench is the central software component of the SCOE system and is hosted on a Windows NT platform. The VW applies object-oriented methodology to the composition of virtual systems, the drag and drop functionality and push button menus provide an intuitive, user-friendly

interface for creating entities, components and subcomponents. The military analyst can use the VW to select objects and create systems by defining subsystem components. The interface provides the capability to compose weapon systems from its component pieces. The analyst is able to choose “what” is to be used on the system, and also “how” it is to be used. Developing a new system piece by piece provides a flexible and powerful means of describing a conceptual battlefield. By using the VW, an experimental entity can be created in a fraction of the time it currently takes a military analyst to define a new entity for a Semi-Automated Forces (SAF) simulation. Using SCOE significantly speeds up the process of defining and testing experimental entities and components.

In the VW, the user is presented a two-window display, similar to the Windows Explorer application on a personal computer (see Figure 3). The left hand window displays a tree structure of entity classes, components, and subcomponents. The right hand window displays specific entities, components and subcomponents that have been loaded in the database and default data for a class type. When a specific component in the right hand window is double-clicked, a separate wizard allows the user to edit specific parameters on specific components.

All entities, components and subcomponents are stored in a Microsoft Data Engine (MSDE) database. The VW MSDE database allows the analyst to have configuration control over the individual pieces of the system being created.

One software component of the VW is the Parser Wizard. The Parser Wizard is used to preload the VW database with IDEEAS and OTB entities generated for past experiments. The Parser Wizard component guides the user through the process of selecting IDEEAS and OTB simulation files to import to the database. Once the entities and components have been parsed and stored in the database, they are available for the user to copy, edit or delete. When the user is satisfied with any modifications made, the Parser Wizard is used to export the entities and components generated in the VW to IDEEAS and/or OTB simulation files.

The VW enables the military analyst to quickly, easily and intuitively modify the parameters associated with existing entities, components and subcomponents. Additionally, entities can be copied and renamed. For the IDEEAS and OTB simulations, each entity and subcomponent type has a default set of associated data. For example, a default

tank might be a T72 tank. If the user creates a new tank, the T72 data is used to initially populate the new entry for this new tank. The user can then customize the tank definition to fit their current use. Therefore, if the analyst needs all new tanks to have a different tank configuration, the default record is edited to accomplish this task.

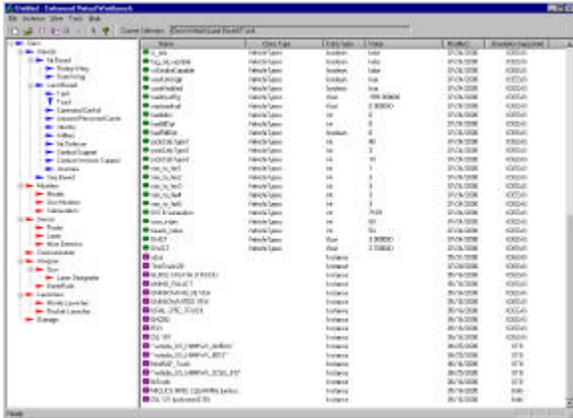


Figure 3 Virtual Workbench Display

When a new entity, component or subcomponent is created or modified, entries are made in the supporting database tables to correctly define this new entry for either the IDEEAS or OTB simulation. Performing automated entry of the components significantly reduces the amount of manual editing required, thereby reducing the number errors introduced in the file editing process. For OTB alone, the process for creating a new entity requires more than 11 flat files to be edited. Using the VW, this edit process is automatically preformed when the analyst creates a new entry.

3.1 Unit Builder

For force-on-force simulations, having a robust Unit Builder is an invaluable tool. The SCOE Unit Builder provides the analyst with another component that significantly enhances his ability to quickly group the experimental entities created by the VW into units to be used for simulation. The Unit Builder is hosted on a Windows NT platform and also has the same look and feel as the VW. A tree with the defined units types is listed in the left window with specific units that have been created shown in the right hand window.

The concept of units in the current SCOE system is only applicable to the OTB simulation. IDEEAS is not designed to model units. The Unit Builder

provides access to currently defined OTB units as well as the capability to modify the composition of these units and/or create new OTB units. The Unit Builder can use all entities, both pre-existing and newly created by the VW. The same Parser Wizard described earlier parses the existing unit definition from OTB.

When complete, the new unit definitions are stored to an MSDE database and available for exporting a new unit file to support integration with OTB. Since IDEEAS models organization at the entity level, it does not support units. However, the information in the database will be used for IDEEAS when the OTB scenario file is translated to the IDEEAS movement files.

4.0 Scenario Mission Planner

The SCOE Scenario Mission Planner (SMP) allows the user to develop force structures, plan a scenario, attach behaviors to entities created by VW and store all scenario information in an MSDE database allowing the user configuration management support over the entire process. Scenarios can be exported from the SMP to support both the IDEEAS and OTB simulations.

The SMP tool contains four major components: Plan View Display (PVD), Scenario Editor, Scenario Conversion Tool (SCT) and Composable Behavior Technology (CBT) Tool. When combined, these components provide the military analyst with a suite of tools for quickly defining and prototyping scenarios using experimental entities to support SBA. Additionally, the SMP tool components provide a reconfigurable environment to support virtual prototyping.

4.1 Plan View Display

Using the Plan View Display, the analyst can rapidly lay down forces for the mission on a 2-D map display. Development of a custom 2-D map component for creating routes, overlays and laying down forces was evaluated. Given the schedule and development constraints for the program, it was decided that an existing PVD would be used. Since the OTB PVD already provides the user most of the needed capabilities, it was integrated into the SCOE effort as the PVD component.

Using the VW, entities and units are created and exported to the reader flat files needed for input to the PVD. The current version of OTB (Build B) generates the scenario files in ASCII format. This allows the VW application to parse the files and make them available for storage in the VW database. This capability allows the user the flexibility for configuration management of scenarios as well as gives the user a way to visualize what units are contained in which scenarios. Using the OTB ASCII scenario file also allowed interoperability with the CBT tool. The input to the CBT tool is the ASCII generated scenario file.

To allow IDEEAS entities to also be planned with the PVD, a special wizard was created allowing the user to relate IDEEAS and OTB entities. The user would use the wizard provided in the VW to relate an IDEEAS and OTB entity. Then, the OTB entity would be available to the PVD for planning. The movement data for this new entity is stored in the OTB ASCII scenario file. Therefore, a translation process for the movement data is provided by the SMP allowing this movement data to be translated to IDEEAS format.

4.2 Scenario Editor

A Scenario Editor component was created in SCOE for two main purposes. The first purpose was to allow the user to visualize scenarios created with the PVD. The second purpose was to allow the user to edit other scenario related files. IDEEAS contains multiple scenario files that can be edited by the user to define scenarios.

In order to maintain consistency between the tools comprising the SCOE system, the Scenario Editor was designed to have the same look and feel as the VW. It presents the user with a split-screen display containing a tree with the simulations supported on the left side and scenario files associated with a simulation on the right side.

The Scenario Editor functionality differs based on whether the scenario files are IDEEAS-generated or OTB-generated. The user has the ability to edit and change the IDEEAS scenario file contents. Because each scenario file has a unique format, customized screens were created to allow the user to edit and change the scenario data. The movement data generated for IDEEAS from the OTB PVD is available for the user to review; however, any editing must be performed using the OTB PVD. Generated

movement data is marked on the screen as “generated” data and specifies the name of the scenario file that requires editing in order to make changes.

OTB scenario files are available for the user to visualize only. If changes to an OTB scenario are required, the user must use the OTB PVD to make them.

4.3 Composable Behavior Technology

There is a growing need for advances in techniques and methods of semi-automated representation of friendly and opposing forces within Advanced Distributed Simulation (ADS). Force modernization has caused an evolution in the tactics and doctrine of friendly and opposing forces. In addition, the focus has shifted from a large monolithic force to forces supporting more regional threats. These events have created a situation where existing battlefield behaviors must be modified and new battlefield behaviors must be created in a timely manner. This is particularly challenging as the specifications for behaviors are costly and time-consuming to develop, and once specified, implementation of those behaviors is labor intensive. A cost-effective methodology for evolving and developing Semi-Automated Forces (SAF) behaviors must be developed. A capability is needed where realistic tactical behaviors can be easily *composed* from a set of behavioral primitives. In addition, this capability must allow the end-user to define new behaviors to meet his simulation objectives.

The development of behaviors for simulated entities in SAF systems is a time-intensive process that excludes interaction of the end user. There is a great need for applications that will allow the end user to compose new behaviors. The Composable Behavioral Technologies (CBT) system allows for creation of custom behaviors for simulated entities for a discrete set of systems. The emphasis of CBT is to empower the simulation end-users to flexibly create behaviors meeting their specific simulation or scenario requirements, and to populate a behavior repository for potential reuse.

Company-level Rotary Wing Aircraft (RWA), tracked and wheeled vehicle behaviors have been identified as the behavioral domain for the current SCOE effort. Though OTB is being used for the purposes of the prototype development and

demonstration, to the greatest extent possible, CBT is a generalized solution that can be applied to other entity-based simulations.

4.3.1 CBT Design Philosophy

The original intent for CBT was to extend ModSAF to allow non-developers to compose behaviors. However, that idea limited the technology to one SAF simulation system and did not promote the approach to SAF simulations in general. To create a methodology that is applicable to multiple SAF systems, the CBT concept was extended to a composable system that “plugs” into other SAF simulations. The resultant system architecture is presented in Figure 4. Because the CBT system is external to the SAF simulation software (e.g. ModSAF or OTB), it provides an environment that is intended to be more flexible for SAF behavioral experimentation.

Since CBT is a composable software system, its development necessitated a language that was easily ported to other hardware platforms. The Java language was selected because of its flexibility as a “compile-once run-anywhere” language. Also, Java is an object-oriented language that allows for a more methodical development environment.

4.3.2 System Architecture

The application components for the CBT architecture include the Behavior Repository, Behavior Editor, Execution Engine, Execution Tool and Repository Manager. These application components are intended to be independent of the SAF simulation in order to provide a high level of reusability across SAF simulation domains. In this architecture, primitive behaviors are considered to be closely linked to the SAF simulation system. Thus, the architecture includes a SAF dependent component, the SAF Specific Services (SSS). The SSS provides both simulation and GUI access, which is supported through two rigorously defined APIs: the PVD API and the Behavior API. The PVD API provides a seamless integration between a specific SAF GUI and the CBT GUI. The Behavior API allows a CBT composed behavior to succinctly and directly access specific SAF primitives.

Users of the CBT system include the SAF Developer, the Configuration Manager and the CBT User. The CBT architecture effectively removes the

software engineer (i.e. SAF developer) from the behavior development cycle except for the development of behavior primitives.

The SAF Developer implements behavior primitives within the SAF simulation system and stores the references in the Behavior Repository. Once the behavior primitives are defined, the CBT User employs the Behavior Editor to compose new composite behaviors from those primitives. The composite behaviors developed by the CBT User are stored in the Behavior Repository and can be used immediately. Because there is a need for control in any type of repository, a Repository Manager is included in the CBT architecture. The Repository Manager is used by the CBT Configuration Manager and provides configuration management functions for the Behavior Repository.

The Execution Tool allows the CBT User to create units and assign behaviors to the units. It also provides the CBT User the ability to create missions for a SAF simulation unit. A mission is a scenario-specific sequence of behaviors. The behaviors are those, which were developed with the Behavior Editor and stored in the Behavior Repository. When the CBT User completes mission specification, the mission data is passed to the Execution Engine. The Execution Engine schedules the mission behavior and initiates communication with the SSS. This communication allows the Execution Engine to direct execution of primitive behaviors within the SAF simulation system. It also relays information about behaviors’ status to the Execution Engine. The status information is processed by the Execution Engine to update the scheduled mission behaviors.

4.3.3 Graphical User Interfaces

An important advantage of CBT is its use of graphical user interfaces to allow the end-user to effectively create behaviors for SAF entities. CBT contains two GUIs: the Behavior Editor and the Execution Tool.

The Behavior Editor builds and displays the BRG in graphical form and enables the user to construct hierarchical, echelon-based behaviors. The Behavior Editor provides the user a simplified logic-diagram format for sequencing primitive behaviors to form composite behaviors. The intent of this format is to provide the user a more understandable means for representing behaviors than other existing methods of behavior representation such as state diagrams.

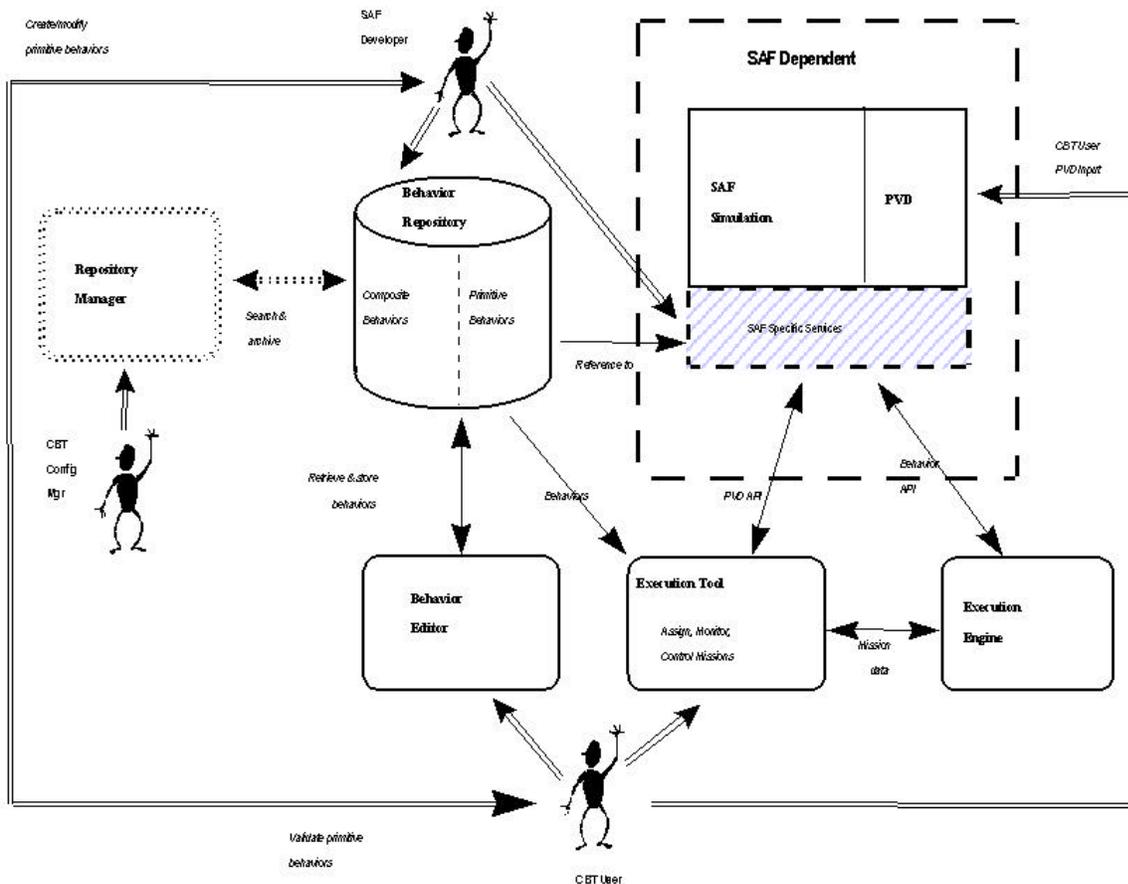


Figure 4 CBT Architecture

The end-user will not have access to the textual representation, but only a visual illustration of the behavior. The visual display will not be programmatic either, but will relate more to how a SME describes a behavior.

The Execution Tool contains a unit editor and a mission editor. The SME will use both jointly to define a mission. The unit editor enables the user to dynamically and explicitly organize a unit. The mission editor permits the user to create and view hierarchical missions. By having both the mission and unit editors work cooperatively, the Execution Tool gives the user the versatility to create a tactically realistic SAF mission. The unit's task organization can change over the course of a mission. The Execution Tool also allows for explicit representation of concurrent behaviors that have varying start and stop times across the unit hierarchy.

4.3.4 Integration with Virtual Workbench

The combination of CBT with the Virtual Workbench provides the simulation user with a powerful, robust tool for developing unique entities and units for training and experimentation. A user will now be able to create a new, prototype physical entity, rapidly develop a behavior for the entity and execute the entity in a simulation seamlessly. This capability will provide trainers and analysts with the robust capability to support the requirements for Simulation Based Acquisition as well as the emerging Future Combat System.

5.0 Data Collection and Analysis Tool

Exercise analysis may concentrate on the behavior of a particular simulation component, or the aggregate behavior of a collection of components.

This analysis can be used to assist in the simulation model verification and validation process. But, data collected during the simulation execution can also be used to evaluate performance of individual simulations and/or participants in an After Action Review. A key benefit of data collection is the ability to review or replay a portion of the exercise at various levels of detail using state of the art graphics projection and visualization capabilities. DCAT was developed to provide the user with the above capabilities.

Originally, DCAT was developed to run on a UNIX platform and operate in exercises transferring data over a DIS network. All data collected by DCAT-UNIX was stored in an object-oriented database system. During the SCOE effort, the tool was converted to run on a Windows NT platform and an HLA interface was added to provide added flexibility. Additionally, the database system was converted to an MSDE database. This database conversion provides added interoperability with the other SCOE tool set components, which also use an MSDE database. The MSDE database is royalty free, allowing the Army flexibility to distribute and use DCAT in various exercises with a minimum cost to the end user.

5.1 Real-Time Monitoring

DCAT is a real-time data capture and analysis application that collects data from a DIS or HLA exercise and provides feedback to the user concerning system performance. The tool is used as an AAR support tool, an experiment debugging and monitoring tool, a real-time experiment analysis tool, and a post-process analytic tool.

The amount and types of data collected from the network during an exercise depends on whether DCAT is running in DIS or HLA mode. If DCAT is operating in DIS mode, the PDU data selected in the "Configure" step is captured from the network and stored in the MSDE database. If DCAT is running in HLA mode, all HLA data transferred across the network during an exercise is collected and stored in the database. This allows the analyst to monitor the exercise in real-time and make adjustments based on the data. In real-time mode, 2D and 3D charts are automatically updated during the experiment run. See Figure 5. In this mode, the application executes the Measures of Effectiveness (MOE) queries against the current database state and automatically updates graphs based on the results of the queries.

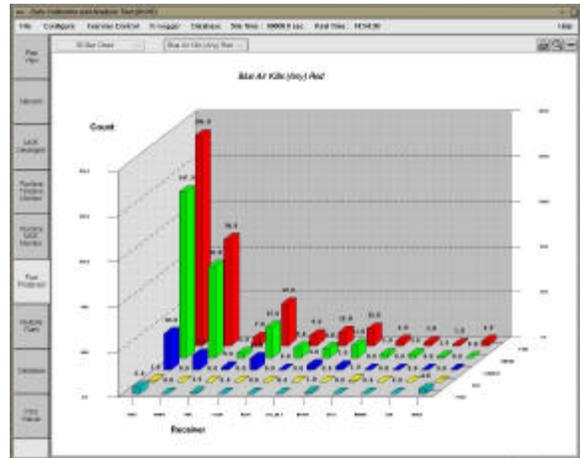


Figure 5 Sample DCAT Chart

5.2 Calculating Measures of Effectiveness

DCAT offers the ability to view Measures of Effectiveness in real time or post processing mode. DCAT generates log files suitable for import into a wide variety of Commercial-Off-The-Shelf (COTS) analytical tools for post-experiment analysis and manipulation.

The MOE developer (Figure 6) is based on the concept of a tuple, which consists of an actor, an action, and a receiver. The tuple defines a query that returns a set of records to the application for the purpose of generating graphs. Filters such as location, entity identity, bumper number and time may be applied to the tuple to further refine the set of objects returned in the query. These filters are created using the filter definition portion of the MOE developer. Once the tuples and filters are defined, the user can combine them using standard algebraic set notation into meaningful expressions. It is these expressions that define the MOEs.

5.3 Chart Generation and Database

DCAT offers charting features that are fast, efficient and easy to implement. The analyst can select from a wide range of tailored graphs and charts including 2D Bar Chart, 3D Bar Chart, Time Series Plots, Line Graphs, and Box Plots. All graphs and plots are automatically scaled and labeled.

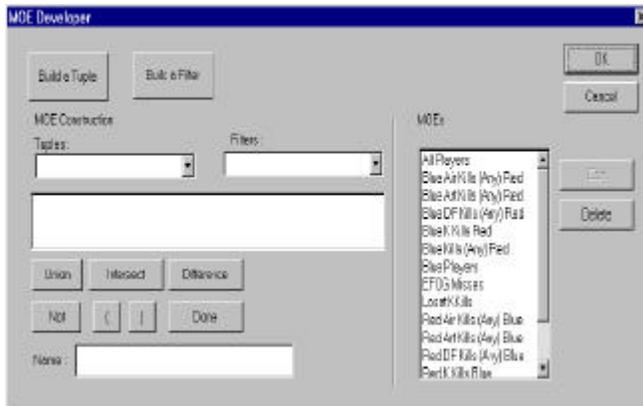


Figure 6 MOE Developer

DCAT utilizes the MSDE database engine for all data storage and retrieval. MSDE is royalty free database engine that operates in both local and multi-user environments. DCAT provides the user with the capability to browse the database records and filter them during and after an experiment. Using the database browser (Figure 7), detailed information concerning specific systems and/or events can be viewed and printed as desired. The database browser can also be used to build an entity filter for use in the MOE development process by reducing the records captured in the database view and creating a filter based on their identities.

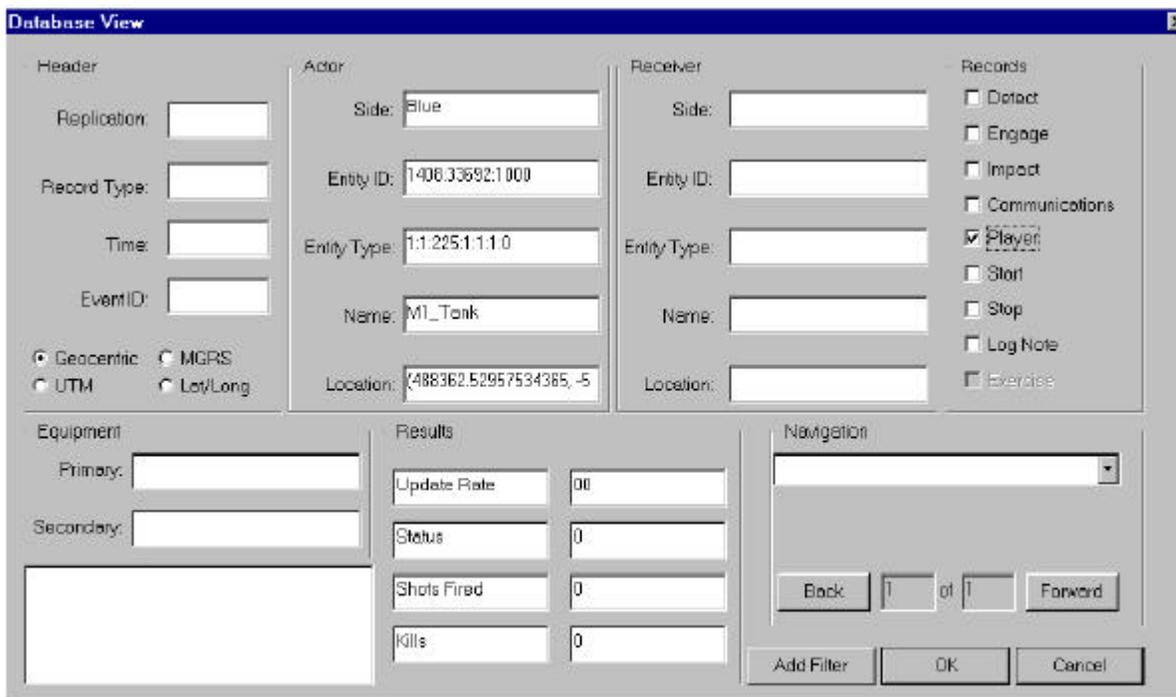


Figure 7 Database View

5.4 DIS/HLA

DCAT supports both DIS and HLA. As part of the configure capability, the user can select the type of network to use. If the DIS option is selected, the user can specify the PDU types are to be processed by DCAT. If the HLA option is selected, DCAT subscribes to all object classes (data elements) and interactions (events) for the Federation Object Model (FOM) used in the current experiment.

6.0 Conclusions

The use of SCOE will greatly enhance the users capabilities for SBA. SCOE provides greater flexibility for less technical operators to adapt to the simulations. It dramatically reduces the amount of time needed to introduce new virtual prototypes to the simulations, and reduces the cost of performing experiments, both in labor with the reduction of operators, and in the hardware needed to perform the experiments. In addition, SCOE provides data collection capabilities that will speed experiment debug time and enhance experiment effectiveness by providing run-time data collection as well as detailed after action review.

SCOE currently interoperates with IDEEAS and OTB, but it could be easily adapted to support other simulations such as SIMNET and CCTT. Since the

U.S. Army has a significant investment in legacy virtual simulations (SIMNET) and new virtual simulation (CCTT). In order to maximize the Army's investment in these simulations, the necessary SW and HW tools which allow virtual simulations to conduct simulation exercises on a common synthetic natural environment is critical to the Army of the future. These simulations need to be able to fully interoperate in a DIS and/or HLA common environment with an acceptable level of "fair fight". The work performed on the SCOE project is a significant step towards achieving and surpassing this goal.

the U.S. Army and NASA. She was the Program Manager on the SCOE project for Quality Research. Ms. Larsen received her Bachelor of Science in Computer Science from University of Missouri in 1983.

7.0 Acknowledgements

The work described in this paper was performed as a joint project between SAIC in Orlando, FL and Quality Research, Inc in Huntsville, AL. As the prime contractor, SAIC provided all the expertise on the CBT tool. Quality Research was responsible for development of other SCOE software tools. The work was performed for the Mounted Maneuver Battlespace Lab and funded by STRICOM under the ACTII contract number N61339-00-C-0013. Many thanks go to Major Joe Burns of the MMBL for having the vision to fund this work under both the SCOE and the SABER development efforts. Also, a special thanks to the entire SAIC and Quality Research team.

Principle Author Biographies

Dr. William Hopkinson is a Senior Systems Engineer for SAIC and has over 15 years academic and military experience in developing simulation-based training systems. Dr. Hopkinson has been a project director and lead engineer for several large simulation training experiments including the Virtual Integration Experiment, the Digital Training Exercise, and the Apache Longbow Combat Identification Study. Dr. Hopkinson received his Ph.D. in Industrial Engineering in 1995 from the University of Central Florida. His dissertation focused on validation of man-in-the-loop simulation for distributed exercises. Dr. Hopkinson is a retired US Army officer.

Karin R. Larsen is the Software Engineering Operations Manager in the Missile and Simulation Group at Quality Research. Ms. Larsen has 17 years experience in developing software applications for