Automating MSDL Scenario Generation and Entity Integration For Simulation Events

Timothy J. Collins
Science Applications International Corporation
Fredericksburg, Virginia
timothy.j.collins@saic.com

ABSTRACT

The Military Scenario Definition Language (MSDL) was developed, in part, to provide a standard language for use throughout the military simulation community with the intent of permitting quick and easy integration and sharing of military scenarios on various simulators. Unfortunately, its promise of reducing technical integration workload across heterogeneous simulations has not been achieved. In response to the development workload demand of the Army Joint Forced Entry Warfighting Experiment (JFEWE), the Battle Lab Collaborative Simulation Environment (BLCSE) Engineering Support Team applied MSDL functionality by automating several of the labor intensive tasks in scenario generation and entity integration. This paper addresses the automated methodology and tools we developed for accurately and efficiently generating MSDL scenarios which can be used in simulation experimentation. This paper also addresses the additional automation of checking the submitted force structures for errors and for unmapped or non-existent units and entities, and then automatically mapping those units and/or entities to the appropriate files so that heterogeneous simulations are able to effectively communicate. The use of automation to generate the scenarios and integrate entities not only saves an incredible amount of time but drastically reduces human error. The automation processes presented in this paper were successfully applied to the JFEWE technical integration, thereby significantly reducing manpower and the time required to build, test, quality assure, and modify scenario files and integrate the federation. The application of these non-proprietary processes and freely available Government off the Shelf (GOTS) tools can significantly reduce the manpower requirements and schedule time required to integrate heterogeneous simulators into the warfighting experiments that Prepare our Forces to Secure our Future.

ABOUT THE AUTHOR

Timothy J. Collins has been involved in Army modeling and simulation as a participant, engineer, and experiment planner since 1994. His occupational career specialty in the Army as a Cavalry Scout provided the knowledge foundation for military tactical maneuver pertaining to scenario development as well as map and terrain navigation pertaining to the implementation of global map informational data and its use in simulation terrain database development. Mr. Collins has a Bachelor's Degree in Computer Science and a Master's Degree in Information Management Project Management from Grantham University and has earned the Modeling and Simulation Professional Certification. Mr. Collins is currently employed by SAIC on contract as a simulation scenario and terrain database development engineer and geospatial subject matter expert for the Army Capability and Integration Center of the Joint Models and Simulations Division based in Fort Monroe, Virginia.

Automating MSDL Scenario Generation and Entity Integration For Simulation Events

Timothy J. Collins
Science Applications International Corporation
Fredericksburg, Virginia
timothy.j.collins@saic.com

PROBLEM STATEMENT AND CONTEXT

Conducting integration tasks for a major Defense simulation event can be a very challenging and time-consuming effort. Several tasks must be accomplished in order to get all the participating heterogeneous simulations communicating using the same language. Every entity from every participating simulator must be known and accurately represented in all the other simulators.

Simulation Experiment Integration Efforts

For the Battle Lab Collaborative Simulation Environment (BLCSE), integration entails establishing communications between simulation systems over a modified version of the MATREX High Level Architecture (HLA) Federation Object Model (FOM). The FOM, as it is called, lists all data objects that will be passed from each of the BLCSE Federation's participating simulators. Most importantly for this task, it lists all the life forms, platforms, and munitions that will be represented.

As the experiment planners determine the order of battle, they determine what units and entities will be conducting certain operations and where. Each of these units and entities will be represented by a specific simulation that is specially designed to accurately model certain types of entities. Unfortunately, there is not a set naming convention from simulator to simulator, or in some cases, even within a single simulator. And, that's not likely to change any time soon due to the collage of both Government and civilian developers working on each system.

Be that as it may, those entities must have representation within all the other simulators participating in the experiment. To accomplish this, each participating system would submit a force structure that depicts all the units and entities planned in the Order of Battle.

These force structures would typically be submitted from each federate to the BLCSE integration team. Once received, this information would be used for two purposes. One purpose is to compare the force structure to the FOM to ensure all entities are represented. The other purpose is to generate a constructive scenario representation.

Normally, these force structures would be submitted in the form of an Excel spreadsheet listing the units and entities in a typical force structure pattern. That is, entities would be entered immediately below the parent organization to which it belongs. Likewise, subordinate units would be below their parent organizations. These spreadsheets would often be delivered at varying degrees of completeness. The same entities, while having common military nomenclatures, would often have only a common name, or something only understood by the force structure developer. For example, the requested entity would be listed as a HMMWV in the submitted force structure. Considering that there are sixty-plus variants of the HMMWV, this can be very confusing for the engineers creating scenarios or the engineer attempting to map entities into the FOM. Another issue was that the force structures weren't always in an Excel spreadsheet format. They were sometimes delivered in any number of document or spreadsheet formats with which the force structure developer felt comfortable.

The bottom line is that the engineers would spend an inordinate number of hours attempting to decipher the force structure and coordinate with the planners to determine exactly what was requested.

The Entity Mapping Problem

The task of mapping these objects has historically been a monumental task. Excessive labor hours were spent parsing through the submitted force structure to find one instance of every entity that was listed. Once each entity in the list was found, the comparisons would begin.

The first comparison would be against a master entity list. The master entity list is in spreadsheet format, and lists every known entity along with both its distributed

and HLA enumerations and naming conventions. These are entities that already have a mapping in the BLCSE Federation. Should an entity be listed in the force structure and not in the master entity list, a new entry would be input. Again, the non-standardized naming conventions of the various simulators, and the risk of human error from manually entering the new entities, resulted in numerous duplicate entries. Additionally, a lack of configuration management for the master list resulted in ad-hoc updates and a non-published list unavailable to most of the experiment participants.

Once the engineer has documented all the newly requested entities into a master entity list, several additional documents must be updated to ensure that each of the new entities is represented. The FOM, the HLA Adaptor, and several OneSAF files also need to be updated with the new entity information.

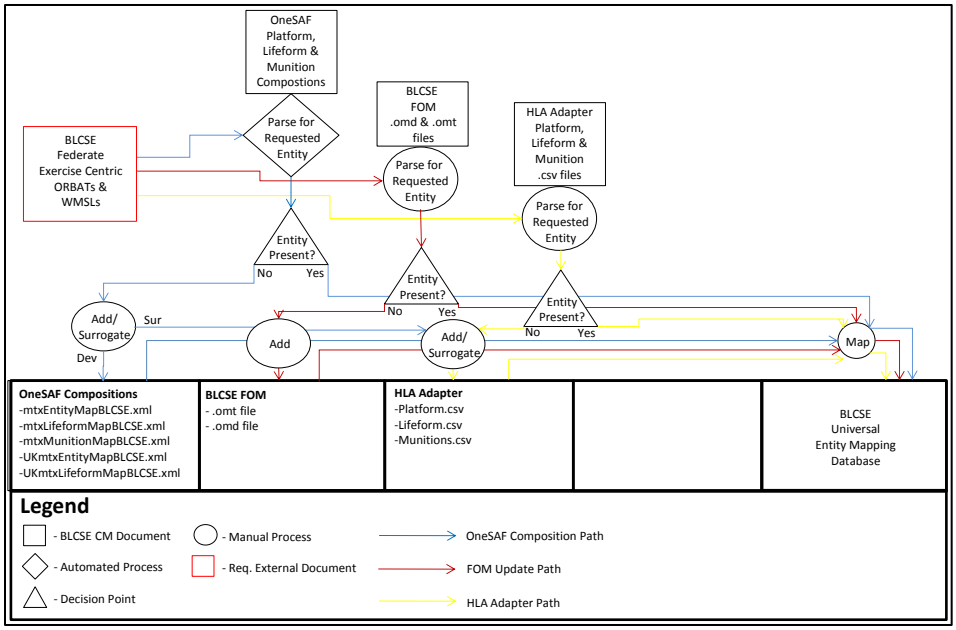

Figure 1 - Historic Entity Mapping Flowchart

As can be seen in Figure 1, the task of adding an entity into all the required locations is extremely complicated and time consuming. For a typical force structure containing several undocumented entities, the time required for updating all the files could take up to a month for the engineer to produce all the usable files. And, those files typically contained errors that would delay the experiment integration schedule.

The figure shows that only one part of the process was automated. And, unfortunately, the engineer that was responsible for this task was never able to provide a working copy of that program.

The Scenario Problem

As with the entity mapping problem, the scenario problem had to deal with unformatted force structures and haphazardly named entities. The engineer responsible for creating the scenario would typically spend the first couple of weeks deciphering the force structure. Upon figuring out the naming convention, the engineer would have to determine whether those vehicles were even available in OneSAF to put into a scenario. Additionally, even if the entities were available, the unit composition may not be available.

The creation of the scenario would entail the engineer hand emplacing each entity onto the OneSAF displayed map and manually editing each entity to apply the BLCSE naming convention. For a Brigade sized scenario this would be done for several thousand units and entities. The amount of time required for generating a Brigade level scenario was approximately two to four weeks. However, because of the intense manual labor and inevitable human error involved with renaming all the entities, there would invariably be entities with duplicate names. This would cause a multitude of problems during the exercise integration, record runs, and in the data collection and analysis.

Because the scenario would be created using OneSAF, the finished scenario would be saved as a OneSAF formatted scenario. OneSAF also has the capability to export the scenario as a Military Scenario Definition Language (MSDL) scenario. And unfortunately, as of this writing, scenarios in OneSAF, saved to the MSDL format, are unable to be reloaded, rendering them useless.

This accentuates one of the big problems with the MSDL format. While it is meant to be the Military Scenario Definition Language, very few simulators are able to ingest the format. And, even fewer systems are able to create a MSDL formatted scenario. Because there are so few systems capable of generating a scenario in the MSDL format, an extensible markup

language (XML) programmer would typically be required to develop the scenario in XML code. This would be a very time-consuming and inefficient way to create a scenario.

The bottom line is that the entity mapping task and scenario generation required intense, error prone manual processes that usually took several months to complete. These processes are not optional and are required for every experiment conducted by the BLSCE experiment community.

THE SOLUTION

The solution to the entity data mapping integration issue was to develop and implement several new processes for the way this aspect of experiment integration was to be conducted. After deciphering the undocumented, manual methods used for adding and mapping the entities into heterogeneous and homogenous simulation exercises, a three-phased approach was adopted and named the Entity Data Mapping (EDM) task (Figure 2).

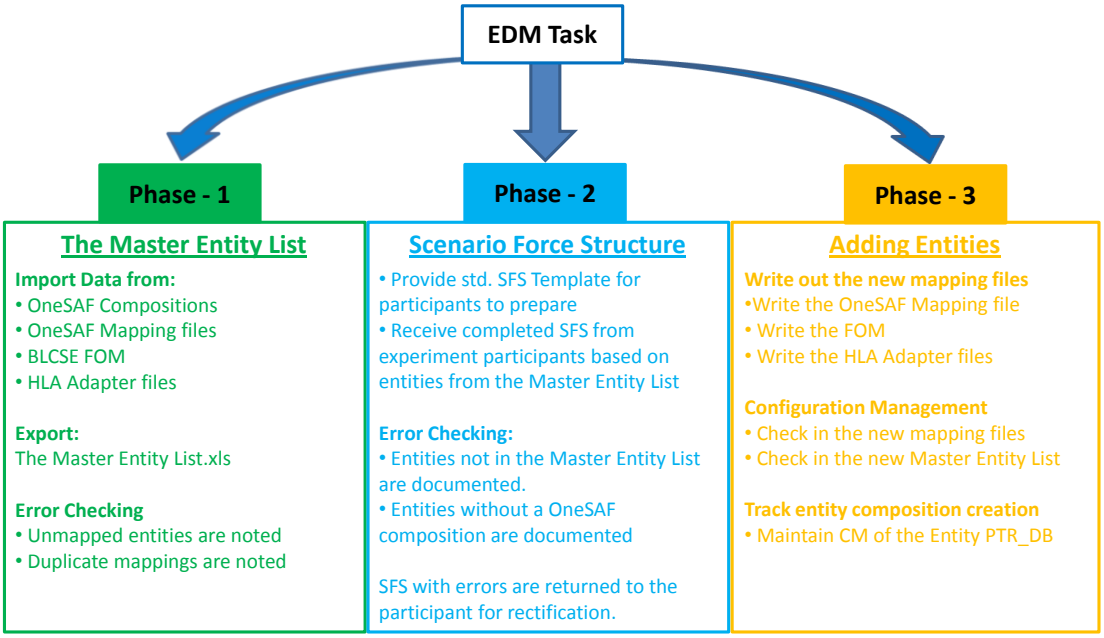

Figure 2 – The 3 Phase EDM Process

First, Phase 1 focused on the configuration management of the simulator mapping files and the BLCSE Master Entity list. The BLCSE Master Entity List (MEL) is a document that lists all the entities currently available in OneSAF that are already mapped throughout the federation.

Next, Phase 2 focused on the standardization of a format in which unit and entity force structures were to be submitted. This is known as the Scenario Force Structure (SFS) standard and is commonly referred to as the Order of Battle (ORBAT). The force structure developers annotate all their requested units and entities in the Scenario Force Structure based on the Phase 1 developed, BLCSE Master Entity List.

And finally, Phase 3 focused on the automation process for generating, and ultimately, rewriting, those configuration managed files from Phase 1. The idea behind Phase 3 is to ensure the latest updates are immediately checked into the BLCSE Configuration Management portal.

Using these three phases, the previously tedious and error prone Entity Data Management task became an efficient automated process.

As a follow on to implementing the Entity Data Management plan, it was realized that, with a minimal amount of additional information in the Scenario Force Structure, a scenario file could be generated.

Phase 1 - The Master Entity List (MEL)

The Master Entity List is an Excel spreadsheet populated with information pertaining to each of the platforms, life forms, and munitions that are used throughout the BLCSE Federation. This list contains all the data for each object that allows the heterogeneous simulators to accurately communicate.

This includes information for the High Level Architecture (HLA) Federation Object Model (FOM) file, OneSAF composition and mapping files, and the HLA Adaptor. This information is used throughout the integration process for heterogeneous simulation entity mapping and for generating the experiment execution vignette force structures.

Each column contains easily readable information regarding one instance of each life form, platform, and munition used throughout the BLCSE Federation. These columns are:

- The External Identifier. Because of the BLCSE FOM format used for BLCSE experiments, this is a DIS enumeration. Any entity coming in from an external source will have this Identifier.
- The OneSAF Name is the file name of the OneSAF Composition.
- The OneSAF Composition is the full directory path to the OneSAF composition files.
- The type column refers to the HLA Type. This is typically a platform, life form or munition.
- The final column is Direction. The direction describes whether the entity data is inbound, outbound, or bidirectional.

To summarize Phase 1 of the EDM task, entity data is collected from multiple heterogeneous simulation translators and OneSAF, and then combined into a Master Entity List.

Phase 2 - The Force Structure Template

The Force Structure Template is a blank Excel spreadsheet except for the column headers. BLCSE experiment participants populate the spreadsheet cells to depict the units and personnel that they would like to represent within their experiment scenario. Using the Master Entity List, they can ensure that the entities that they enter are indeed present and ready for the experiment. Additional information, such as the Army Capabilities Integration Center (ARCIC) ID, the common name, and the echelon information are all included.

The following is a list of the Force Structure column headers along with a description of what should be entered for each.

- Common Name – The common name can be any name that the Force Structure developer feels identifies the unit or entity.
- ARCIC Identifier – The ARCIC ID column is a twelve character unique identifier for each unit and entity represented in the simulation experiment. This column is automatically generated based on entries into columns C through N. Each character is meant to represent a host of additional information such as the site generating the object, unit name, unit echelon, and entity representation, including unit responsibility and placement, and the country that is being represented.
- C2 Role – The C2 Role describes the responsibility that entity has within its assigned unit.
- URN – The Unit Reference Number (URN) is a number assigned to each active military unit by the Department of Defense.
- LDIF Version/Date – The Lightweight Data Interchange Format (LDIF) Version and Date describes where the C2 Role and URN information is obtained.
- Alias Name – This is usually a familiar name that will be easily recognizable to all those participating in the experiment.
- Object Type – The object type specifies whether the object is a unit, life form, platform, or munition.
- Type – The type column describes what the object is in terms of its military classification. Some possible options are: Combat, Combat Service Support, Non-Combatant, Terrorist, etc.
- Category – Describes the military category of the object. For example; Infantry, Armor, Combat Support, Reconnaissance, etc.
- Behavior – This describes the normal behavior associated with the unit or entity and might include entries such as, groundTransport, Reconnaissance, targetAcquisition, etc.
- O.B. – The Order of Battle (OB) column is populated with the name of the OB if one is available.
- CDR – The CDR column, an abbreviation for commander, lists the ARCIC ID of the parent unit. This applies to units only. It is assumed that entities are entered directly under the unit to which they are assigned. Therefore, it is left blank for entities.
- Unit Composition – The unit composition should be the directory path to the OneSAF unit composition. This applies to unit compositions only.
- OneSAF/HLA Name – The OneSAF/HLA name should be the entity composition name without the .xml file extension. This applies to entities only.
- Entity Composition – This column should be the directory path to the OneSAF entity composition. This applies to entities only.
- Requested Entity (If Unavailable) – This column is used to request entities that do not seem to be available in OneSAF or the Master Entity List.

- Hitch/Mount To – The ARCIC ID of the vehicle to which to Hitch a trailer or Mount the entity on to.
- Comms Role – The comms role describes whether the entity will be used as a relay or to perform some other communications or network function.
- Entity DIS Name – The name of the entity as it would appear in a participating DIS native simulator.
- DIS Enumeration – The DIS, seven digit enumeration.

The completed Scenario Force Structure (SFS) is a very important part of the automation process. In essence, the completed Scenario Force Structure spells out all the forces and equipment that will be represented during the experiment.

Correct spelling, punctuation, and capitalization are vital to ensure the creation of an accurate scenario and proper heterogeneous simulator entity mapping. This is absolutely necessary because, based on what entities are entered into the SFS, validation must occur to ensure each entity has a OneSAF composition, and that each entity is represented in the Master Entity List.

To summarize Phase 2 of the EDM task, the completed Scenario Force Structure is submitted and error checked. Any new entity requests are documented and added to the Master Entity List.

Phase 3 – Rewriting the Mapping Files

All of the input files from Phase 1 are re-written based on any new entities that were requested, or discovered, during the Phase 2 Scenario Force Structure validation process. The results of Phase 3 are an updated and configuration controlled FOM and entity mapping files. These files are checked into configuration management for configuration control and are then ready for entity integration into the simulation experiment.

Automating Entity Data Management

Automating the Entity Data Mapping task entailed writing scripts capable of reading in several different file formats, and then parsing through each for representation of each entity, and finally writing out the Master Entity List and the updated mapping files. Figure 3 (bottom of this page) provides the process steps involved in all three phases of the automated EDM task.

Automating Phase 1 – Producing the MEL

Several different files needed to be merged into one human readable document, the Master Entity List. From OneSAF, there are three files that were used for the initial entity entry. (OneSAF was chosen because it is the primary environment simulator behind BLCSE experimentation).

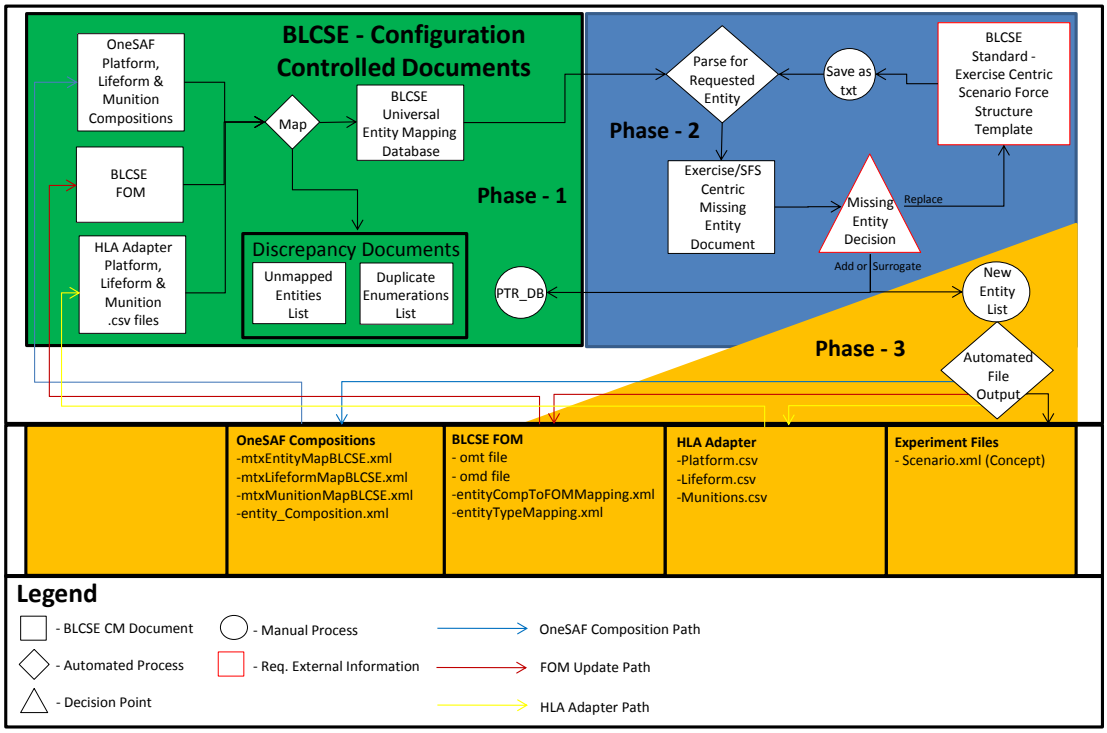

Figure 3 – Automating EDM

The OneSAF files are called:

- mtxEntityMapBLCSE.xml
- mtxLifeforMapBLCSE.xml
- mtxMunitionMapBLCSE.xml

Entity information, such as the entity name, composition, type, and direction were taken and written to an array. As you can tell from the file extensions, these are written in the Extensible Markup Language (XML). Therefore, entity information was extracted based on particular strings indicating where each entity's information began.

The HLA Adapter files were searched for each entity based on the entities from OneSAF already written to the array. Upon finding some representation of each entity, pertinent information was taken and added to the array per entity. The HLA Adapter files used are the:

- Platforms.csv
- Lifeforms.csv
- Munitions.csv

Notice that these files are all in the Comma Separated Value format. This indicates that each entity's information is on a line and the information is separated by a comma.

Finally, the BLCSE FOM was parsed for a representation of each entity. The FOM file used is the .omt file. Again, this file was searched for specific strings that would indicate where each entity's information began and could act as a cue for what data would be extracted. Once an entity was found within the FOM, that entity's data would again, be placed into the array so that it stayed associated to the entity.

Once all the input files have been parsed and the entity data entered into the array, it would then be written out to the Master Entity List.

There are, of course, exceptions. What if there is an entity in the array that was not in the FOM? What if there is an HLA Adapter entity that is not in OneSAF? That's where the error checking and reporting comes into action. Along with the Master Entity List, two files are written describing whether an entity has no mapping, or, and just as problematic, if the entity has duplicate mappings. This provides the EDM engineer with information pointing to where there may be problems within the mapping files.

Automating Phase 2 – Process the SFS

The key aspect of Phase 2 is the receipt of the completed Scenario Force Structure. This part of the process entails comparing the submitted Scenario Force Structure to the Master Entity List and to all the available OneSAF compositions.

In a perfect world, the SFS developer would have populated the Scenario Force Structure based on the Master Entity List. This is not a perfect world. The Scenario Force Structure typically contains several errors in multiple locations. These typically include misspellings, capitalization, and punctuation errors. An automated error checking script was written to identify and log those objects in the Scenario Force Structure that are not in the Master Entity List and do not have a OneSAF composition. The error log gives the engineer a place to look in the SFS that probably contains an error.

If the Scenario Force Structure is complete and without errors, the EDM process ends here. However, if there are entities listed that are not mapped in the Master Entity List or are not represented with a OneSAF composition, a couple of tasks must still be executed that are largely manual and may require the experiment manager's guidance.

First, unmapped entities are manually entered into the Master Entity List spreadsheet. The responsible engineer must research the entity for the appropriate DIS enumeration, and then enter all the data in the respective columns. The tasks executed during Phase 3 will update all the simulator and gateway mapping files. The experiment manager's guidance is not necessary for this.

Next, if an entity specified in the Scenario Force Structure does not have a corresponding OneSAF composition, several actions need to occur. The Scenario Force Structure developer should be contacted and a determination needs to be made as to whether;

a. It was entered in error
b. They would like to replace the entity with an existing composition, or,
c. The composition needs to be created.

In all cases, a Problem Tracking Report (PTR) is entered into the BLCSE PTR database and coordination between the Scenario Force Structure developer, the experiment manager, and the EDM engineer is required. If the decision is either a. or b., the EDM engineer simply enters the corrected or surrogated data into the Master Entity List. However, if it's decided that a composition should be developed, that responsibility usually falls on the site responsible for generating the SFS and will probably take anywhere from one day to a couple of weeks. Regardless of which decision is reached, the EDM engineer can enter the appropriate data into the Master Entity List when the data is available. However, the automation script will continue to generate an error on the missing composition until it is added.

Automating Phase – Producing the Maps

Phase 3 is the automatic development of all the mapping files used by all the systems with all the known entities that are to be used in the experiment.

Using the missing entity list output log from Phase 2, the EDM engineer manually enters the new entity's information into the Master Entity List. This process was left as a manual process due to the wide range of information that must be researched for a new entity. The new entity name, OneSAF composition, and DIS enumeration are all required. And, in order to acquire an appropriate DIS enumeration, DIS Standard 2.0.4 is referenced to ensure an accurate DIS enumeration representation across the federation.

Once the new entities are entered into the Master Entity List, a script is executed that automatically enters the new information into all the required mapping files. This includes;

- The OneSAF files
  - mtxEntityMapBLCSE.xml
  - mtxLifeformMapBLCSE.xml
  - mtxMunitionMapBLCSE.xml
  - entityCompToFOMMapping.xml
  - entityTypeMapping.xml
- The BLCSE FOM files
  - The .omt file
  - The .omd file
- The HLA Adapter files
  - Platform.csv
  - Lifeform.csv
  - Munitions.csv

These completed files are finally checked into the BLCSE configuration management portal for use in the BLCSE experiment.

Finally, based on the completed mapping files that are now configuration controlled, a new Master Entity List is generated to ensure accuracy across the federation.

EDM Automation Summary

The Entity Data Mapping automation process is accomplished using Python scripts. Python scripts can be quickly written and are very flexible regarding what file formats are able to be ingested and written out. Additionally, the scripts are quickly and easily tested and modified because compiling is not required. Considering the nature of these Python scripts, all scripts are best run from the command line.

To summarize and simplify (See Figure 4, at right), the process takes the federation mapping files, concatenates all entities into a single document, parses the units and entities that are to be used in the experiment, finds and documents unmapped entities, and then rewrites the files.

Automated Scenario Generation

Automating the scenario generation task entails utilizing input from the completed Scenario Force Structure and writing that information into the XML formatted MSDL Scenario. While the primary simulation environment behind BLCSE experimentation is OneSAF (OneSAF has its own scenario format), it was still decided that MSDL would be used; so that, should additional systems gain MSDL functionality, they would also benefit from the Scenario Generator.

The scope of the MSDL standard, as outlined in the Simulation Interoperability Standards Organization (SISO) Standard 007-2008, is to provide the mechanism that permits simulations to utilize the MSDL schema to develop and reuse military scenarios across MSDL compliant simulations and scenario generation tools (MSDL PDG, 2008). It further explains that the MSDL format will support the Modeling and Simulation community by providing a common mechanism for verifying and loading military scenarios. It will also support the ability to create reusable scenarios that can be shared across federated simulation domains between simulation systems and C2 devices. The description of MSDL, as provided in the SISO standard, is precisely the scenario development result that would greatly benefit the BLCSE simulation federation.

The key to the scenario automation process is an accurate Scenario Force Structure. As previously described, the Scenario Force Structure is essentially a list of the units, personnel, and equipment that each BLCSE Federate would like to use during the experiment. The highest echelon unit is listed first, with the next echelon level unit placed directly below.

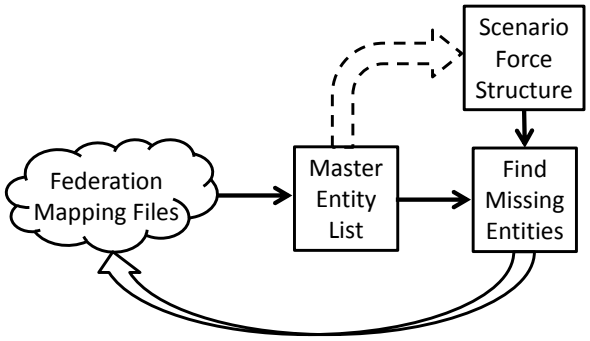

Figure 4 – Simplified EDM

Additionally, listed below each unit, are the personnel and equipment that are assigned to that unit. A representative simplified, platoon level force structure is provided in Figure 5.

1st Platoon	Unit
Plt. Leader	Entity
Plt. Sergeant	Entity
1st Squad	Unit
Sqd. Leader	Entity
Fireteam A	Unit
Rifleman 1	Entity
Rifleman 2	Entity
Grenadier 1	Entity
Machine Gun 1	Entity

Figure 5 – Force Structure

The Scenario Force Structure is completed by each participating site using the Microsoft Excel database format. Unfortunately, this does not provide for optimal results within Python scripts. Therefore, the scenario engineer must save the scenario as a Comma Separated Value (CSV) file. This allows the script to easily read in each field and place the data into various arrays.

There are several arguments that must be entered into the scenario generator script for a usable scenario to be generated, for example, the name of the SFS file, and the terrain database on which to place the units and entities, and the echelon of the highest unit in the scenario. The scenario development engineer must have a good understanding of the experiment order of battle as he or she creates the scenario.

As with the EDM process, the scenario generation scripts can be run from the command line. However, a Scenario Generator Tool user interface (Figure 6, at right) has been created to allow for quick execution and error free MSDL scenario generation.

As is shown in Figure 6, the MSDL Scenario Generator requires information in only three text boxes. And, to make matters simpler for the scenario engineer, all these are automatically populated through either file browsing or selecting from lists.

The first box is for the Scenario Force Structure name. This is selected by utilizing the 'Browse' button. The file selected needs to be the CSV formatted version.

The second box is a list box that provides the user with a selection of OneSAF Terrain Format (OTF8), terrain databases. The terrain database used for the scenario must be listed. The terrain extents for these are outlined in the script so that the units and entities from the SFS are emplaced within the terrain database extents.

The third box is the echelon level of the highest unit in the SFS. This is required to properly set up the parent and sub-unit information within the MSDL scenario. Again, the echelon is selected from a drop down list so that there is no ambiguity about what may be entered.

Step 1 – Click the 'Validate Force Structure' button
Validation is automatically accomplished with one script. First, each entity in the SFS is compared to the Master Entity List. If the entity is not in the Master Entity List, the scenario developer is notified via screen output that a log file has been created. Next, the script ensures that a OneSAF composition is available for each unit and entity. Again, if no composition is found, the unit or entity is documented and the scenario developer is notified. If there are no unmapped entities and all the units and entities have a OneSAF composition, the actual MSDL Scenario generation process can begin.

Step 2 – Click the 'Generate Scenario' button
Logically, the 'Generate Scenario' button executes the scenario generation script. This script reads in the CSV formatted scenario force structure and creates an MSDL formatted scenario.

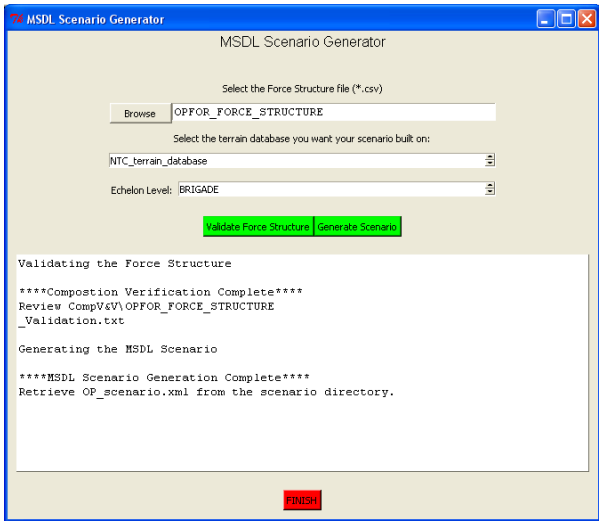

Figure 6: The MSDL Scenario Generator

MSDL Scenario Generation Summary

The scripts that are used to validate the Scenario Force Structure and then generate the scenario were written in Python. The user interface was written in Tkinter with links to the driving python scripts done in TCL. While the force structure and scenario generation scripts can

be executed from the command line, the user interface lets the scenario developer quickly validate the SFS and MSDL scenario in true Windows fashion.

To simplify and summarize, the MSDL Scenario Generation process quickly reads in a file's data, verifies that it's correct, and writes it out in MSDL format (Figure 7).

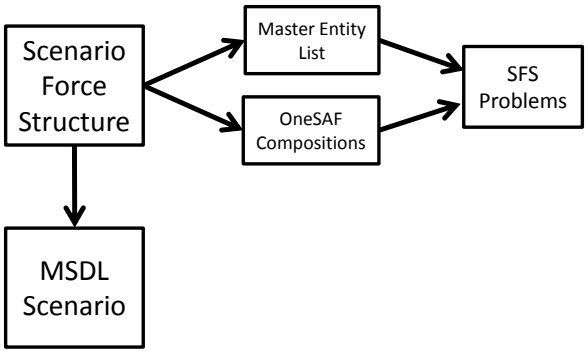

Figure 7 – Simplified Scenario Generation

As indicated in Figure 7, one branch of the process provides information that pertains to problems with the Scenario Force Structure. The majority of these issues are usually problems with the spelling or capitalization in the OneSAF composition column. While it cannot be automatically fixed, it does point the engineer to the issue. Occasionally, the issue is that there is not a valid OneSAF composition for the entity listed in the SFS. If that is the case, the SFS must go through the previously described EDM process to rectify the issue. A SFS with no errors can be very quickly turned into an MSDL scenario.

THE RESULTS

The new EDM process and the Scenario Generator tool has now been used in two major BLCSE experiments. And, while there were hiccups and bugs, the tools saved weeks of labor by automating tasks that were previously manually intensive. The tools significantly reduce integration time and risk associated with heterogeneous simulation experimentation.

EDM Use Case

For a typical experiment, there are ten to twenty new entities that need to be mapped to the various simulators and across the network. The following graphics display the time and labor required of this task, with one displaying the timeline when using the old method and the other displaying the timeline with the new method.

The following figure (Figure 8) displays the task of adding an entity using the old method.

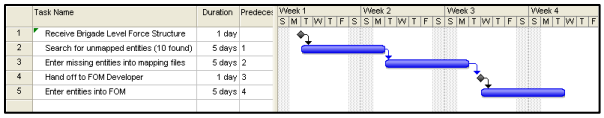

Figure 8 – Old Timeline for Adding Entities

Notice in the graphic that the time to process a single Brigade level scenario was about three and a half weeks utilizing the time of one simulation engineer and one FOM programmer. Remember that this task was to be repeated for each Brigade level scenario that was to be executed. So, considering that there are five or more Brigades represented in these large scale experiments, this results in close to sixteen weeks of error-prone, entity mapping labor. Also realize that, due to the manually intensive aspect of this task, human error is likely to cause additional labor to be required for the force structure.

Now, Figure 9 displays the new process. Notice first that the timeline is changed from weeks to hours within a single day. The tasks involved are slightly modified in that no additional FOM development is needed.

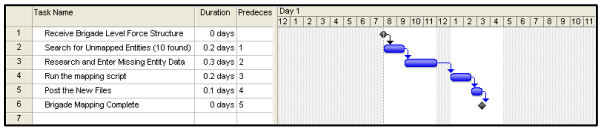

Figure 9 – New Timeline for Adding Entities

These two figures clearly display the drastic impact that can be realized by using BLCSE's automated entity mapping process. The same force structure, submitted with several errors, can be integrated and ready for the experiment is less than a full day, and through the utilization of only one engineer. As Figure 9 depicts, the simulation engineer can even take an hour long lunch break.

This new, automated process has been proven during the integration events associated with the last two large scale BLCSE simulation experiments.

MSDL Scenario Generation Use Case

For this use case, let's assume that all the units and entities entered into the submitted brigade level force structure are accurate and that all the compositions exist.

Prior to the new, automated process, creating a brigade level OneSAF scenario would be accomplished by a simulation engineer deciphering the units and entities

from the force structure, and then painstakingly emplacing each unit and entity onto a OneSAF plan view display map. Finally, all units and entities would be edited to ensure that it had the correct identifier. As indicated in Figure 10, this process would take about a week and a half and in some cases much longer.

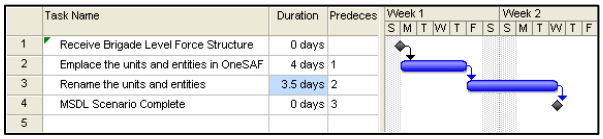

Figure 10 – Old Scenario Creation Timeline

As with the EDM process, the timeline has changed from weeks and days to hours within a single day. As long as the Scenario Force Structure is correct and the composition is present, creating an MSDL scenario now takes less than half a day, as indicated by Figure 11.

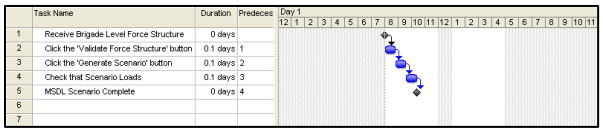

Figure 11 – MSDL Scenario Creation

Using BLCSE's automated Scenario Generation Tool reduces the amount of time required to create OneSAF scenarios and reduces human-error prone issues previously noticed in the experiment scenario creation process.

ISSUES AND RISKS

Throughout the implementation of these processes, several unexpected project risks arose that placed the whole automation effort in jeopardy. In fact, some of the problems are continuing, with no solution yet implemented. However, these are still very new processes, and solutions are currently being explored for development.

The biggest issue that was presented throughout the development process, and continues through normal use, concerns the Scenario Force Structure. While the automation scripts save an incredible amount of time on both the EDM and Scenario Generation processes, those areas where humans must enter data is still troublesome. The Scenario Force Structure must still be fully manually created. This is a time consuming process prone to human error and the Scenario Force Structure validation process relies on accuracy. Case sensitivity, spelling, and punctuation were, and continue to be, very important for two reasons. First, in the EDM task, in order for the script to find each entity to ensure that it is mapped, the spelling and capitalization must be exact. There are several instances of the same vehicle with only slight differences in the capitalization. Second, during the scenario generation task, the same spelling and capitalization issues occur. However, the OneSAF composition is also entered. And, while the forward and backslash directory separators between Windows and Linux issues were easily handled in the script, capitalization cannot be ignored. The engineers responsible for both the scenario generation and the entity mapping must spend several hours per submitted force structure verifying the entries. Luckily, the force structure validation script will point them to the lines that have problems.

The MSDL Scenario is written in XML format. So, this should be a very simple document to create once the required tags are known. Python provides a quick method for writing XML and this was used to make the XML code 'pretty'. The problem arose that OneSAF uses very specific tags and a non-standard MSDL header. In addition, the automation scripts were written as OneSAF was transitioned from versions, 3.0 to 4.0 to 5.0. During this change, the XML tags for each line changed causing a re-write of much of the script.

The automation of the EDM task was presented with a very high-risk during the rapid OneSAF changes from version 3.x to 5.0. During this time, the external identifiers of entities were changed from an HLA enumeration and name to a DIS enumeration. This caused errors on multiple levels within the automation scripts. All of the keys in the files that are read in to generate the Master Entity List changed. The information required in the Master Entity List changed, as did the format and data required of the mapping files. This all caused a major rewrite of the still-in-testing automation scripts. However, thanks to the ease of use and modification of Python scripts, this potentially catastrophic risk was overcome.

The highest risk to the automation process was the lack of testing prior to the tools being delivered. While both sets of tools were still in the Alpha and Beta test phases, the customer perceived the benefit and value of the tools and immediately requested that the tools be delivered for use. Bypassing software test processes for direct implementation for an experiment integration event caused many bugs to present themselves as scenario engineers at each of the BLCSE federation sites attempted to create their scenarios. And, as any software developer knows, once a product is perceived as unreliable, it is very difficult to reverse. However, the efficiency, benefits, and reliability of the automation tools have made them all the standard procedure for BLCSE events.

CONCLUSION

The utilization of automation scripts to quickly integrate entities and create scenarios for major Army heterogeneous simulation experiments has changed the way in which integration events are executed. Simulation engineers previously assigned these tedious, time consuming tasks can now be utilized for more pressing issues. To be sure, the entirety of the integration process is now more efficient.

The Entity Data Management integration task was tedious and challenging. An inordinate number of hours were spent manually searching through various formats of entity data and then trying to map the information into various files. Additionally, experiments were put at risk due to human error that is bound to occur with manually intensive tasks such as these. Automating the EDM task drastically reduced the time required to map entities across the federation. The EDM task was previously a continuously repeating high-risk task. Thanks to the new Entity Data Management process and the automation scripts that have been implemented, this is now a low risk, rapidly accomplished, secondary task.

The decision to automate the scenario generation task for the BLCSE federation was very beneficial. By implementing a series of automation scripts along with a user interface, the scenario creation task has realized far greater efficiency in a fraction of the time that this task had previously taken. Historically this has been a trial and error type of task that could drag on for multiple weeks. The automation script generates error-free MSDL scenarios that are correct the first time and completed in a matter of hours.

Processes to further reduce human error and increase efficiency for both of these tasks are continually being explored for feasibility. The goal is to make the Army BLCSE experiment integration process as efficient and error-free as possible.

ACKNOWLEDGEMENTS

I would like to acknowledge the following individuals and agencies for their support. Steve Wurster, my manager, for assigning me the task to reorganize EDM and for setting me up for success. Margie Woodruff, the Scenario Engineer from the BLCSE Experiment Support Section, for her contribution to the Scenario Force Structure formatting and refinement and script testing. Khoi Do and Matt Wintercorn, OneSAF programmers from the BLCSE Federation Management Support Section, for their contributions in troubleshooting and deciphering OneSAF's MSDL capabilities. And finally, the Government and contractor employees of the BLCSE Experiment Support Group for Beta testing and patiently using the automation tools during experiment integration events.

REFERENCES

Dayley, Brad (2007), Python Phrasebook, Indianapolis, Indiana: Sams Publishing.

Purdue University On-line Writing Lab, (2002). *Using American Psychological Association (APA) Format.* Retrieved March 13, 2009, from http://owl.english.purdue.edu/handouts/research/r_apa.html#Examples

MSDL PDG, (2008). Simulation Interoperability Standards Organization (SISO) Standard for: Military Scenario Definition Language SISO-STD-007-2008. Retrieved 7 Feb. 2011 from: http://www.sisostds.org/ProductsPublications/Standards/SISOStandards.aspxhttp://www.sisostds.org/ProductsPublications/Standards/SISOStandards.aspx

Verzuh, Eric (2008), The Fast Forward MBA in Project Management, 3rd Edition, Hoboken, NJ: John Wiley and Sons, Inc.

Welch, Brent B., Jones, Ken, & Hobbs, Jeffrey (2003), Practical Programming in TCL and Tk, 4th Edition, Upper Saddle River, NJ: Prentice Hall PTR

Wittman, Robert L. & Abbott, Jeff (2006), Keeping up with The Military Scenario Definition Language (MSDL), Retrieved 13 Apr. 2011 from http://www.onesaf.net/community/documents/Papers_Presentations/Published/SIW06S_018_KeepingUpWithMSDL.pdf