Encoding Plans Using the Military Scenario Definition Language (MSDL)

Tabitha Arcila, Dr. Douglas Reece
Applied Research Associates, Inc.
Orlando, FL

Sponsored by Defense Advanced Research Projects Agency (DARPA) Information Processing Technology Office (IPTO) Program Name: Deep Green. Issued by Applied Research Associates. The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

ABSTRACT

The Defense Advanced Research Projects Agency (DARPA) sought a common data language to support their Deep Green research program – a program which aims to develop a simulation-based Course of Action (COA) analysis tool for brigade commanders. Given a limited set of available languages, Military Scenario Definition Language (MSDL) was proposed as a logical choice for the program. However, MSDL had never before been implemented to run fully automated simulation software. Thus, Deep Green stakeholders were both optimistic and apprehensive about developing software around a prototype data language.

Despite the fact that MSDL was on the verge of standardization, it quickly became evident that the existing version 1.0 was not adequate for representing the brigade-level courses of action (COAs) that were at the heart of Deep Green. In order to make MSDL viable, a new version (designated MSDL-DG) came into being. This new MSDL variant took months to define and even longer to implement.

This paper gives an introduction to the Deep Green program, and describes some of the problems encountered when encoding brigade-level COAs into MSDL. Included are some of the encoding conventions that were adopted, MSDL schema modifications that were implemented, innovative tools and processes used in the encoding process, and insights offered for future data representations of COAs – with a focus on Battle Management Language (BML).

ABOUT THE AUTHORS

Tabitha Arcila was part of the Test and Evaluation team for Phases I and II of DARPA's Deep Green project. She obtained a Bachelor's Degree in Mathematics from Rollins College and a Master's Degree in Operations Research from the University of Central Florida. Tabitha began working for Applied Research Associates in 2008, where her primary focus has been in experiment design, metrics development, data analysis, and testing.

Dr. Douglas Reece is a Principal Scientist at Applied Research Associates. He has been a principal in the Test and Evaluation effort on DARPA's Deep Green program for 2 years. Dr. Reece has been investigating and developing behavior models and intelligent components for military simulations for over 15 years.

Encoding Plans Using Military Scenario Definition Language (MSDL)

Tabitha Arcila, Dr. Douglas Reece
Applied Research Associates, Inc.
Orlando, FL

INTRODUCTION

Prior to the research conducted on DARPA's Deep Green program, Military Scenario Definition Language (MSDL) was somewhat of a theoretical scenario initialization language whose envisioned purpose was to load scenarios into systems simulating military operations. During the course of our research, we were able to implement a modified version of MSDL in order to not only initialize, but run, a fully automated military simulation.

As one might expect, the generation of complex MSDL data files required the development of innovative processes and software tools. Our research, processes, tools, and suggestions for future data representations are described in this paper.

DEEP GREEN

Deep Green is a DARPA research program designed to provide decision-making support to U.S. military commanders. The premise behind Deep Green is to ensure that commanders always have options at their disposal during the course of a battle. During plan execution, updates from real-life, ongoing military operations allow Deep Green to estimate the likelihood of success or failure for a commander's plan. Rather than reacting to actions taken by the enemy, Deep Green technology would allow commanders to continuously plan and re-plan throughout a battle. In addition, a rapid, high-fidelity simulation tool allows commanders to make modifications to plans and explore the execution of options through what-if analyses.

Two competing development teams were tasked with building modular software systems to include three main components of Deep Green – Commander's Associate, SimPath, and Crystal Ball. See Figure 1 for a system overview.

Commander's Associate is the user interface responsible for ingesting a commander's plan via multimodal sketch and speech recognition. A commander uses a highly responsive touch-screen tablet to draw a plan while verbally elaborating on plan details via a headset. The combined sketch and speech input is recognized by Commander's Associate and converted to a machine readable language.

The SimPath component accepts the machine readable COAs as input and simulates the unit actions and engagements that ensue. At a minimum, one friendly COA and one enemy COA are required as input. SimPath uses its fully automated simulation engine to wargame the COAs and ultimately produce a graphical representation of all possible outcomes. This graphical representation is called a "futures graph".

Next, Crystal Ball reasons over the outcomes contained in the futures graph and calculates the likelihoods that each will occur. An overview of the futures graphs along with analysis tools are provided back to the commander via Commander's Associate. The highly advanced user interface allows the commander and staff to explore the futures and analyze the wargames in depth. The commander can use the information presented to decide upon a successful Course of Action.

In order to provide test data to the two competing Deep Green development teams and maintain system modularity, a common data language was essential for this program. All candidate languages had to be somewhat well-defined, capable of supporting the complex elements of a brigade-level COA, and able to interchange data with C4I devices.

MSDL

In the Deep Green Broad Agency Announcement (BAA), DARPA stated a desire to use MSDL to interchange information between the Deep Green components. In addition, MSDL was theoretically able to be integrated with OneSAF, the exercise driver for all three phases of the research program. Thus, selecting MSDL as the language for Deep Green would simplify the conversion of test data generated by OneSAF for the Deep Green components. Moreover, MSDL was in the process of being standardized by the Simulation Interoperability Standards Organization (SISO).

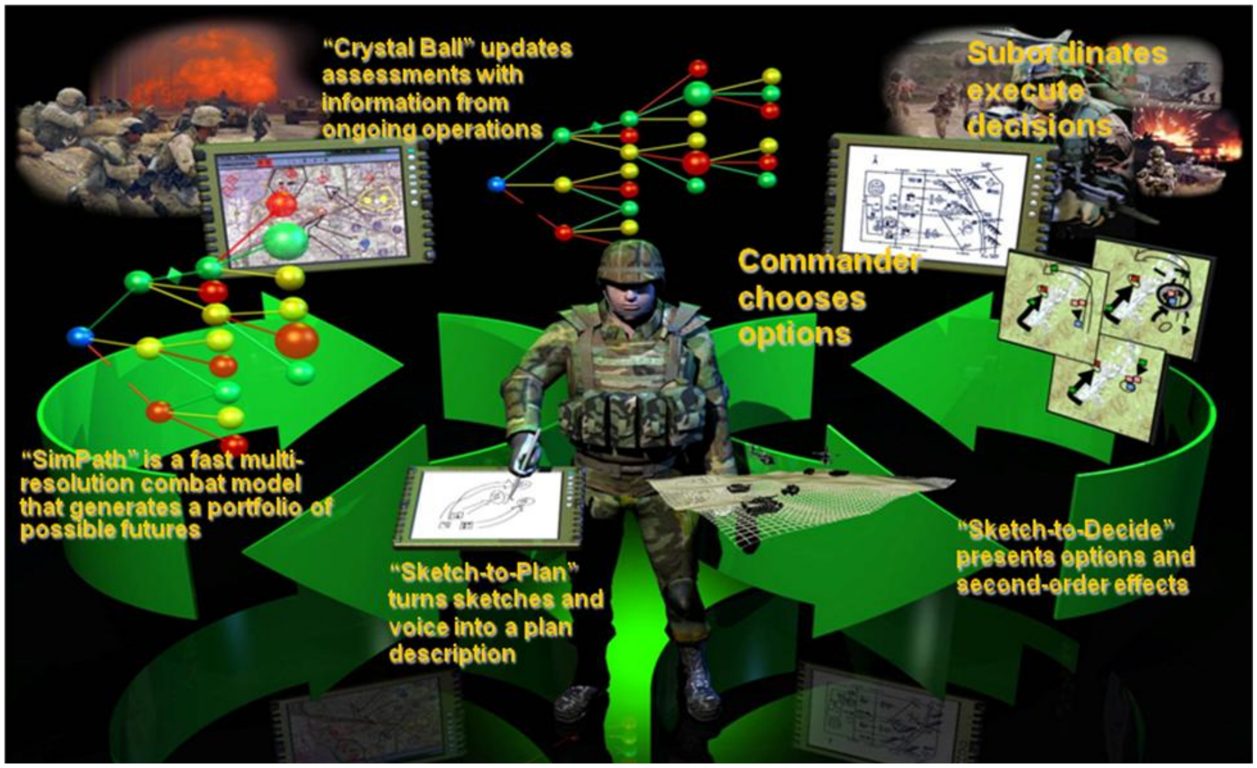

"Crystal Ball" updates assessments with information from ongoing operations
Subordinates execute decisions
Commander chooses options
"SimPath" is a fast multi-resolution combat model that generates a portfolio of possible futures
"Sketch-to-Plan" turns sketches and voice into a plan description
"Sketch-to-Decide" presents options and second-order effects

Figure 1. Overview of the Deep Green system (DARPA, 2007).

MSDL Background

MSDL is an XML-based language created for the purpose of loading and exchanging military scenario information within simulations and C4I devices. Although MSDL was originally exclusively utilized by OneSAF, the vision is that other simulations will soon follow suit. Initially, the theoretical use case for MSDL in OneSAF is simply to initialize a scenario. For example, an MSDL data file would provide the units, locations, equipment, scenario date and time to load into a simulation environment. Following initialization, human pucksters would monitor and control the simulation by tasking units for the duration of the simulation.

MSDL Schema

The standardized MSDL v1.0 schema includes scenario elements such as: Scenario Date/Time, Overlays, Coordinate Data, Weather, Units, Equipment, and Tactical Graphics. While the v1.0 schema is useful for describing the environment of the scenario and setting the stage for a simulation, there are no elements for unit tasking and task sequencing. This information is critical for Deep Green's SimPath component to run simulations in an automated fashion, eliminating the need for human intervention (pucksters).

MSDL MODIFICATIONS

Using MSDL v1.0 as a starting point, Deep Green contractors collaborated on how to incorporate the missing plan elements into a modified MSDL schema. An earlier version of MSDL (MSDL v0.0) was discovered, which included many of the necessary elements for Deep Green that MSDL v1.0 was lacking. It seems that the removed elements were not well-defined at the time of MSDL's standardization and therefore do not appear in the approved v1.0 schema. In addition, we realized the need for supplementary information in order to support operationally correct unit tasking.

MSDL-DG

Our modified version of MSDL, coined MSDL-DG, was born. The new version provides a means to represent all of the required military Course of Action elements, including task information necessary to support SimPath. A comprehensive discussion of all of the elements that were added, modified, and deleted from the MSDL v1.0 schema is included in "Extending the Military Scenario Description Language (MSDL) to Represent Deep Green Course of Action Descriptions" (Lacy, et al., 2010).

Task Hierarchies and Conditional Execution

In particular, one of the most noteworthy structural features that proved useful in representing dynamic COAs was the inclusion of hierarchical tasks. The hierarchy displayed in Figure 2 supports the decomposition of plans and allows various levels of the plan to be synchronized. This is an important feature for plan representation that is currently missing from Battle Management Language (BML).

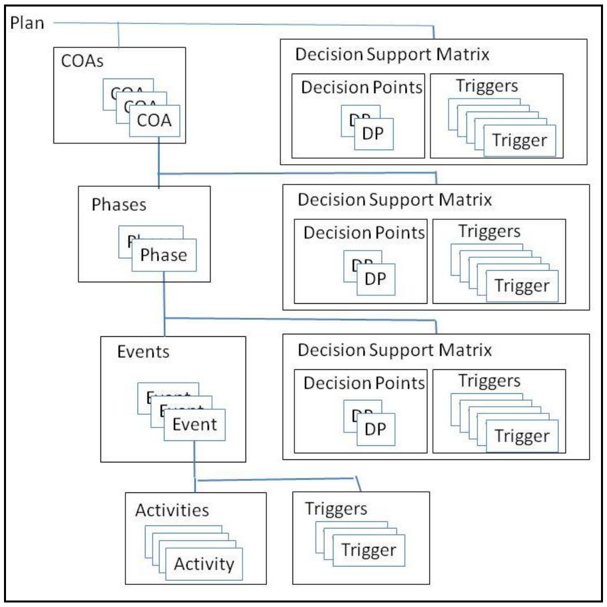


Figure 2. Subtask hierarchy for plans in MSDL-DG.

At each level of the task hierarchy in MSDL-DG there is a “decision support matrix” (DSM) that contains Decision Points (DPs) and Triggers. Tasks reference DPs at the same level, and those DPs refer to Triggers in the same DSM. Conditions in the triggers can cause the associated task to become activated. Thus, the inclusion of DPs and Triggers directly support a commander’s Decision Points described in a COA, in addition to “on order” tasks for units.

Following months of collaboration, Deep Green developers were forced to stop making schema modifications in order to concentrate on maturing the software components. Although MSDL-DG was not perfected, the Deep Green teams agreed to work with the imperfect schema. Without functional software components or sample test data, it was difficult to anticipate whether the MSDL-DG schema would suffice for Deep Green. The only way to tell for sure was to actually encode a plan.

ENCODING A PLAN

Capturing a plan in MSDL involved many steps. Since we were the first to generate functional MSDL data files, there was no precedent. We established processes and encoding conventions as we went, continually creating new ways of doing things. As manual methods grew cumbersome, we quickly realized the need to develop automated tools.

In addition, communication between military subject matter experts (SMEs), the development teams, and the test/evaluation team was critical. SMEs decided which elements were critical to military plans, while the development and test/evaluation teams determined which plan aspects could be realistically described in machine language. Compromise became necessary, as SMEs learned that many facets of a military staff’s handwritten plan would not be understood by a computer (or at least not with the current version of MSDL-DG).

COA Origination

The COAs generated by the subject matter experts (SMEs) were simplified operational orders. In the first phase of the Deep Green program, the evaluation focused on maneuver elements. Thus, the critical aspects of the COAs were the maneuver sketch, the tasks to maneuver units, and decision points. Figure 3 shows a typical sketch (without terrain), which includes tactical task graphics as well as control measures. Figure 4 shows accompanying text with tasks to maneuver units. Note that it includes “on order” tasks. Finally, Figure 5 shows the commander’s Decision Points in the COA; these are conditional tasks and task reorganizations made at certain points in the COA (illustrated by stars on the sketch).

Knowledge Engineering

The SMEs sent the COA drafts to the test/evaluation team for review. All of the COA elements were converted manually to their corresponding MSDL-DG representation and organized in a spreadsheet. Clarification was sought from the SMEs for all ambiguous COA aspects, and substitution was often required for those aspects that did not translate easily to MSDL-DG.

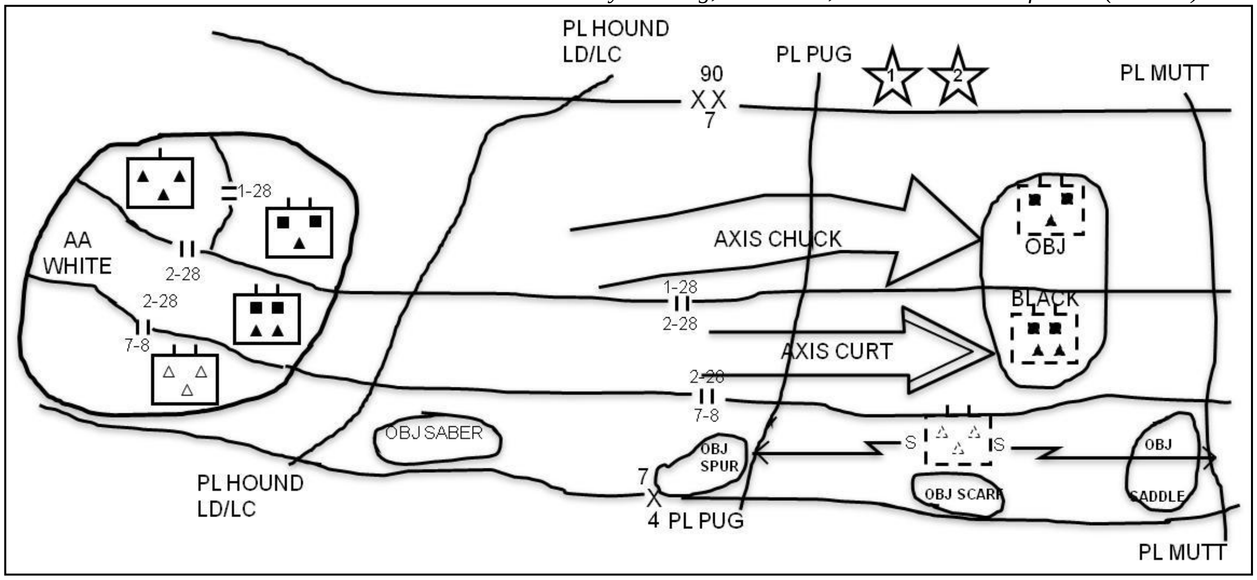
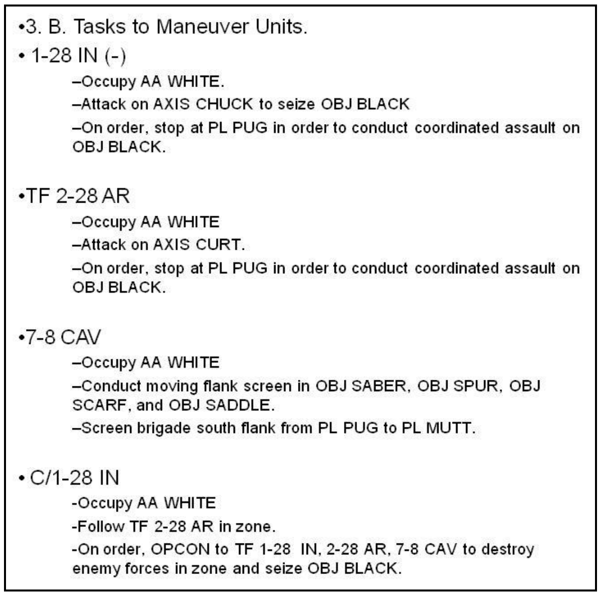


Figure 3. Sample COA sketch provided by the SME.



•3. B. Tasks to Maneuver Units.

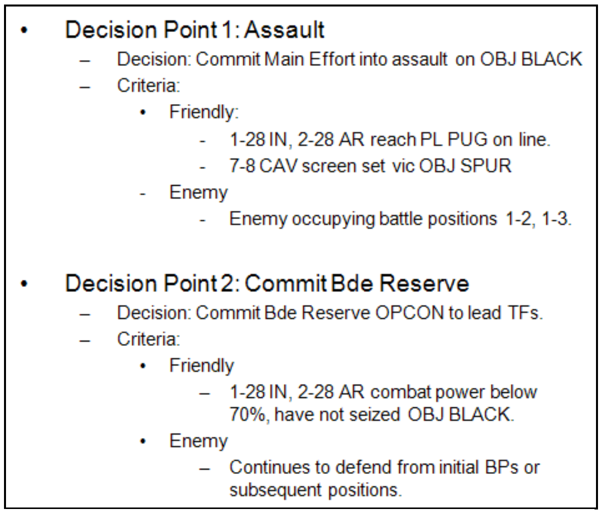
• 1-28 IN (-)
- –Occupy AA WHITE.
- –Attack on AXIS CHUCK to seize OBJ BLACK
- –On order, stop at PL PUG in order to conduct coordinated assault on OBJ BLACK.

•TF 2-28 AR
- –Occupy AA WHITE
- –Attack on AXIS CURT.
- –On order, stop at PL PUG in order to conduct coordinated assault on OBJ BLACK.

•7-8 CAV
- –Occupy AA WHITE
- –Conduct moving flank screen in OBJ SABER, OBJ SPUR, OBJ SCARF, and OBJ SADDLE.
- –Screen brigade south flank from PL PUG to PL MUTT.

• C/1-28 IN
- -Occupy AA WHITE
- -Follow TF 2-28 AR in zone.
- -On order, OPCON to TF 1-28 IN, 2-28 AR, 7-8 CAV to destroy enemy forces in zone and seize OBJ BLACK.

Figure 4. Tasks to maneuver units in sample COA.



- Decision Point 1: Assault
  - Decision: Commit Main Effort into assault on OBJ BLACK
  - Criteria:
    - Friendly:
      - 1-28 IN, 2-28 AR reach PL PUG on line.
      - 7-8 CAV screen set vic OBJ SPUR
    - Enemy
      - Enemy occupying battle positions 1-2, 1-3.
- Decision Point 2: Commit Bde Reserve
  - Decision: Commit Bde Reserve OPCON to lead TFs.
  - Criteria:
    - Friendly
      - 1-28 IN, 2-28 AR combat power below 70%, have not seized OBJ BLACK.
    - Enemy
      - Continues to defend from initial BPs or subsequent positions.

Figure 5. Decision Points in sample COA.

Tasks

Many of the unit tasks in the COAs were easily encoded in MSDL-DG. To codify the constraints of military doctrine, the specification document for MSDL-DG was extended with tables that list legitimate values for task attributes (“What”, “How”, “Why”) and also which combinations of values were allowed. Further, since “Where” attributes in the COAs were generally indicated by graphics, the tables describe which graphics are required or optional with each task. For example, an “Assault to Clear” task contains a valid combination of “How” and “What”. It also must reference either a Clear graphic or an Objective graphic, and may be further specified by left or right Boundary Lines, an Assault Position, an Axis of Advance, a Direction of Attack, or a Phase Line (Applied Research Associates, 2009).

In addition to constraints on task attributes and required graphics, we encountered several task encodings that required *business rules* to define the semantics of the graphics associated with the task. For example, a task assigned to the 7-8 CAV unit that reads “Conduct moving flank screen in OBJ SABER, OBJ SPUR, OBJ SCARF, and OBJ SADDLE” was intended by the SME author to mean that 7-8 CAV would screen for the main effort, reaching the objectives in the order listed. See Figure 3 and Figure 4. Thus we defined a rule that a Screen task referencing multiple Objective graphics should be executed by advancing the unit to the Objectives in sequence.

Events

The subtask structure shown in Figure 2 allows mission phases and conditional tasks to be represented several different ways. For the Deep Green COAs, we encoded all base tasks to subordinates in one Event, and used separate Events for tasks triggered by DPs or "on order" conditions. Mission phases were implemented as different Activities in the same Event. We did not use MSDL Phases or Events to define mission phases. While this approach may not have used the MSDL elements as intended, we felt it was simplest to use DPs at the lowest level possible.

With an infrastructure such as MSDL provides, there are many possible ambiguities in execution that have to be resolved through the use of conventions. For example, if an Activity or Event does not reference a DP or trigger, when does it get executed? What happens if two Events are simultaneously triggered? In this project, we defined some conventions to guide COA encoding and execution, such as:

- A unit is allowed to have only one active Activity at a time. If a new Activity is triggered before an executing activity is completed, the new Activity preempts the current one.
- All Activities that do not reference Triggers are started whenever the Event that contains them is started.
- One Event is given the sequence number 0. This is the Event that starts active.

Triggers

Triggers are effectively predicate functions used to start Activities and Events. MSDL-DG provides for several types of triggers, including:

- Movement Event
- Friendly Event
- Combat Strength (friendly unit or enemy in an area)
- Enemy Activity
- Boolean – And, Or, and Not

There are time relations and reference points in the trigger representation as well, so triggers can be satisfied "Before", "At/During", or "After the Start or End" of one of the above events. For example:

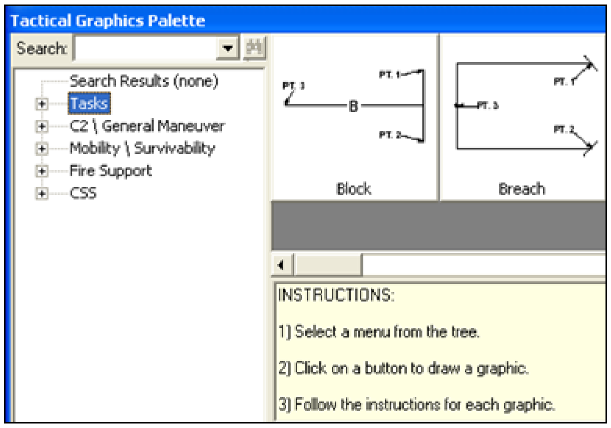
- Several DP conditions were based on a force having seized an objective. Since there is no direct trigger condition for "Seize", we created triggers on the operational definition of Seize: a unit occupying the objective and having no threats on the objective. This condition was encoded as a "Movement Event [After] unit U is Entering OBJ" AND NOT "Enemy Event [During] a Platoon is performing Any-activity on the OBJ."
- Many COAs contained DPs that committed the brigade reserve to either of two units, if one of the two units fell below 70% strength. However, one unit had priority over the other. We encoded this as two separate DPs, which included conditions that verified the strength of both units prior to commitment. This ensured that the reserve did not commit to the lower priority unit if the higher priority unit was also under strength.

TOOLS

After all of the COA particulars were encoded into MSDL-DG elements and captured in the spreadsheets, we used various software tools to produce the MSDL-DG files from the spreadsheets. Because we were pioneering the MSDL data file generation process, these tools were often hastily modified as our needs evolved.

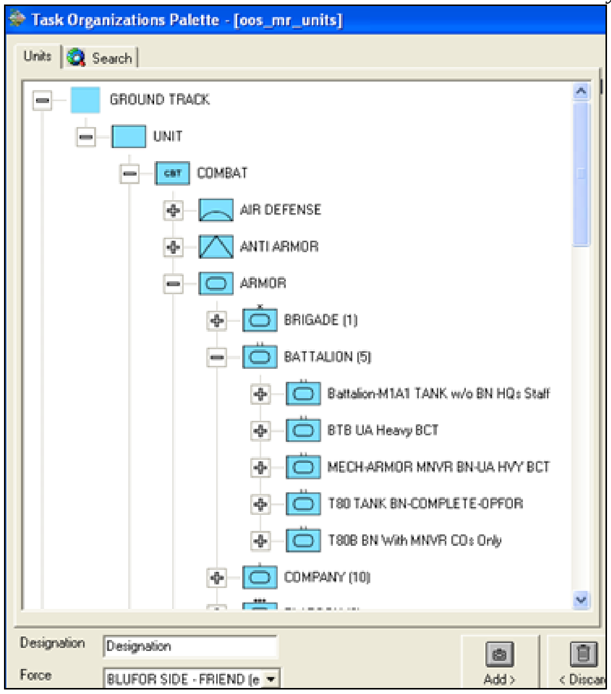
MSDE

An add-on to Microsoft PowerPoint, called Military Scenario Development Environment (MSDE), was extremely useful for converting SMEs' hand-drawn sketches to MSDL-DG. MSDE enabled us to re-draw the COA sketches using the tool's built-in unit and tactical graphic libraries. Also, the topological map overlay was helpful for capturing unit and graphic coordinate data. See Figure 6 and Figure 7.



INSTRUCTIONS:
1) Select a menu from the tree.
2) Click on a button to draw a graphic.
3) Follow the instructions for each graphic.

Figure 6. MSDE's Tactical Graphics library.


Task Organizations Palette - [oos_mr_units]

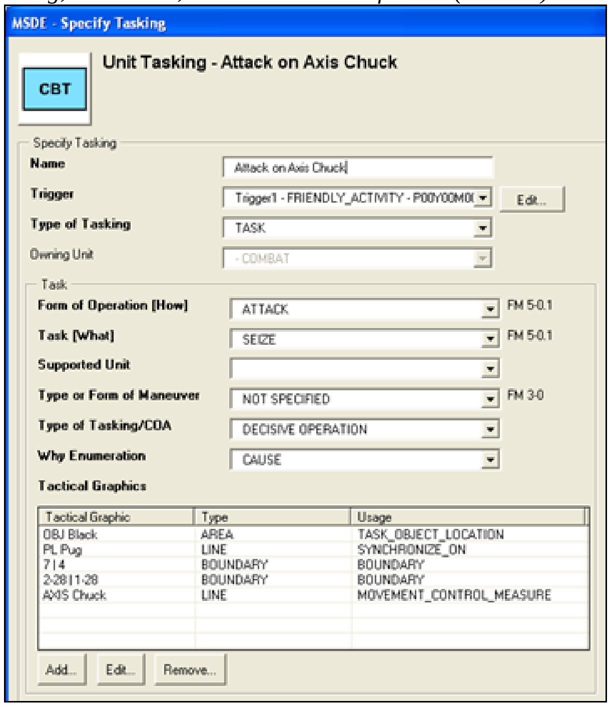
Units | Search

- GROUND TRACK
  - UNIT
    - COMBAT
      - AIR DEFENSE
      - ANTI ARMOR
      - ARMOR
        - BRIGADE (1)
        - BATTALION (5)
          - Battalion-M1A1 TANK w/o BN HQs Staff
          - BTB UA Heavy BCT
          - MECH-ARMOR MNVR BN-UA HVY BCT
          - T80 TANK BN-COMPLETE-OPFOR
          - T80B BN With MNVR COs Only
        - COMPANY (10)

Designation: Designation

Force: BLUFOR SIDE - FRIEND (e

Figure 7. MSDE's Unit library.


MSDE - Specify Tasking

Unit Tasking - Attack on Axis Chuck

Specify Tasking

Name	Attack on Axis Chuck
Trigger	Trigger1 - FRIENDLY_ACTIVITY - P00Y00M0(
Type of Tasking	TASK
Owning Unit	- COMBAT

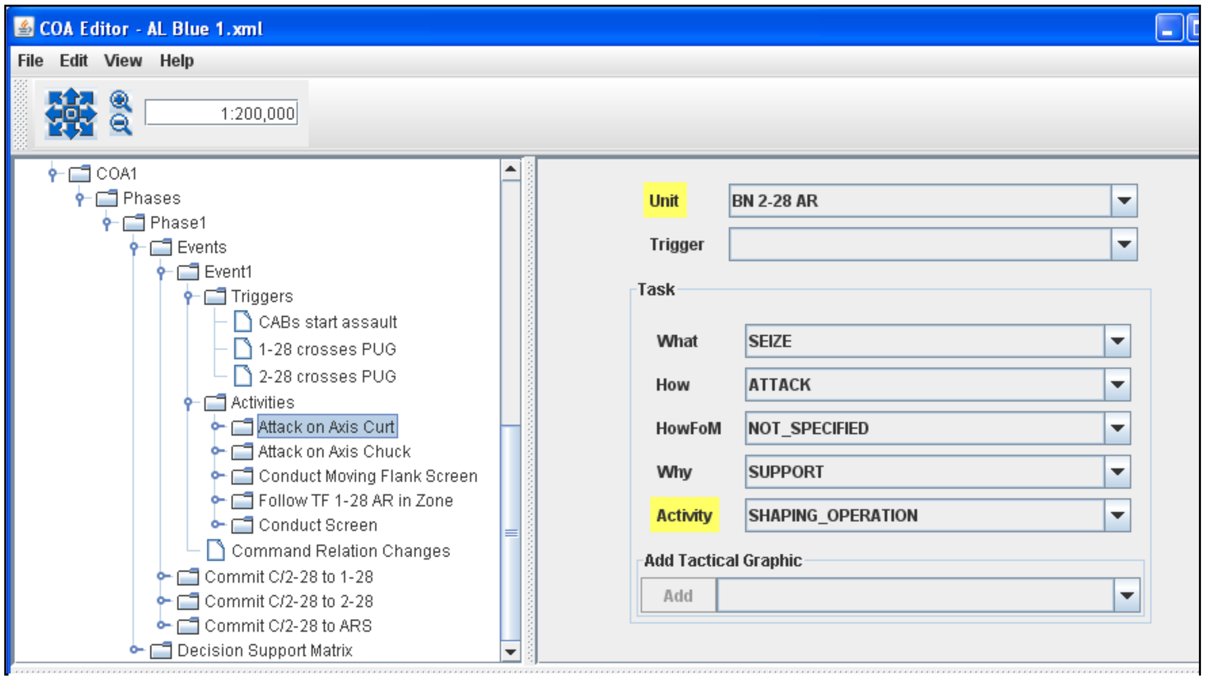
Task

Form of Operation [How]	ATTACK	FM 5-0.1
Task [What]	SEIZE	FM 5-0.1
Supported Unit		
Type or Form of Maneuver	NOT SPECIFIED	FM 3-0
Type of Tasking/COA	DECISIVE OPERATION	
Why Enumeration	CAUSE	

Tactical Graphics

Tactical Graphic	Type	Usage
OBJ Black	AREA	TASK_OBJECT_LOCATION
PL Pug	LINE	SYNCHRONIZE_ON
7 | 4	BOUNDARY	BOUNDARY
2-28 | 1-28	BOUNDARY	BOUNDARY
AXIS Chuck	LINE	MOVEMENT_CONTROL_MEASURE

Figure 9. MSDE's Task drop-down menu.


COA Editor - AL Blue 1.xml

- COA1
  - Phases
    - Phase1
      - Events
        - Event1
          - Triggers
            - CABs start assault
            - 1-28 crosses PUG
            - 2-28 crosses PUG
          - Activities
            - Attack on Axis Curt
            - Attack on Axis Chuck
            - Conduct Moving Flank Screen
            - Follow TF 1-28 AR in Zone
            - Conduct Screen
          - Command Relation Changes
        - Commit C/2-28 to 1-28
        - Commit C/2-28 to 2-28
        - Commit C/2-28 to ARS
      - Decision Support Matrix

Unit	BN 2-28 AR
Trigger	

Task

What	SEIZE
How	ATTACK
HowFoM	NOT_SPECIFIED
Why	SUPPORT
Activity	SHAPING_OPERATION

Add Tactical Graphic

Figure 8. Screenshot of the COA Editor tool.

In addition, the tool also included various hierarchical drop-down menus, useful for inputting the interconnected MSDL-DG COA elements, such as Tasks, Decision Points, and Triggers. See Figure 9.

COA Editor

The COA Editor was created out of the need for a more stable tool to view and edit MSDL-DG files. It employed a very simple user interface, consisting of a single window with drop-down menus for the MSDL-DG element values. See Figure 8. In contrast to

MSDE, it was not possible to construct a complete MSDL-DG file solely using the COA Editor; some elements were neither viewable nor able to be edited. Thus, it was necessary to use a standard .xml editor to modify any fields that were unavailable in the COA Editor.

VERIFICATION AND VALIDATION

After MSDL-DG files had been created through the use of MSDE and the COA Editor, there was an additional step to ensure that the files were free of critical errors and that they validated against the MSDL-DG schema. Any files containing errors would cause unpredictable results when input to the delicate Deep Green systems that were still in early development stages.

The MSDL-DG structure is such that elements reference other elements throughout the file. For example, a Task assigned to a particular Unit would reference the Unit's Universally Unique Identifier (UUID), as well as the UUIDs of any associated tactical graphics. Using a previous example (from Figure 3 and Figure 4) of the Screen Task assigned to 7-8 CAV, the MSDL-DG file would contain a Screen Task referencing the UUIDs for 7-8 CAV, OBJ SABER, OBJ SPUR, OBJ SCARF, and OBJ SADDLE.

Early on, we ran into problems when editing, deleting, or creating elements that were automatically given new UUIDs by our tools – all of the references to the "old" UUID were not automatically updated or deleted. Thus, the result was what we called "dangling UUIDs" – references to elements that no longer existed elsewhere in the file. In the example above, deletion of the OBJ SADDLE graphic would not automatically remove the UUID reference included in the Screen Task. Consequently, the Deep Green system would get stuck in an infinite loop attempting to locate the mysterious OBJ SADDLE graphic.

Because each MSDL-DG file could potentially contain hundreds or even thousands of UUIDs, it was unrealistic to keep track of all of the references manually. This led to the creation of additional tools that ran scripts for common, known errors in the files (such as "dangling UUIDs"). As we discovered new sources of error, we continuously updated our scripts in an effort to deliver high-quality test data files.

DISCUSSION AND FUTURE DEVELOPMENT

MSDL-DG provided a solid infrastructure for representing COAs for a U.S. Army Heavy Brigade. Tasks can be represented adequately with the added XML elements. However, in the future, a more complete set of tasks—including tasks for other services and other nation forces—should be reviewed in actual COAs to identify business rules for the use of task attributes and graphics.

The COAs we encoded made considerable use of conditional tasking. MSDL-DG had the infrastructure to allow us to represent such COAs. However, this aspect of MSDL had not been developed far enough to establish clear rules for sequencing tasks, or semantics for how simulations should execute them. The Deep Green program selected a convention for uses of hierarchical, conditional tasks; but ideally, the MSDL standard should specify the use cases and usage rules so that complex COAs can be implemented in a standard way.

Encoding the conditional expressions in the COAs was the most difficult part of the encoding process, and the part that we feel still needs work. Using triggers and conditional tasks is essentially programming complex brigade behavior into a simulation, and is therefore subject to all the pitfalls of software development. Unfortunately, in the first phase of Deep Green, the combat simulation was not sufficiently mature to execute all of the brigade COAs—especially not ones with complex conditional tasks. Thus, we do not know if the conventions and encoding approaches we took would have been completely successful.

Battle Management Language (BML)

Since it is not clear how the current and future standard MSDL will include unit plans in its specification of scenarios, we briefly investigated various BML efforts to gain insight into the progress being made elsewhere in representing COAs (Schade and Hieb 2006; Tolk, Diallo et al. 2007; Blais, Chartrand et al. 2010). While there has been development on how to combine standard databases of terms, doctrine, information sharing mechanisms, grammars, etc., there have not been many reports of applications to a significant variety of real COAs. In some cases, C-BML seems to be focused on real time command and control, and in these cases it does not have to address the issues of conditional tasking.

We did find examples of using C-BML to represent general military plans; these efforts also report challenges with representing conditional tasks, dynamic task organizations, etc. (de Reus, de Krom et al. 2008). If C-BML will be applied to COA representation, it will have to address the same challenges we faced with

MSDL-DG, including specifying rules for combining task attributes and tactical graphics, representing task hierarchies, the semantics of conditional execution, and the richness of predicate functions.

ACKNOWLEDGEMENTS

The Deep Green program was a collaborative research effort between Applied Research Associates and the two development teams – BAE Systems and SAIC. MSDL-DG was the result of hard work put in by many individuals from each of these teams, particularly James McDonough (BAE), Keith Stubbs (SAIC), Theresa Tamash (Dignitas), and Lee Lacy (DRC). Also, much time and patience was exerted by the subject matter experts responsible for the generation of more than 20 COAs, and the encoding tools developers Jeff Abbott (CAE), Gene McCulley (Stackframe), and Stephanie Graffuis (Stackframe).

REFERENCES

Applied Research Associates, (2009). Specification for: Military Scenario Definition Language (MSDL) for Deep Green (MSDL DG). *Deliverable B001 Exercise Driver Plan for BAA 08-09: Deep Green- Test and Evaluation.*

Blais, C., S. Chartrand, et al. (2010). Coalition Battle Management Language (C-BML) Phase 1 Information Exchange Content and Structure Specification. *Proceedings of the Spring 2010 Simulation Interoperability Workshop*.

de Reus, N., P. de Krom, et al. (2008). BML--Enabling National C2 Systems for Coupling to Simulation. *Proceedings of the Spring 2008 Simulation Interoperability Workshop*.

Defense Advanced Research Projects Agency (DARPA), Information Processing Technology Office (IPTO), (2007). BAA 08-09 Deep Green Broad Agency Announcement (BAA) for Information Processing Technology Office (IPTO). Retrieved June 28, 2010 from http://www.darpa.mil/ipto/solicit/baa/BAA-08-09_PIP.pdf.

Lacy, Lee W., Theresa Tamash, & Doug Reece (2010). Extending the Military Scenario Description Language (MSDL) to Represent Deep Green Course of Action Descriptions. *Proceedings of the Spring 2010 Simulation Interoperability Workshop*.

Schade, U. and M. R. Hieb (2006). Formalizing Battle Management Language: A Grammar for Specifying Orders. *Proceedings of the Spring 2006 Simulation Interoperability Workshop*.

Simulation Interoperability Standards Organization, (2008). Standard for: Military Scenario Definition Language (MSDL), SISO-STD-007-2008. Retrieved June 28, 2010 from http://www.sisostds.org/index.php?tg=articles&idx=More&article=40&topics=18

Tolk, A., S. Diallo, et al. (2007). A System View of C-BML. *Proceedings of the Fall 2007 Simulation Interoperability Workshop.*