Faculty and Researchers   Faculty and Researchers' Publications

2011

Coalition Battle Management Language (C-BML) Phase I Standard: Trial Use Findings and Next Steps

Blais, Curtis; Alstad, Anders

https://hdl.handle.net/10945/30788

Coalition Battle Management Language (C-BML) Phase I Standard: Trial Use Findings and Next Steps

Curtis L. Blais
MOVES Institute
Naval Postgraduate School
700 Dyer Road Rm269
Monterey, CA 93943
831-656-3215
clblais@nps.edu

Anders Alstad
Norwegian Defense Research Establishment (FFI)
P.O. Box 25, NO-2027
Kjeller, Norway
Tuluksak, AK 99551
(+47) 63 80 74 59
anders.alstad@ffi.no

ABSTRACT: *The Coalition Battle Management Language (C-BML) is a common language for expressing and exchanging plans, orders, requests, and reports across command and control systems, modeling and simulation systems, and robotic systems. The Simulation Interoperability Standards Organization (SISO) has approved a Product Development Group (PDG) to generate a specification and guidelines document for the C-BML standard through a three-phase effort. Phase 1 specifies the underlying data model for the standard and provides preliminary building blocks for generation of C-BML expressions together with numerous examples of application of the standard. Phase 2 will specify a formal grammar governing generation of valid C-BML expressions. Phase 3 will specify a formal semantic model for C-BML.*

This paper identifies several of the organizations participating in a trial use of draft Phase 1 products, describes their intentions for trial use of the draft products, and provides some initial findings from trial use of the products. Findings from the trial use will inform finalization of the draft Phase 1 specifications in preparation for the start of a comment round on the product.

1. Introduction

The Coalition Battle Management Language (C-BML) is a common language for expressing and exchanging plans, orders, requests, and reports across command and control (C2) systems, modeling and simulation (M&S) systems, and robotic systems. The Simulation Interoperability Standards Organization (SISO) has approved a Product Development Group (PDG) to generate a specification and guidelines document for the C-BML standard through a three-phase effort:

- Phase 1 specifies the underlying data model for the standard and provides preliminary building blocks for generation of C-BML expressions together with numerous examples of application of the standard.
- Phase 2 will specify a formal grammar governing generation of valid C-BML expressions.
- Phase 3 will specify a formal semantic model for C-BML.

1.1 C-BML Trial Use

In late January 2011, the C-BML Drafting Group (DG) provided to the PDG draft Extensible Markup Language (XML) schemas, guidelines document, and examples for use by the community in a Trial Use period. The period was originally scheduled to run through July 2011, but several organizations who indicated they are participating in the trial use were not able to complete their efforts and provide feedback in this timeframe. At the time of this writing, the trial use

period remains open for an unspecified (to be decided by the C-BML PDG) period. Refer to [1] for a description of Phase 1 draft products.

Trial use participants have been asked to provide feedback to the C-BML PDG in the form of comments on usability of the products as well as information regarding what portion of the product (in particular, the subset of constructs from the XML schemas) was evaluated during the trial use. Findings will be used by the C-BML Phase 1 DG to finalize draft products, including the Phase 1 Specification, in preparation for a comment round by the community.

This paper identifies several of the organizations participating in a trial use of draft Phase 1 XML products, describes their intentions for trial use of the draft products, and provides some initial findings from Trial Use of the Phase 1 XML schemas. Findings from the Trial Use will inform finalization of the draft Phase 1 specifications in preparation for the start of a comment round on the product.

2. C-BML Phase 1 Trial Use Participants and Initial Findings

The following subsections identify each of the C-BML Phase 1 trial use participants and their respective projects and trial use goals. Initial findings reported to date are included in the discussion.

2.1 Canada: Computer Generated Forces

Objective: Canadian researchers are in the process of building a new prototype system linking a computer-generated force (CGF) to one or several C2 systems. The decision on which portion of the C-BML schema will be used is still pending. A first (limited) working version of the prototype should be ready by the end September to middle of October, 2011.

POC: Fawzi Hassaine, fawzi.hassaine@drdc-rddc.gc.ca; Kevin Heffner, k.heffner@pegasim.com

2.2 France: APLET

Objective: French researchers plan to implement the C-BML Phase 1 Trial Use schema in the French APLET simulation, focusing on the "light" portion of the schema. They expect to perform tests at the end of July and provide feedback sometime in August, 2011.

POC: Bruno Gautreau, bruno.gautreau@cassidian.com

2.3 Germany: C-BML Evaluation and Graphical User Interface

Objective: FKIE will conduct the trial use, focusing on the "light" portion of the draft XML schemas. They plan to adjust their previously developed graphical user interface (GUI) to work with this portion of the schema..

POC: Ulrich Schade, ulrich.schade@fkie.fraunhofer.de

2.4 Norway: C-BML Evaluation

Objective: Norwegian researchers designed and implemented a proof-of-concept C-BML framework. This framework has capabilities to parse, validate, and persist C-BML expressions. The framework has been used to gain experience with processing C-BML expressions and to evaluate the complexity and ease of use of the C-BML Phase 1 products. Initial findings from the work are provided below.

Overall evaluation:
C-BML expression instances (e.g., the Guidelines examples) appear relatively easy to read and understand for humans. C-BML appears as a view of the Joint Consultation Command Control Information Exchange Data Model (JC3IEDM) that effectively restricts C-BML producers to create expressions that map to valid and complete JC3IEDM transactions (i.e., a received C-BML expression persisted to a JC3IEDM-database with the provided mappings appears to result in a valid JC3IEDM database).

Experience from implementing the proof-of-concept C-BML framework shows that processing received C-BML expressions is relatively complex. This complexity appears to be rooted foremost in the combination of nested XML elements and cross-referencing of XML elements (to elements both within the current C-BML expression and within earlier expressions). In the proof-of-concept C-BML framework, this complexity has been solved by going through multiple processing phases. One complex (but unlikely) challenge the framework was not able to solve was circular references among C-BML instances in one expression.

While the processing of received C-BML expressions appears complex but doable (with some JC3IEDM-knowledge), it appears even more complex to produce C-BML expressions. The most complex part of C-BML expression production relates to the nesting and referencing of C-BML elements/instances.

There are many pitfalls in C-BML expression production because there will be many complicated rules to evaluate. For example: Was the element being referenced previously created by my system or by others? If no, what other elements will the new element reference? Have referenced elements been sent before? Is the element defined inside the current expression? These challenges will apply both if an earlier persisted C-BML expression is reconstructed/loaded from a JC3IEDM database and if a completely new C-BML expression is made.

Through the design and documentation, it appears the C-BML Phase 1 designers have made several assumptions regarding how the XML Schemas should be used. Some of these assumptions are documented in the Guidelines document while others are undocumented or implicit. It is good that Phase 1 designers have left some flexibility to information exchange model (IEM) implementers, but experience from implementing a proof-of-concept C-BML server shows that more clarification is needed.

Questions and comments on C-BML exchange rules:

a) Rule 001: *The exchange of C-BML expressions shall not change, modify or otherwise alter the content and structure of said expressions.. Any additional elements required to exchange C-BML expressions that are specific to a given implementation and/or a given IEM are not considered to be normative and therefore should be able to be removed and/or ignored when processed by other systems and/or disseminated using another IEM.*

   i) Assume a RESTful[1] IEM where producers and consumers see C-BML expressions as resources that can be written and read to, for example, a central server. Rule 1 may be broken if a producer sends, for example, a "WhoTypeType" report to the server defining a "Who" and the server, immediately or in the future, forwards the report to consumers with a "WhoRef" (containing a URL[2] to the ObjectItem-resource) instead. We need to see if this is a correct interpretation.

   ii) Assume that an IEM persists received C-BML expressions and allows them to be accessed through remote requests. Is the

IEM then required by Rule 1 to reconstruct the persisted C-BML expressions exactly as they were received? If yes, this will only be possible if the expression is persisted in plain text (which only allows schema validations, not content validation). If C-BML expressions are persisted to a JC3IEDM-compliant/equivalent database, it is virtually impossible to reconstruct C-BML expressions to the exact structure they had when received. For example, if an expression contains one instance and multiple references to that instance, it is impossible to know which database relation should be reconstructed as a C-BML instance or a C-BML reference (if any).

b) Rule 002: *C-BML expressions shall be independent from the IEM or the architecture in which they are used. For example, HLA (High Level Architecture) Time Management or Data Distribution Management data elements should not be included as part of the C-BML expression since the elements are not present in all IEMs and cannot be generalized to all architectures.*

   i) Does this mean that all C-BML expression instances must make sense over any IEM and C-BML architecture? Should expression instances be made so that they can be published on multiple (or linked) IEMs? If yes, how should the <***Ref> elements be interpreted on IEMs based on, say, Simple Mail Transfer Protocol (SMTP) or HLA? Will it then be valid to provide (REST) URLs in ***Ref-elements?

   ii) Assume a C-BML consumer that utilizes a SMTP-based IEM. Will it be required that the C-BML ***Ref-elements refer to instances only defined inside the same expression instance? It is possible (for all kinds of IEMs) to rely on all expressions being received in the sequence they were sent and thus be able to reference instances sent in earlier expressions? If so, this would be a very fragile setup. For example, if an application consumes expressions based on a subscription, the subscription might filter out expressions that define instances that are references in later expressions. Another problem is with expressions published before a

[1] REST: Representational State Transfer
[2] URL: Uniform Resource Locator

federate starts to subscribe. It should therefore be possible to request required instances at any time. RESTful services are well suited to this, since they produce URLs that point to specific instances (which is suitable for use in ***Ref-elements). If the Simple Object Access Protocol (SOAP) is used, the ***Ref-elements will (probably) contain the same reference(s) as the corresponding OID-elements[3] and thus require a SOAP application program interface (API) for accessing the referred instance. As (among other things) the content of ***Ref-elements will depend on a specific IEM, how can users comply with Rule 2?

Other questions and comments:

1. Is the ObjectType in WhyTypeType added for completeness (allowing the expression producer to choose between WhyTypeType.ObjectType or SupportingType) or does it have a different meaning than SupportingType?

2. OID is optional for some elements and required for others. OIDs are required for some elements in order to make it possible to reference them inside a C-BML expression. Why not make it required to provide valid OIDs for all <OID>-elements (e.g., Context OID)? Is there any benefit to allow some OIDs to be optional?

3. Can two different C-BML expressions (e.g., report or order) define the same instance? For example, is it allowable to duplicate the definition of an ObjectItem in two different reports, or must the second report use a reference to the ObjectItem? Is this IEM-specific? If yes, does it break expression exchange rule (Rule 002 above)?

4. Is it invalid to define the same instance twice or more inside the same C-BML expression? Is this IEM-specific?

5. Should "current state" OIGs be linked to its containing order OIG through CONTEXT-ASSOCIATION instances (we could not find mappings for this)?

---

[3] OID: Object Identifier

6. The role of the "Context" (OIG) element should be described in more detail. It appeared that Context-elements could be created, for example at the beginning of a scenario run, and that a C-BML expression producer could select which Context to associate with its reports and orders (and thus make creation of common operational pictures and subscription topics easy to implement). As now understood, the Context-element in C-BML expressions are used only to link together the different parts within each C-BML expression (i.e., only a way to persist an expression).

7. It is a bit confusing to have Report-elements within a containing Report-element. Not a big issue, but should at least be clarified, or rename the outer Report-element to something ReportContainer (since that is what it is).

8. An IEM that can accept Military Scenario Definition Language (MSDL) documents should:
   a. Allow clients to access MSDL documents on request;
   b. When a C-BML server (i.e., an IEM) receives an MSDL document, it should generate a set of C-BML reports that reflects, for example, the units, types, and organisation hierarchy defined in the MSDL document. All these reports should be combined into a single C-BML report expression.

9. When implementing an IEM with logical validation of C-BML expressions, it appears to be necessary to store all C-BML expressions exchanged in a scenario run in a single database (because of references to elements in other expressions). This appears to be the only way to ensure that all database relations are valid (and thus that all C-BML references are valid). If this assumption is correct, it restricts IEM developers to either using a star topology (with a central server storing all C-BML expressions) or forcing all C-BML consumers to receive and store all C-BML expressions (i.e., not allowing consumer-specific subscriptions). This is not really an issue, but should be clarified or described more fully.

10. The process of storing (in a JC3IEDM database) a received C-BML expression does not result in any update or delete operations,

only insert and read operations. This gives two benefits:

  a. A C-BML server should be fast because it should never need to wait for concurrent access locks.
  b. The C-BML full schema is well-suited for RESTful web services because instances (e.g., reports, ObjectItems, Locations) are never updated or deleted, and therefore can be provided as web resources that can be long-term cached by standard web caches/proxies and thus be very fast and effective to access.

11. Late-joining federates are an important issue that is not described in the Guidelines Document. This is an important issue that must be addressed! Resending all already-exchanged C-BML expressions to a federate when it joins the federation will work if it joins shortly after a scenario is started. When a scenario has run for some time and/or many C-BML expressions hav been exchanged this is no longer a good solution. A C-BML server with an API for accessing C-BML complex elements like ObjectItem and Report alleviates some of these issues, but does not provide a easy and effective way of extracting the current common operational picture (COP). Any suggestions? We assume this is difficult because one C-BML server or JC3IEDM database potentially can store several COPs.

Proof-of-concept C-BML framework:
This section describes the Norwegian C-BML framework and early experiences with it. Currently, support for the following types of C-BML reports is implemented in the framework: WhoTypeType, WhoStatusType, WhoHoldingType and WhoLocationType (this includes support for the ObjectItem, ObjectType, ObjectStatus and Location hierarchies). Order and request C-BML expressions have not been implemented, but should be supported by the framework.

C-BML framework approach:
The framework is implemented in Java. Only support for parsing, validating, and persisting C-BML expressions is implemented. The framework is packaged as a library that potentially can be utilized inside different kinds of IEMs. Currently there are no plans to implement a full IEM with this framework.

Upon receiving a C-BML expression, the framework goes through the following phases:

1. *Conversion from XML to object instances*: Performed by the JAXB framework that comes with Java.
2. *Search for C-BML instances with a OID:* Creates a map linking OIDs and C-BML instances. Performed by code that crawls through all parts of the received C-BML expression. Evaluates that all OIDs are valid integers. A valid OID is generated for the instances that have OID=0.
3. *Convert/map all C-BML instances from phase 2 to JC3IEDM-equivalent object instances:* This phase converts all C-BML instances that map directly to JC3IEDM. When the mapping logic finds OID-references in C-BML instances, it first checks for the referenced instances in the DB and, if not found there, checks the map created in phase 2 above. If a C-BML instance is found in the map, it immediately is converted to a JC3IEDM-equivalent object.
4. *Convert/map all reports/orders/requests inside an expression to JC3IEDM-equivalent objects:* This step is necessary in order to convert the C-BML elements/instances that do not map directly to a single JC3IEDM entity (hierarchy). This phase uses the JC3IEDM-equivalent objects from phase 3 above.
5. *Persist the JC3IEDM-equivalent Context-object to db4o (an object-oriented database):* The JC3IEDM Context-object becomes the root of an object-hierarchy that represents a complete C-BML expression. Db4o effectively persists the whole object hierarchy in a single operation.

Experiences from creating the framework:
We recognize there are many potential approaches to processing C-BML expressions. The C-BML expression examples provided in the Guidelines document appear reasonably easy to interpret for humans, but it was necessary to perform multiple design iterations before being able to construct a framework that could support parsing, validation, and persistence of all kinds of C-BML expressions. The following experiences were obtained from creating the framework:

- Conversion from XML to Java objects was very easy. JAXB was able to generate all necessary Java code without any special configuration.
- To be able to effectively process a C-BML expression, all C-BML instances with an OID must be located before the actual JC3IEDM-mapping can start. As a result both phase 2 and

phase 3 steps described above must contain code that can traverse all parts of a C-BML expression. This makes the processing logic more complicated and duplicates some code. Another option could be to use XPath to search for OIDs in a C-BML expression when the mapping logic finds a reference. This would be much slower and probably also require a substantial amount of code.

- The conversion in phase 3 (e.g., ObjectItems and Locations) was mostly a direct mapping from C-BML entities to JC3IEDM-equivalent objects.
- The first implementation iterations for the C-BML framework used Hibernate (mapping to an SQL database) instead of db4o. Creating Java classes with mappings to JC3EIDM tables was found to be too complicated for this proof-of-concept and db4o was therefore chosen instead.

POC: Anders Alstad, anders.alstad@ffi.no

2.5 Sweden: C-BML Schema Evaluation

Objective: Swedish researchers are evaluating both the C-BML draft schemas, expecting to be completed by the end of July, 2011.

POC: Per Gustavsson,
per.m.gustavsson@saabgroup.com

2.6 United States: Scripted BML Server

Objective: The George Mason University (GMU) C4I Center maintains an open source BML server, available at https://netlab.gmu.edu/trac/OpenBML, known as the Scripted BML Server (SBMLServer). GMU previously posted a server with script supporting the CompositesLight portion of the draft C-BML schema, and has recently added a version of the server that supports the full Composites schema. For cases other than the CompositesLight portion, this implementation does not breakdown the XML input into JC3IEDM for storage. Instead, it stores the entire BML document as text, retrievable by a key element (OrderID at present) and with all the other capabilities of SBMLServer (publish/subscribe, RESTful operation, logging/replay, etc.). GMU is eager to have members of the C-BML community try the new capability and provide feedback.

POC: Mark Pullen, mpullen@netlab.gmu.edu

2.7 United States: Marine Air-Ground Task Force (MAGTF) Tactical Warfare Simulation (MTWS)

Objective: Evaluate the ability to construct expressions using the Phase 1 C-BML XML schema for commands used to drive the Marine Air-Ground Task Force (MAGTF) Tactical Warfare Simulation (MTWS) and, time permitting, for reports produced by MTWS during execution. The work is expected to provide an evaluation of the richness and coverage (or lack thereof) of the Phase 1 schema across several warfare areas found in MTWS (ground combat, air operations, fire support, combat engineering, combat service support, etc.). This work is in progress, to be completed by 31 December 2011.

In its general use today, MTWS execution is driven by interaction with a group of human operators and exercise controllers. Some work has been done to provide interactions with real-world command and control systems, but this has been implemented through a limited set of tactical messages, using established formats such as the United States Message Text Format (USMTF), Over-the-Horizon Gold (OTH-Gold), and Variable Message Format (VMF). The key assertion is this—if MTWS commands and reports were encoded in C-BML, then potentially, they can come from or be provided to any source, human or automated (of course, humans will typically not deal with C-BML expressions directly, but will see or input the content of plans, orders, requests, and reports through some graphical user interface).

A sample (less than 10%) of MTWS commands for planning, ordering, and reporting is provided below.

Orders
Air Mission Cancel
Air Mission Commit
Air Mission Define
Air Mission Divert
Air Mission Hold
Air Mission Land
Air Mission Launch
Air Mission Orbit
Air Mission Vector
Landing Ship Land
Wave Land
Wave Load
Fire Mission Cancel
Fire Mission Check Fire
Fire Mission Fire
Ground Engineering Construct
Ground Engineering Remove
Cargo Load
Cargo Unload
Ground Unit Dismount
Ground Unit/Task Force Face
Ground Unit/Task Force Formation

Ground Unit/Task Force Front
Ground Unit/Task Force Halt
Ground Unit/Task Force Mission
Ground Unit Mount
Ground Unit/Task Force Posture
Ground Unit/Task Force Resume
Maritime Sonar Activate
Maritime Sonar Deactivate
Maritime Surface Craft Move

Plans/Planning

AZ Hour Define
Air Mission Define
Beach Define
Fire Mission Define (e.g., on-call missions; also serves as a call for fire)
Serial Define
Wave Define

Reports (Solicited)

Air Mission
Air Track Data
Ship-to-Shore Assault Schedule
Ship-to-Shore Wave Status
Fire Support Fire Mission Status
Ground Planned Missions
Ground Unit Situation
Intelligence Ground Detection
Engineering CE Operations Status
Maritime Air Defense Status

There is another category of reports called "unsolicited" or "spot" reports that are generated by MTWS during execution and sent to selected exercise control consoles (according to assigned roles) to inform the controllers and operators of actions or outcomes that have occurred in the simulation (e.g., a moving unit arrives at a destination, battle damage assessment from a fire mission). These are also candidates for encoding in C-BML for transmission to other simulation or C2 systems.

It is interesting to note that MTWS also has a number of commands that are used to initialize the exercise data base. These are expected to fall within the application of the MSDL standard [2] rather than C-BML. There is a clear overlap in concerns that needs to be addressed through further discussion and coordination across the C-BML and MSDL PDGs. Some of the MTWS commands that need to be studied for possible expression in MSDL include: Aircraft Define; Airfield Define; Ground Structure Define; and Ground Unit Define.

There is also a class of commands in MTWS that could be considered under the category of simulation or exercise control, such as override commands to prevent a ground entity from being detected or manually causing a detonation in the exercise conduct. Such commands need further study to determine if there are use cases for having these be expressed in C-BML; for example, to allow a remote operator who is employing some other simulation or C2 system to initiate such actions in the MTWS execution.

Another aspect of this study is the recognition that orders occur at different levels of task organization (e.g., from a task force down to an individual unit or entity) and at different levels of specificity (e.g., a more general order to conduct some mission compared to a precise order to change state, as in changing the direction a unit is facing, its posture, or its formation). Such study crosses over into key concerns for the use of C-BML for robotic forces [3].

POC: Curtis Blais, clblais@nps.edu

2.8 United States: Coalition Battle Management Services (CBMS)

Objective: Evaluate expressiveness and computability of C-BML in various applications, including CBMS / VR-Forces, OneSAF, and JSAF.

POC: Saikou Diallo, sdiallo@odu.edu

3. Conclusions and Summary

Even from very preliminary findings from the trial use period, the Phase 1 C-BML draft products are proving to be useful across a broad set of applications. The broad exposure and usage, across several nations, should facilitate the process of review and approval by the SISO community when the full product draft becomes ready for comment. The C-BML PDG would like to thank all those who are taking part in the Phase 1 trial use. Findings from these activities will be invaluable in providing practical feedback to the PDG and DG for completion of Phase 1 drafting efforts.

4. References

[1] Blais, C.: "Coalition Battle Management Language (C-BML) Phase 1 Specification: Road to Balloting," Paper 11S-SIW-001, *Proceedings of the Spring Simulation Interoperability Workshop*, Simulation Interoperability Standards Organization, Boston MA, April 2011.
[2] Standard for: Military Scenario Definition Language, SISO-STD-007-2008, Simulation

Interoperability Standards Organization, 14 October 2008.

[3] Davis, D., C. Blais, and D. Brutzman: "Autonomous Vehicle Control Language for Simulated and Real Robotic Forces," Paper 06F-SIW-004, *Proceedings of the Fall Simulation Interoperability Workshop*, Simulation Interoperability Standards Organization, Orlando FL, September 2006.

Author Biographies

CURTIS BLAIS is an Associate Research Professor in the Naval Postgraduate School Modeling, Virtual Environments, and Simulation (MOVES). He has over 37 years of experience in M&S research, development, management, and education. His research interests include application of web-based standards to C2 and M&S system interoperability and exploration of human social culture behavior (HSCB) modeling. Mr. Blais holds Bachelor of Science and Master of Science degrees in Mathematics from the University of Notre Dame.