18TH ICCRTS

Eighteenth International Command and Control Research and Technology Symposium

C2 IN UNDERDEVELOPED, DEGRADED AND DENIED OPERATIONAL ENVIRONMENTS

June 19 - 21, 2013

Alexandria, Virginia USA

Topics: Modelling and Simulation
Architectures, Technologies and Tools
Networks and Networking

Title: An Approach Using MIP Products for the Development of the Coalition Battle Management Language Standard

Authors: **Kevin Heffner (POC)**
Pegasus Research and Technologies
PO Box 47552
CP Plateau Mont-Royal
Montreal (QC) H2H 2S8 Canada
+1-514-360-4920
k.heffner@peretec.com

Nico Bau
Michael Gerz
Fraunhofer FKIE
Fraunhofer Str. 20
53343 Wachtberg-Werthhoven
Germany
+49 228 9435 515
{michael.gerz|nico.bau}@fkie.fraunhofer.de

An Approach Using MIP Products for the Development of the Coalition Battle Management Language Standard

Nico Bau, Michael Gerz, Kevin Heffner

Abstract

The development of interoperability standards can facilitate communication among information systems by defining a common way to exchange information. These standards are in fact comprised of normative and informative products that typically specify the details and examples that enable heterogeneous systems produced by different organizations to be integrated successfully and then to interoperate, as per system requirements.

Identifying and managing such requirements is a key element to building successful standards – those that ultimately are adopted, utilized and meet stakeholder expectations. Applying systems engineering principles combined with a well-defined Enterprise Architecture process-driven approach already has allowed for the *Multilateral Interoperability Programme* (MIP) Working Group to produce the *MIP Information Model* (MIM) as a proposed successor to the *Joint Consultation Command and Control Information Exchange Data Model* (JC3IEDM). The *Coalition Battle Management Language* (C-BML), being developed by the Simulation Interoperability Standards Organization (SISO) has utilized the JC3IEDM as the underlying data model for its Phase 1 product development. The current paper reports on preliminary work that has been done using the proposed MIM and associated toolset as the basis for the C-BML Phase 2 product development.

1. Introduction

"*A standard is a document that establishes engineering and technical requirements for products, processes, procedures, practices and methods, and has either been decreed by authority or adopted by consensus.*" [1]. Standards development organizations produce products such as technical specifications and other supporting documentation for the purposes of guiding and/or constraining system development, integration and maintenance or other aspects of a system's life-cycle. These products are not the end-user system, but rather provide assurance that the end-user system will possess certain characteristics (i.e. functionality and quality factors) and thus meets stakeholder expectations. In fact, system *designers* or *developers* generally are the primary users of technical standards products. Therefore, the *system* users and the *standard* users form two distinct user groups. Ensuring that end-user/stakeholder requirements are consistent with the standards users' technical perspective can be challenging, especially when these two groups represent different organizations with potentially different underlying motivations.

In addition, the development of new technical standards often is influenced or even triggered by the availability of emerging technologies that offer potential benefits to stakeholders. Several authors have identified deficiencies in traditional systems engineering approaches regarding the proper management of changing requirements associated with emerging technologies and evolving operational requirements and stakeholder needs [2][3][4][5][6]. All of these authors prescribe the use of so-called *agile*, iterative system and software engineering processes that address many of these deficiencies. However, such methodologies are less frequently applied to *standards* development processes. Nonetheless, reference [3] describes the benefits of applying an agile systems engineering approach for the development of interoperability[1] standards in the transportation sector.

In the context of the MIP Block 4 Working Group, Lang et al [6] propose an enterprise architecture approach for developing the block 4 MIP Information Model (MIM), proposed successor to the JC3IEDM. This approach applies a Model Driven Architecture (MDA) methodology combined with the use of the NATO Architectural Framework (NAF) [7]. Heffner and Gupton [8] propose a Standards Development Framework (SDF) for the SISO Coalition Battle Management Language (C-BML) that is based on a similar approach to the one defined by the US Intelligence Community/DoD for a Keyword Query Language Specification [9]. Consistent with [6], the C-BML SDF approach also embodies the enterprise architecture and agile systems engineering methodologies.

---

[1]NATO definition: "The ability to act together coherently, effectively and efficiently to achieve Allied tactical, operational and strategic objectives."[7].

This paper describes recent efforts to implement the C-BML SDF using the MIM process and tools in order to establish a well-defined, well-documented, repeatable, manageable process and production chain for developing and maintaining SISO Phase 2 C-BML standard while leveraging the MIM types, process and tools.

Following the introduction, section 2 provides an overview of C-BML while section 3 describes the MIM. Section 4 outlines the general standards development approach while section 5 presents some preliminary results in applying this approach to the development of C-BML. Section 6 provides conclusions and describes some remaining challenges and areas of future work.

2. Coalition Battle Management Language (C-BML)

SISO currently is developing C-BML, a standardized formal language for the exchange of digitized military information among command and control, simulation and autonomous systems. C-BML is an interoperability standard that can greatly facilitate the preparation and execution of military scenarios in support of military enterprise activities such as: *Training*; *Support to Operations*; and *Concept Development and Experimentation*. Preliminary research using C-BML already has shown the benefits that include:

1) reduced exercise/experiment/planning preparation time;

2) increased realism of the training/experimentation environment;

3) reduced cost associated with the decrease in the number of required simulator operators [12][24].

The following sections describe the C-BML standard in terms of language components and the corresponding standard specifications.

2.1. Practical Definition of C-BML

C-BML is intended to be an unambiguous, formal, language for communicating military information for machine-to-machine communication. In general terms, a *grammar* is a set of rules that dictate what valid sentences or expressions can be constructed for a given language.

Initiated in 2006 with the formation of the C-BML Product Development Group (PDG), SISO's development of C-BML has proven to be a difficult task, as witnessed by the time it has taken to produce an initial balloted Phase 1 specification [11]. As early as 1999, Argo et al. already proposed a Battle Management Language (BML) suggested that the BML expressions be based on a structure that included 5Ws to facilitate the programming of simulated/automated units: Who What Where When Why [2]. The 5Ws can be described as follows:

- **Who**: The tasking unit; The tasked unit; The supported unit; The supporting unit; The target; The reporting unit; The object of a report.
- **What**: The type of operation or task to be executed; The event being observed.
- **Where**: Where is the task to be executed; Where is the event being observed.
- **When**: The time the task to be executed or has been executed; the time an event observed.
- **Why**: The purpose, motivation, desired effect or result.

C-BML has followed these basic definitions. A graphical example of a simple C-BML task is shown in Figure 1, (the Why has not been included for clarity).

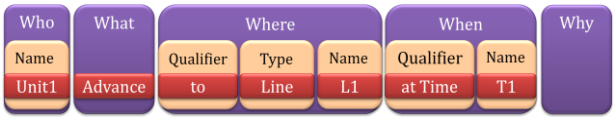

Figure 1 – Graphical C-BML Example illustrating 5Ws

In practice, C-BML expressions will be communicated using one of several concrete syntaxes such as the eXtensible Markup Language (XML) or Java Serialized Object Notation (JSON). An example of a simplified XML expression for an Air Interdiction task is shown in Figure 2.

2.2. C-BML Product Development Plan

C-BML is of the family of Battle Management Languages (BML) and like other languages is comprised of: *vocabulary*; *grammar*; and *semantics*. The vocabulary and grammar are required to construct valid, syntactically correct expressions representing military information. However, additional information, such as doctrine, is required to correctly interpret the intended *meaning* of this information, which may differ across services, nations or depending on the nature of the operation. In addition to the vocabulary and grammar components of the C-BML standard, the SISO C-BML PDG also has defined the need for a C-BML ontology to capture such additional information.

```
<Order>
    <AirTask>
        <TaskeeWho>
            <UnitID>CA-UAV</UnitID>
        </TaskeeWho>
        <What>
            <ActionCode>AI</ActionCode>
        </What>
        <Where>
            <WhereID>14010000784100000427</WhereID>
            <WhereLocation>
                <GDC>
                    <Lat>40.062195</Lat>
                    <Lon>47.57694</Lon>
                    <ElevAGL>3000.0</ElevAGL>
                </GDC>
            </WhereLocation>
        </Where>
        <StartWhenTime>
            <StartTimeQualifier>AT</StartTimeQualifier>
            <DateTime>20091022141229.359</DateTime>
        </StartWhenTime>
        <AffectedWho>
            <UnitID>OMF195-B12</UnitID>
        </AffectedWho>
        <TaskID>1409999900000000019</TaskID>
    </AirTask>
    <OrderIssuedTime>20091022141443.000</OrderIssuedTime>
    <OrderID>1409999900000000030</OrderID>
    <TaskerWho>
        <UnitID> 1-HBCT</UnitID>
    </TaskerWho>
</Order>
```

Figure 2: Simplified XML C-BML Example

In 2006, the SISO C-BML Study Group produced a report [10] with the following recommendation:

> *"[…] For all versions, the Study Group recommends using the [Command and Control Information Exchange Data Model] C2IEDM and its successor (Joint Consultation Command and Control Information Exchange Data Model – JC3IEDM) as a basis for C-BML reference implementations and standards. […]"*

Reference [10] further recommends that the first version of C-BML be described as a data model (i.e. base vocabulary) defined in XML as a subset of the C2IEDM. However it was recognized that there might be a need for extensions to meet requirements from the Modeling and Simulation (M&S) community. It also was recommended that the second version of C-BML introduce the C-BML grammar, while the third version addresses the need for ontology-based solutions.

Therefore, the SISO C-BML Product Development Group has established a three phase plan for developing C-BML as follows:

Phase 1: Establish a *vocabulary* or basic lexicon composed primarily terminal symbols;

Phase 2: Define a *grammar* or set of production rules that indicates how to combine the vocabulary to form valid expressions; and

Phase 3: Introduce an o*ntology* or set of relationships that can facilitate the interpretation of C-BML expressions.

In reality, the plan calls for overlap of the phases, as shown in Figure 3 wherein phase 1 also includes preliminary grammar, and phase 2 includes preliminary ontology work.

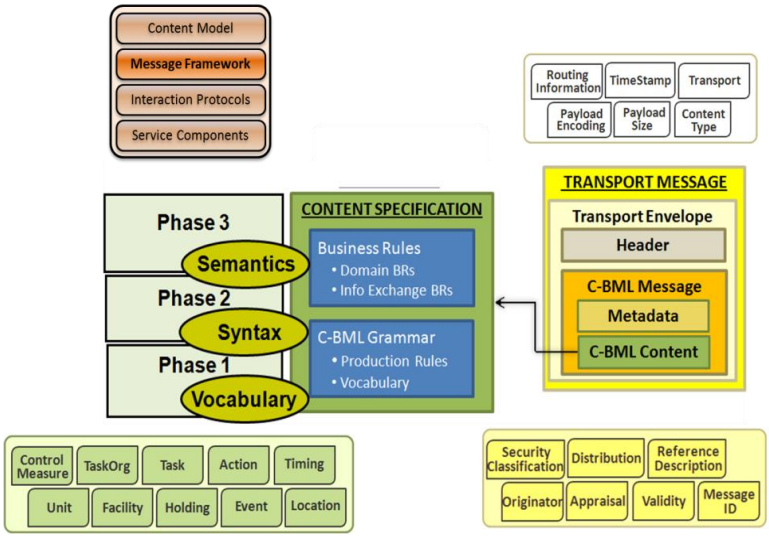

Figure 3– SISO C-BML Overview

The C-BML Phase 1 development activity recently has been completed, resulting in a balloted standard. The C-BML Phase 1 product is consistent with the recommendation to use the JC3IEDM [13] as the underlying data model to define the C-BML vocabulary. The C-BML Phase 2 development activity seeks to build upon the vocabulary defined in Phase 1 and complement this with formal grammar definition and basic ontology.

Figure 3 also illustrates additional elements of the Message Framework proposed as part of the C-BML SDF [8], such as the C-BML message structure and the distinction between production rules (i.e. grammar) and business rules (i.e. domain-specific or additional logic that is not specified as part of the grammar).

The following sections provide some basic definitions of language constructs required for the remainder of the paper.

2.2.1. C-BML Vocabulary and Grammar Considerations

A *formal* grammar is a set of *mathematical* rules that can be used by lexers and parsers for processing language expressions. In general terms, a language L *can* be generated from a formal grammar G, although, strictly speaking, this is not always the case:

$$L(G) \quad (1)$$

A grammar can be defined by a set of production rules P that operate on a set Σ of elements referred to as symbols. Σ is comprised of two sets of symbols: the set of non-terminal symbols N, and the set of terminal symbols T. Terminal symbols are elementary symbols that cannot be broken down further and for the intents and purposes of C-BML they can be considered to form the C-BML vocabulary and may include *keywords*, *identifiers*, *codes* and *values of core data types*. Non-terminal symbols are clauses, phrases and expressions of which a subset is the so-called set of *start symbols*, σ. Non-terminal symbols are used, for example, to represent entities such as *units*, *control-features* or properties such as *temporal-validity* and *location*. Start symbols indicate the roots of valid complete expressions or sentences (e.g. *report*, *order*, *request*, *acknowledgement*, etc.). Hence, formal grammars can be expressed as quadruples:

$$G = (T, N, \sigma, P) \quad (2)$$

Formal grammars can be represented as trees, or more specifically, Abstract Syntax Trees (AST), where the leaves are terminal symbols and branches are non-terminal symbols. In order to process formal language expressions using software components, AST are transformed into Concrete Syntax Trees (CST) that also are known as *parse trees* used by parsers.

Examples of BML grammars are the Command and Control Lexical Grammar (C2LG) [14] and the Operations Intent and Effects Grammar (OIEG) [15]. Both of these grammars borrow from Lexical Function Grammar (LFG) framework that has the benefit of being well-adapted for analyzing and generating natural languages. However the usefulness and applicability of a LFG approach for specifying C-BML remains to be seen since many users have expressed the desire for a "simple" grammar that, if necessary, references an ontology that provides information required for interpretation. It can be argued that it is not for the language to impose too many restrictions on what constitutes a *meaningful* expression, but rather only to specify what constitutes a syntactically and structurally complete and correct expression. In other words, the semantics generally should not be enforced by the grammar, but rather specified by an ontology.

2.2.2. C-BML Ontology

As described above, a formal language can be defined by a grammar as the set of valid expressions or sentences that are syntactically correct. But in order to interpret these expressions, additional semantics may be required. In some cases an ontology may not be needed by C-BML consuming applications. However, for applications that utilize inference or reasoning engines, additional information may be required to properly process C-BML encoded information. Ontological means can be used to relate elements of formal language expressions and state facts and assertions that are difficult or verbose to express using traditional formal grammars.

Hence, the C-BML ontology complements the grammar by adding additional relationships and constraints among *data elements*. Ontologies also allow for specifying information about *data instances* as well. Hence, ontologies may be used during application development to ensure the proper utilization of the C-BML language by applications or may be used during application run-time to construct a knowledge repository to store and derive new information.

The C-BML ontology defines a set of *universal* relationships or semantics that are common to all C-BML producer and consumer applications (e.g. a taxonomy of control features). However, it is unlikely that one ontology will meet all of the service-specific or community-specific needs and therefore ontology extensions will be required. Hence,

the C-BML ontology could be included in the standard as a *reference* ontology based on NATO doctrine and procedures, while allowing for specific communities of interest to extend the ontology to meet their needs.

The current approach calls for the use of The Unified Modeling Language (UML) from the Object Management Group (OMG) as the central modeling language. Therefore it is of interest to consider how one may represent ontologies using UML. UML can be used to represent conceptual models, sometimes referred to as Platform Independent Models (PIM) in the Model-Driven Architecture (MDA) terminology. But, UML alone does not lend itself to specifying model constraints and for this reason the OMG has developed the complementary Object Constraint Language (OCL) that provides a formal expression of rules such as invariants. Although ontologies also can be considered as conceptual models, UML and OCL are not well-suited to specify ontologies since many of the ontology constructs are lacking. The Web Ontology Language (OWL) has been developed for this purpose and is better suited to represent certain aspects of conceptual model, in particular the specification of restrictions. In fact, the OMG has recognized the usefulness of ontologies and of the OWL specification and has created a UML profile called the Ontology Definition Metamodel (ODM) that allows for the representation of an OWL ontology in UML. Within the context of the current approach, the intent is to produce an ontology in the form of a set of OWL modules that are generated from an UML ODM ontology constructed using the process and tools outlined in this paper. The requirements for the C-BML ontology are still being collected and consequently, this work is still of an exploratory nature.

2.3. C-BML Development Process and Tools – From Phase 1 to Phase 2

The C-BML Phase 1 development activity did not employ a formal process and dedicated tools for elaborating the main product artifact, the C-BML schema, illustrated in Figure 4. The schema was handcrafted directly using XML editor tools and therefore although an implicit model can be associated with an XML schema; no corresponding logical data model or conceptual model was constructed as the basis for the schema. This has been the source of many difficulties, perhaps the most important of which is the inherent difficulty in applying changes to the existing C-BML Phase 1 product. This has made it very difficult to maintain or evolve the Phase 1 products. Also, no formal requirements have been gathered or managed. Thus many questions subsist: *What requirements have been satisfied by specific schema elements? What were the reasons behind a specific modeling strategy? What changes need to be applied in order to maintain consistency with the underlying JC3IEDM foundation?* Lessons learned from C-BML Phase 1 drafting activity have been inputs into the C-BML SDF that highlights the need for a Logical Data Model representation and the ability to generate more than one concrete syntax or physical model. The agility that results from this approach is consistent with the MIM process and tools and therefore presented an excellent basis for the C-BML Phase 2 drafting activity.

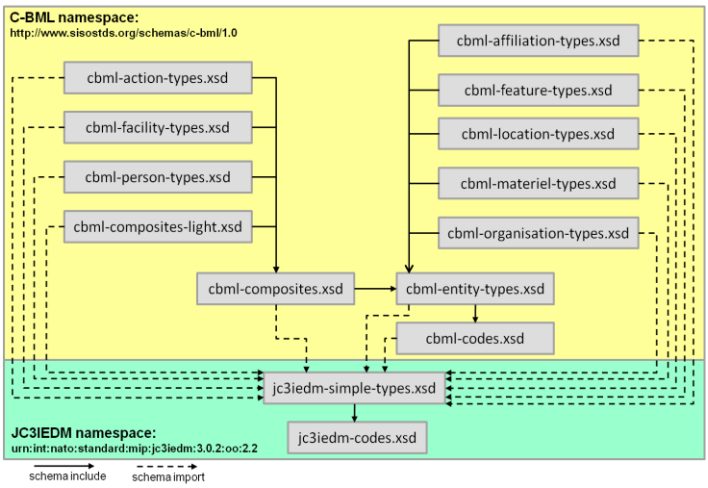

Figure 4- SISO C-BML Phase 1 Schema Structure

2.4. Status of C-BML Phase 1

Figure 4 illustrates the re-use of the JC3IEDM codes and simple types (shown in the green layer) represented using dashed lines. In this figure, C-BML elements are represented as: codes, entity-types, complex-types (e.g. action-types, facility-types, person-types etc.), and *composites*. The composites include definitions for elements that represent the 5Ws, discussed in the previous sections. Following a successful balloting process in September 2012, the C-BML Phase 1 product is in the instances of becoming an official standard, pending final ballot comment resolution.

The C-BML Phase 2 Development Activity already has been initiated and has identified several areas that need to be addressed, including: 1) establishing a set of stakeholder requirements; 2) defining a normalized, logical data model (i.e. PIM); and 3) creating a mechanism for the automatic generation of physical model or Platform-Specific-Model

(PSM), including XML Schema Description (XSD) documents and possibly preliminary OWL ontology modules. This paper describes how the use of the MIP Block 4 model and tools will help to achieve the Phase 2 objectives in the form of a well-defined, well-documented, sustainable process and tool chain.

3. MIP Information Model

The Multilateral Interoperability Programme (MIP) is a joint effort of 29 nations and NATO to support interoperability of command and control systems. Its standardization endeavors cover technical as well as procedural and operational aspects of the information exchange. The current MIP specification, the MIP baseline 3.1, is based on the Joint Consultation, Command and Control Information Exchange Data Model (JC3IEDM), which has been ratified under NATO STANAG 5525. For more than two years, MIP has been working on a successor to the well established JC3IEDM that combines the rich operational content of the JC3IEDM with state of the art technologies. This new model, called MIP Information Model (MIM) breaks with several design constraints of the JC3IEDM while at the same time maintaining all the operational concepts. Thus, the MIM has operationally the same expressiveness as the JC3IEDM. The first and most obvious difference between the JC3IEDM and the MIM is the choice of modelling language. While the JC3IEDM is described as an entity-relationship model using the IDEF1X notation, the MIM is described as a class model in the UML.

This difference has several implications:

- **Platform Independence:** Since the JC3IEDM was modelled in a way that it directly maps to a database schema that can be used with the Data Exchange Mechanism of MIP, the JC3IEDM can be seen as a PSM. This makes it more difficult to create other representations of the JC3IEDM such as XML schemata or ontologies, even though these mappings have been done in the past. The MIM, in contrast, has been designed from the beginning to support the approach of MDA and as such can be considered a PIM. Concepts such as primary keys or globally unique identifiers have been removed from the model and will be re-introduced when generating PSMs.
- **Clarified semantics:** As a PSM with a long history, the semantics of the JC3IEDM are not always easily comprehensible, since the structure of the model is influenced by technical constraints and design rules as well as operational requirements. Much effort has been spent on clarifying the meaning of the MIM. Toward this goal, all of the associations of the MIM have been evaluated with respect to their definitions, role names and navigability. Furthermore, a rewording of all definitions has resulted in a better comprehensibility of the intended meaning and usage of attributes and classes. One of the most important additions was the use of stereotypes on attributes to categorize them according to the UN CEFACT class words.
- **Formal Consistency Rules:** In the JC3IEDM several usage and consistency rules (often called business rules) have been expressed in tabular form and free text. In the MIM, most of these rules have been addressed by making the structure of the model more explicit. For example, rules constraining the allowed values in attribute combinations have been remodelled such that only valid combinations are expressible in the model. In the cases where this was not possible or desirable, the rules have been formalized using the OCL.
- **Documentation:** The documentation of the MIM is currently under development. The first chapters already have been written. The documentation will be part of the MIM, using object diagrams to illustrate the intended use of the model. Some scripts have been implemented to ensure the consistency of the examples and the underlying class model. Generating the full documentation from the model automatically will ensure an up-to-date and consistent documentation, subsequent to model changes.

Another important difference between the JC3IEDM and the MIM is the conceptual separation of the information model from the exchange interface specification. In the future, MIP will deliver multiple small interface specifications, each covering one specific operational capability. These specifications will all be based on the MIM but will use only a small subset of the model's elements. This so-called capability-based approach allows the MIP Community to be much more open to input from other communities. In the JC3IEDM the addition of a single attribute or value to a coded list would require the release, implementation and test of a new baseline. In the future, these modifications will only appear in those interface specifications that are based on the modified part of the respective model.

3.1. Model Description

In addition to being platform independent, the MIM has some additional features that make it easier to understand and use. One of its key features is the separation of metadata (e.g. time, source, security classification, etc.), information groups (e.g., overlays), and operational core elements (e.g. objects, actions, plans/orders, etc.). This means that the core elements can be described in a stateless, source-independent, and context-free manner and consequently allows for a much cleaner and stricter specification. For example a person could have multiple statuses in the JC3IEDM. However, the reason for this was not documented. It might have been for one of the following reasons: state - the status may change over time; source - different reporters may report different statuses; context - the status may be different for a planned situation or none of these and one object may actually have multiple statuses at the same time, reported by the same reporter in the same context. So the MIM took the approach to remove these different dimensions. Consequently, the association between Object and Status became a one-to-one relationship and the status attributes have been merged with the Object hierarchy. Since adding these different dimensions back to the model is a simple transformation, the MIM did not lose any expressiveness.

The high-level core elements are depicted in Figure 5. Since all operational concepts of the JC3IEDM have been retained, this view looks very similar to a view of all independent entities of the JC3IEDM. The core of the model comprises an extensive hierarchy of battle space objects such as Organisations, Materiel, Facilities, Features, etc. This taxonomy contains approximately 150 different classes.

Another part of the model allows the specification of Actions along with their Resources, Objectives and Effects. At the time of writing, the Action structure of the MIM is under discussion and will be revised in accordance with feedback from the C-BML community.

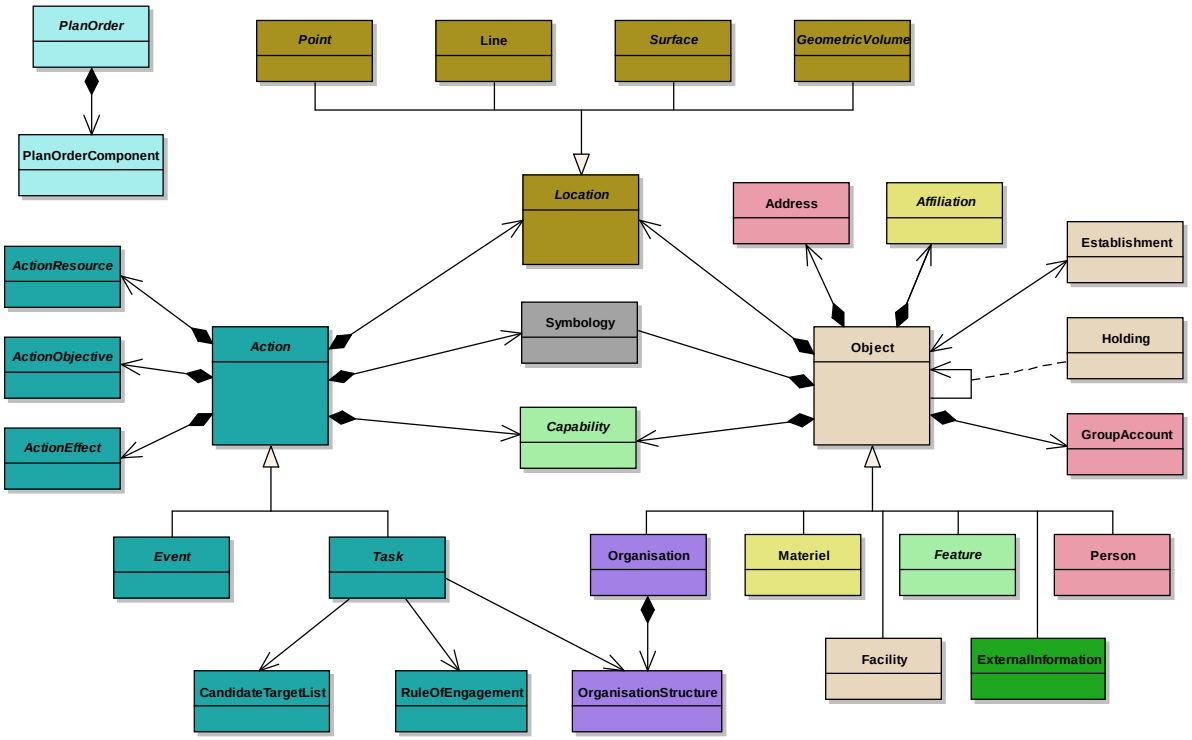

Figure 5- MIM Core Classes

The Location hierarchy includes absolute and relative points, lines, surfaces and volumes. One of the many differences between the MIM and the JC3IEDM is that in the MIM Locations are modelled as part of a composition relationship (or strong association), which means that, according to UML semantics, location instances cannot be re-used. This gives locations the notion of value objects, i.e. they are defined by their exact coordinates and do not need an additional identifier in the PSM.

Since it is assumed that metadata is applicable to all kinds of information and all information may be grouped, information groups and metadata are not linked explicitly to the core elements of the MIM. Instead, a transformation

will create the necessary links when generating the PSMs. This greatly reduces the number of associations in the MIM and thus greatly improves the comprehensibility and clarity of the model.

3.2. Change Process and Tools

The experience of maintaining and extending an extensive information model in a multinational environment has shown that it is essential to keep track of all changes that modify the model in order to be able to trace them back to their authors and rationales. Furthermore, the established process of developing the model requires that all changes and their rationale be accepted unanimously. Thus, in a community-driven specification process, change proposals have to be discussed and documented prior to applying them to the model. To ensure that a proposed change can be applied to the model without manual intervention once it has been accepted by all stakeholders, Fraunhofer FKIE has developed a tool that accepts change proposals in an XML format as input to the tool that applies them to the model. While performing the formally described changes on the model, the tool also enforces several consistency checks and notifies the user of possible derivations from design rules and best practices. Since the tool can be used to validate a change proposal prior to putting it up for vote, it is obvious that an accepted change proposal will be applicable to the model without requiring manual intervention and thus the possibility of introducing errors is nil.

Another major advantage of this process is that it creates the potential for parallel work. Since each change proposal only specifies particular desired modifications to the model, the tool performing these changes can identify overlaps in conflicting change proposals. Even though this may seem trivial, it is the basis for the previously described capability-based approach in which each capability package defines a small subset of the MIM and then modifies/extends this subset, as required.

The MIP has developed and maintains several different tools that support the previously mentioned change process, as shown in figure 6:

CP Editor: The CP Editor is a tool that can be used to load and browse the MIM and to create change proposals. It still is in early stages of development and but already has the capability to visualize minimal subviews of the MIM. A minimal subview is defined as all classes, attributes and associations that are required to be compliant with the MIM. The idea of a minimal subview is similar to the concept of a *Transactional* as described by OMGs *Shared Operational Picture Exchange Services* (SOPES). The graphical editor is shown in Figure 7. The left side of the editor is a tree view of the model, showing all packages, classes and attributes as well as all tagged values of the currently selected element. At the bottom, all associations of the currently selected element are shown. The center and the right side of the editor are two different views on the subview. The center is a graphical view with the explicitly included classes shown in light blue and the required classes shown in gray. The right side is a more textual view of the same subview definition.

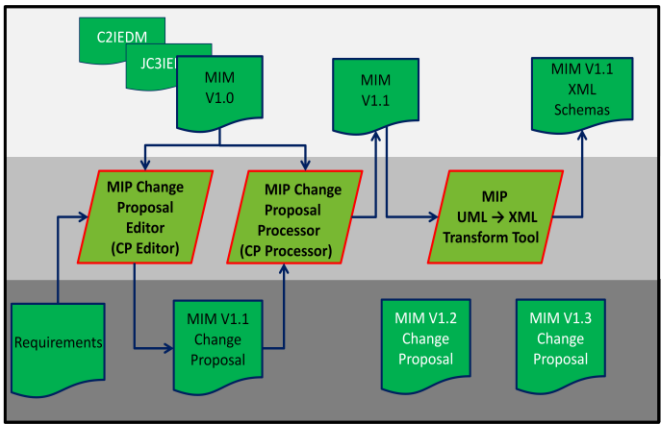

Figure 6 - MIM Tools Overview

CP Processor: The CP Processor applies a formal change proposal to a model and can execute change proposals created using the CP Editor. Currently, two types of change proposals are supported:

1. A subview definition (also called Business Object Change Proposal) is a change proposal that creates a minimal subview which contains all elements defined in the change proposal. By default, the minimal subview does not include optional attributes. However, the subview definition can define optional elements explicitly, as well as suppressing mandatory attributes by setting them to a fixed value.
2. A formal change proposal describes the intended changes both formally and textually. These formal changes are basic operations such as “create”, “modify” or “delete” on UML elements such as packages, classes, attributes, stereotypes and associations.

Transform Tool: According to OMGs MDA approach, a PIM such as the MIM can be transformed into a PSM. The transform tool supports multiple transformations that can be applied to the model in order to (re)introduce certain

aspects or patterns in the model. For example it is possible to add the value “unknown” to all enumerations in order to allow users to express that a value may not be known.

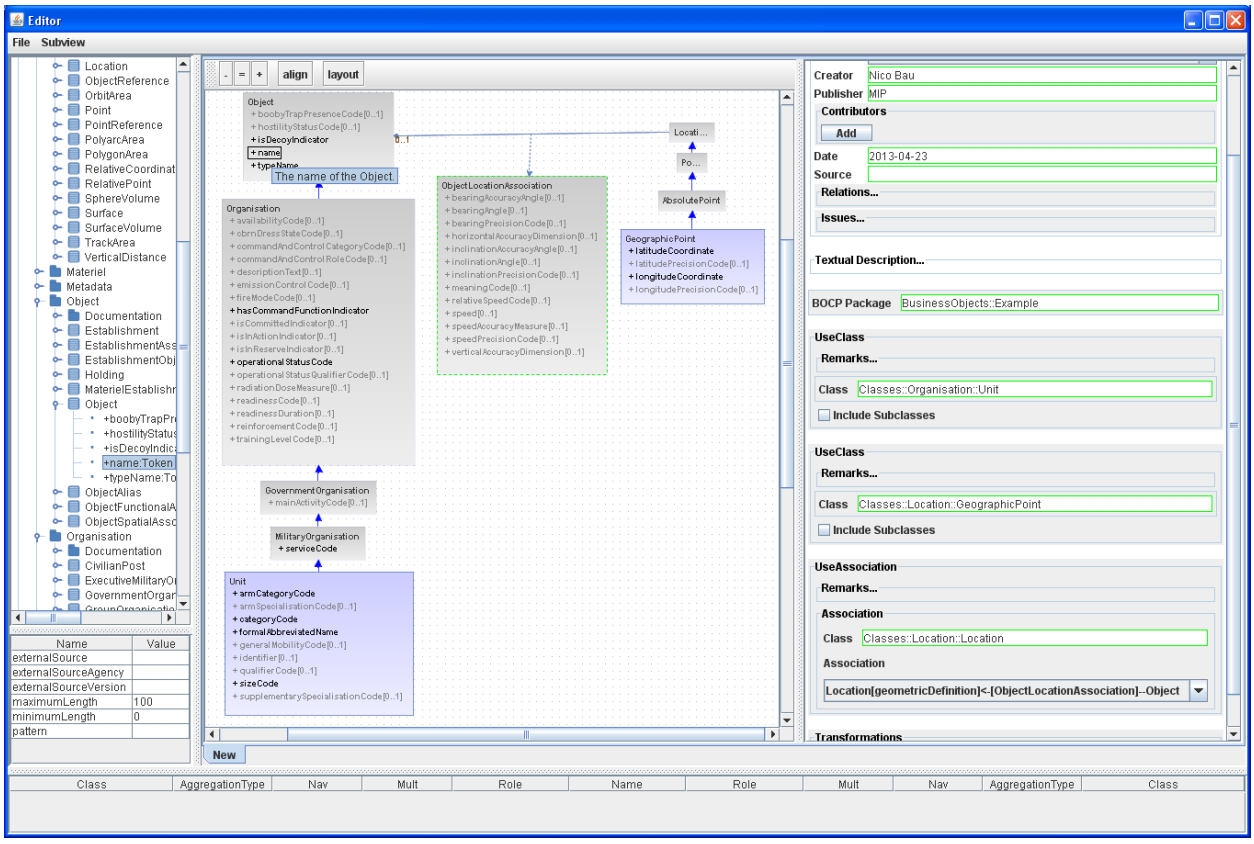

Figure 7-MIP Model Editor

3.4. Applying the MIM Change Process to C-BML

As illustrated in the C-BML Phase 1 model structure represented in Figure 4, the preliminary C-BML standard already reuses many JC3IEDM types. However, the model shown is an implicit model captured as a set of XML schemata that have been built using an ad-hoc process. Applying changes to the model has proven problematic for several reasons: *What is the motivation for a specific change? How will this change affect the existing model (i.e. what are the consequences)? Does it “break” the model? Do the stakeholders agree with the change based on the known consequences of the change?* The inability to answer these questions in a timely and efficient manner likely has contributed to the difficulties in respecting the C-BML standard development timeline.

The MIM Change Proposal process outlined above applies systems engineering best practices and leverages enterprise architecture constructs as outlined in the MDA approach. The structured approach that has been developed to define and manage change proposals for the MIP community also can be utilized in the C-BML standard development activities – especially since the foundation for the C-BML standard are the MIP models.

The following sections describe a systems engineering/enterprise architecture approach for developing interoperability standards and how it can be applied to the development of the C-BML standard.

4. A Systems Engineering / Enterprise Architecture Approach for Interoperability Solutions

The term “Systems Engineering” (SE) can be traced back to Bell Telephone Laboratories (circa 1940) while the concepts date back to the 1900s [16]: “…[SE] *has emerged from the post World War II military-industry-academic complex that was embroiled in an accelerating weapons race…*” [1].

The SE Vee Model is more than 20 years old and has been used and re-used in a variety of derived SE methodologies, including iterative approaches, system of system (SoS) approaches, family of systems (FoS), and dual V-Model [17].

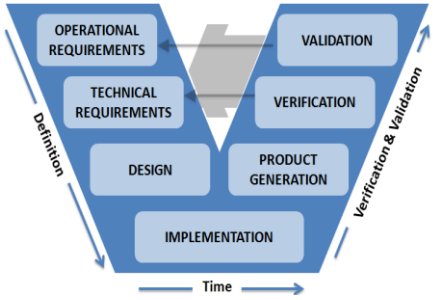

Figure 8-Systems Engineering Vee Model

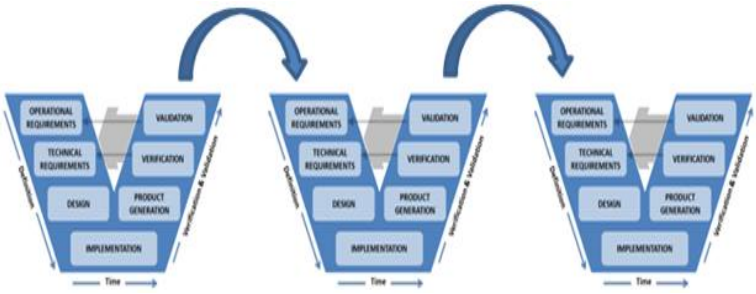

Figure 9-Iterative Vee Model

The basic seven SE elements comprising the Vee model are shown in Figure 8, although the exact terms have been modified slightly from the original model and generalized for use with software systems. The Vee model is not a standard, but it embodies various SE processes, the simplest of which is an improved or extended waterfall method[2], originally introduced in 1970 as a sequential software engineering methodology [18].

The waterfall model assumes that requirements do not change during the development process. Although more flexible than the waterfall model, the basic Vee model still has several flaws, and the sequential nature of the activities still is present as a linear progression through the following phases: 1) definition; 2) implementation and 3) integration and testing, with stakeholder needs and requirements definitions activities cross-connected with validation and verification activities, respectively.

4.1. The Iterative Vee Model

To remediate the basic sequential nature of the Vee Model, the iterative Vee Model, incorporates several "Vee" iterations within each engineering phase, as illustrated in Figure 9Error! Reference source not found..

The main advantage of the iterative Vee model is that it maintains the rigor and traceability of the Vee model while introducing the flexibility and other benefits of iterative, incremental methodologies. Though the iterative Vee model supports changes in requirements while enabling traceability, Requirements Engineering has emerged as a key component of Systems Engineering and is deserving of further amplification.

4.2. Requirements Engineering

The discipline of Requirements Engineering (RE) is traditionally a software engineering process with the aim of identifying, analyzing, validating and documenting system requirements. An integral part of SE, it involves the following requirements activities: elicitation; analysis; documentation; validation and management. It also is particularly relevant to the development of standards. Proper RE assumes that requirements may change over time and should allow distinguishing characteristics such as: description, notes, priority, owner, status, complexity, version, phase etc.

Agile software development methodologies also have RE activities, but software quality factors and non-functional requirements are not always well-handled [19]. Software quality factors include considerations such as maintainability, usability, reliability, efficiency, and portability [20].

4.3. Requirements Management

Systems engineering typically deals with specifying, designing, building and testing systems. Hence, the requirements management activity focuses on system requirements. In the context of standards development, the *system* is a *standard* and this introduces several particularities. Once completed, the standard can be used to specify

[2] http://www.waterfall-model.com/v-model-waterfall-model/

or constrain a system design. In the case of C-BML, the end-user system is a C2-simulation federation or System of Systems (SoS) and three levels of requirements can be distinguished: *standards*; *system design*; *SoS/Federation design*. However, *interoperability* standards such as C-BML should consider primarily *interoperability* requirements.

As an emerging technology, C-BML has many uses, some of which are based on current short-term needs and others are based on future concepts that still require maturation and validation. As part of the requirements elicitation activity, stakeholders will provide all types of requirements and thus, at times, it can be difficult to extract out only the subset of technical requirements that is relevant to the standard [18]. Reference [18] advocates a systems engineering approach to standards development.

The SE methodology for standards development must include a RM activity that is grounded in operational requirements. These requirements in turn must be traceable to derived requirements that finally are traceable to the specific elements of the standard to which they relate.

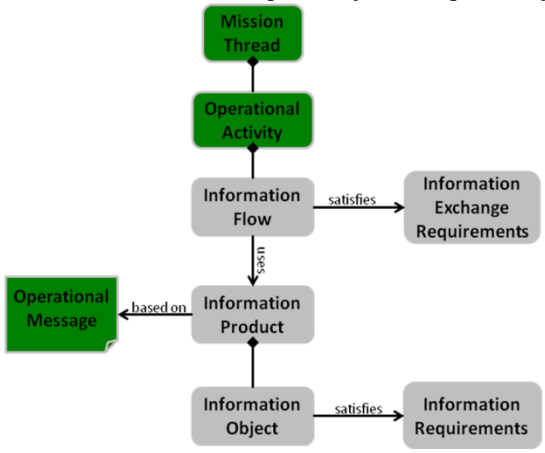

Figure 10-C-BML Requirements Map

Consistent with the NATO Architecture Framework (NAF), Figure 10 illustrates the underlying requirements elicitation mechanism. Requirements are derived from *information flows* that enable specific *operational activities*. The information flows involve the exchange of *information products* that are comprised of *information objects*. To maintain operational relevance, information products are based on actual operational messages as per existing procedures. In many instances not all of the information elements in a given operational message are required by the information product since the latter is intended to communicate a subset of the information in the operational message. For example, in the case of C-BML, simulations generally cannot parse free-text elements of operational messages intended for human consumption and therefore all free-text elements should not be included, by default. Nonetheless, responses from simulated forces may include free-text fields indicating, for example, the reason for a negative acknowledgement of a specific task execution. In general, two types of requirements are identified: *information requirements* (IR) and *information exchange requirements* (IER). In general, IER may be operational requirements, system-specific requirements or technical requirements. For the purposes of this approach, IER are those requirements that are associated and/or derived from the operational information flow. IR refers to the set of lower-level requirements related to specific information elements or data elements.

As part of the proposed approach, the requirements management includes traceability both to and from the PIM such that program managers rapidly can determine which requirements have been satisfied or are planned to be implemented in a given model revision. At the same time, a modeler easily can have access to the set of requirements that are satisfied by a specific model element such that change proposals can be handled rapidly and efficiently without *breaking* the model.

4.4. Modularity and Agility

Once all IER have been traced to model elements, the information model describes the superset of all information elements that need to be exchanged. However, the underlying operational processes often only require a small part of the IER to be satisfied and thus need only a subset of the full model which likely reflects IER from several communities of interest. Thus, the model should be modular in order to create meaningful subsets which allow communities of interest to address their specific IER. When constructing the MIM from the JC3IEDM, increased modularity was achieved by dissecting overly generic constructs of the JC3IEDM into semantically grouped structures. For example the generic OBJECT-ITEM-ASSOCIATION, a single association which is used in the JC3IEDM to express many different relations that objects can have (such as A is-parent-of B, A is-left-of B, A detects B,..), has been split into several different associations that describe a number of more specific relations (such as social relationships, spatial relationships, functional relationships,..). This allows communities of interest to select only the associations and types that are necessary for their applications.

If a community of interest wishes to extend or modify a subview to address their specific IER, the previously described change process can be applied toward the definition of community specific change proposals which

subsequently can be discussed and agreed on within the community. This allows for a very flexible and agile development of an extended subview. Once the subview is mature and the community decides to share some or all of their extensions, the change proposals may also be officially submitted for consideration by the C-BML Product Support Group or by the MIP. Thus, community-specific changes that may be of interest to the larger community can be harmonized over time.

One of the early outcomes of the process described in this paper is already reflected in the current version of the MIM. Based on preliminary work, the C-BML community submitted a change proposal specifying a package structure to better organize the classes of the MIM. This change proposal was provided to the MIP community and subsequently put into the model.

4.5. Maintainability of the Standard

Operational processes such as the *AMN Coalition Mission Threads* often span multiple communities of interest. In order to allow an uninterrupted flow of information, it is essential that all participating systems are interoperable within the scope of the operational scenario. So even though each participating community may have identified their unique IER, the overlap of the exchanged information should be sufficient to support the complete process. The identification and harmonization of this overlap, especially in an international context, is an important task.

The JC3IEDM and its successor, the MIM, already constitute a solid model corresponding to a harmonized set of generic IER. Several aspects ensure that the MIM can be maintained and extended in the future.

- First and foremost, the MDA approach, which allows users to automatically generate PSMs that are tailored to their specific needs, alleviates the need to standardize on design styles and formats. The described process allows communities to define their own subview without the need to consider platform specific aspects. They can then use the provided transformations to generate PSMs for a specific exchange format in a specific design style. Adding support for a new format or design style is as simple as creating a new generator. Thus, the identification of overlaps in the information domain of different communities of interest can be done on a platform independent level, where it is easier to identify commonalities.

- Second, the standardized change process enables traceability of information elements to specific communities of interest and even individual IER. Thus, it becomes possible to start a harmonization process when two communities intend to modify the same model element.

- Third, the change process allows the model and specific subviews to evolve in parallel. The CP Processor will identify conflicting modifications if an existing change proposal is applied to a new version of the model. Only in cases where a conflict has been identified manual intervention is required.

- Fourth, by integrating all consistency rules and the documentation in the MIM, it becomes much easier to keep these different artifacts consistent as the standard evolves.

4.6. Requirements Traceability and Validation

Traceability of requirements is at the heart of development practices for the aircraft industry, as specified by the aircraft industry so-called airworthiness standards, such as DO-178: *Software Considerations in Airborne Systems and Equipment Certification* [21]. A distinguishing feature of DO-178 compliant software development processes is that traceability from system requirements to all source code can be required.

For the technical standards development, requirements management helps to clarify aspects such as their relative importance, urgency, priority, etc. and thus facilitates the elaboration of standards products development plans. The ability to link elements of technical standards back to derived and operational requirements also helps to understand why the standard was constructed in a certain manner. Moreover, as requirements for standards evolve over time, the link between elements of the standard and the requirements becomes an invaluable part of a managed change request process. Otherwise, how does one know whether a specific change can be applied without *breaking* the standard, i.e. causing provisions to become inconsistent? That is to say, how can one be sure that proposed changes will satisfy new requirements while satisfying existing requirements?

4.7. Automating the Standard Development Process

Enterprise Architecture requirements management approaches now are integrated into UML tools and provide the means for ensuring traceability of requirements [6][22]. Therefore the following features and capabilities are readily available to aide in establishing an enterprise architecture, systems engineering standards development environment:

Automating the Standard Development Process

Once a process has been defined for developing the standard and for producing a set of products or *artifacts* comprising a technical standard, it then is possible to automate the generation of artifacts.

Process Documentation

The process itself must be well documented and well understood to be utilized successfully by stakeholders. The process can be captured as part of the UML model itself. Using automated documentation generation capabilities, the process description can be exported as a set of Hypertext Markup Language (HTML) pages.

Requirements Specification Generation

Requirements can be captured and traced as part of the model and specifications can be generated automatically at regular intervals in order to facilitate organization of requirements and internal and external validation.

PIM Definition and Automatic PSM Generation

The PIM can be represented as a UML model and alternately as a set of OWL ontology modules. Different PSM can then be generated in formats such as XML, one of the *de facto* choices for representing structured data and also some languages. However this approach also allows for the generation of other formats, such as the High-Level Architecture (HLA) or JavaScript Object Notation (JSON).

XML Schema Description (XSD) documents are being used increasingly to define interoperability standards such as SISO Military Scenario Definition Language (MSDL), C-BML and also the National Information Exchange Model (see https://www.niem.gov). However, at the heart of an interoperability standard, there is a model and it is not always easy to conceptualize or understand the relationships of the various model elements by inspecting of the XSD. Although XSD are model representations, they are not necessarily *normalized* models. For complex standards, maintaining schemas manually is labor intensive and can be error-prone [8].

Therefore there are benefits associated with a structured approach of developing *normalized* PIM using languages such as UML. Consistent with the Model-Driven Architecture (MDA) approach, XML schemata and other desired outputs (e.g. PSM) can be generated from the UML PIM. XSD is one of several possible model transformation outputs but other PSM can then be generated, such as JSON, often used in conjunction with RESTful style web services. Also, UML profiles now are available for architectural frameworks such as the NAF, for various platform-specific language and other technologies (e.g. C++, C#, JAVA OWL, DDS, WS etc…) as well as for SE with the *Systems Modeling Language Profile for UML* (SysML). Finally, UML vendor tools generally offer automated documentation generation features as well including exporting model descriptions to RTF and html formats.

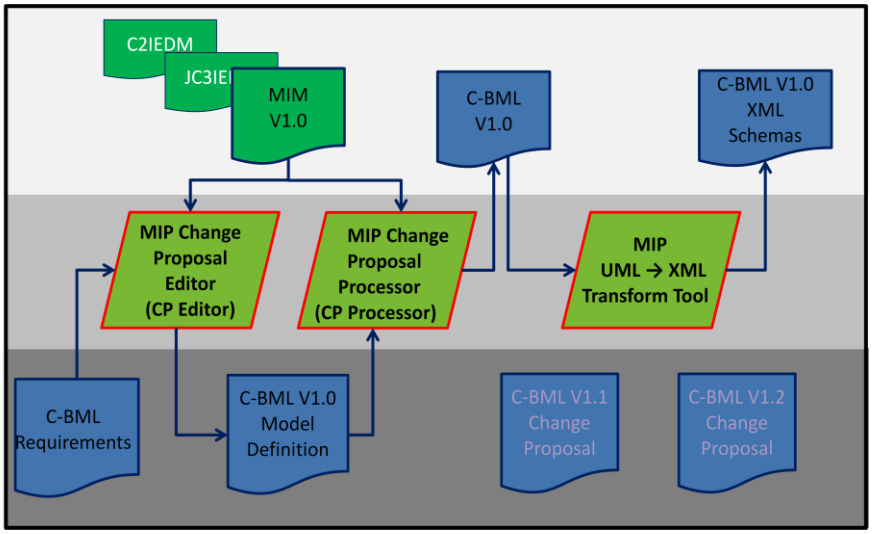

Figure 11 - MIM-based C-BML Production Chain

5. Defining a MIM-based C-BML Subview

5.1. Building on the MIM Foundation

Figure 12 illustrates the types of domain entities, events and properties that comprise the C-BML domain model. The figure also highlights the strong influence and applicability of the MIP JC3IEDM to C-BML.

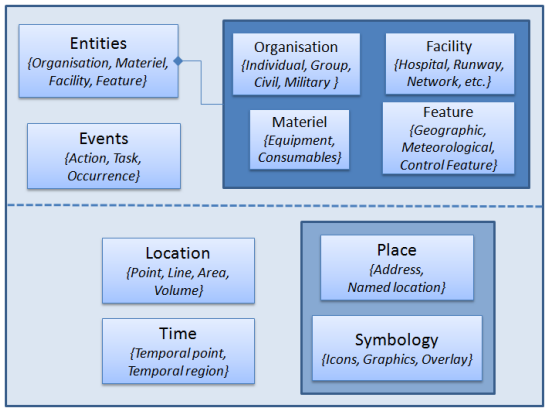

Figure 12 – C-BML Entities, Events and Properties

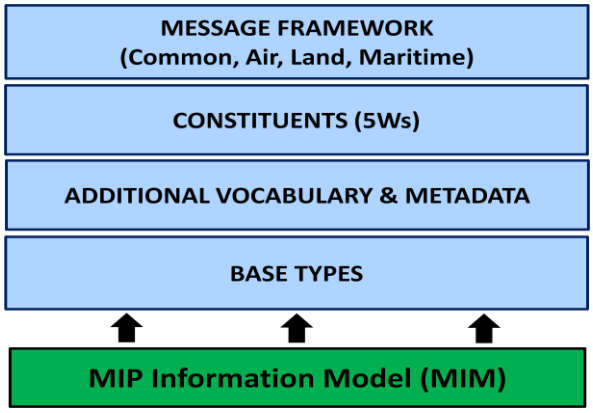

Figure 13 – Proposed C-BML Logical Data Model

5.2. Layered Approach

Figure 13 shows the layered structure of the proposed C-BML Phase 2 model, as advocated in this paper. The first layer is composed of a subset of *base* types that are taken directly from the MIM. The second layer adds additional vocabulary (i.e. terminals) and additional metadata associated with the definition of C-BML Messages as shown in Figure 3. The constituents, also known as 5Ws, comprise the third layer and are the primary inputs into the fourth layer, the message framework that defines the sets of military messages that can be constructed. Note, that the message framework is not a message catalog such as those defined by Formatted Text Message (FTM) standards. The message framework provides the means to represent information contained in domain-specific operational messages in a digitized machine-computable form while satisfying IER for information flows and interaction protocols for complex operational message exchanges, such as those associated with Call-For-Fire or Close-Air-Support requests.

The separation of concerns is an important aspect of developing interoperability solutions, as described by Lang et al [6], as well as for organizing the model and standard in a modular form. Standards serve different purposes for different users from various communities and the model structure must provide for domain extensions without breaking interoperability. The modularity of any solution is one of the keys to ensuring its maintainability. Concerning standards development, another important aspect is the ability to rapidly generate new revisions of the standard based on revised requirements.MSG-085: Standardization for C2-Simulation Interoperation

The NATO MSG-085 Technical Activity (TA) has been mandated by the NATO Collaboration Support Office (CSO) as follow-on activity to the MSG-048 (C-BML) TA [23]. With participation from 13 nations, MSG-085 has been working in the area of C2-SIM interoperation since 2010 and currently is slated to run through 2013. MSG-085 is working in the areas of military scenario definition, initialization, and execution using C-BML and also the SISO MSDL. The main objectives of the MSG-085 TA are as follows:

- Clarify and complement existing C-BML and MSDL requirements;
- Propose a set of C-BML orders and reports to serve as a common reference set;
- Assess and leverage available C-BML implementations;
- Address C2 and simulation initialization requirements; and
- Demonstrate the operational relevance and benefits of the approaches considered.

MSG-085 is tasked with assessing the *operational relevance* of C-BML and to assist in increasing the *Technical Readiness* Level of C-BML technology to a level consistent with operational employment by stakeholders. To

accomplish this mission, MSG-085 has formed two sub-groups: the Operational Sub-Group (OSG) and the Technical Sub-Group (TSG), that focus on operational and technical requirements for C2-SIM interoperability. Moreover representation is present from each service (Air, Land, and Maritime) to ensure the operational relevance of C-BML for multi-national and multi-service use.

Recent research and experimentation have been conducted by NATO MSG-085 that has formed several Common Interest Groups (CIG) to focus on specific areas of interest. CIGs were established for each of the Air, Land and Maritime domains. The Air Ops, Land Ops and Maritime Ops CIGs addressed domain-specific requirements for extensions to existing C2-SIM interoperability standards [25][26][27][28][29]. The OSG, TSG and the domain CIGs have contributed to establishing requirements for C2-SIM interoperability in a UML domain model consistent with the approach described in this paper. The OSG also has led the elaboration of a set of Operational Concept Description (OCD) documents, one for training and the other for mission planning (course of action analysis) [30][31]. The TSG has contributed to an UML-based collaborative workspace for organizing and tracing requirements for subsequent MSDL/C-BML language development. This workspace has been extended for optimal use of the MIM and MIM tools.

5.3. Collaborative Model Development Environment

The current paper advocates the use of UML as a means to formalize requirements and reference architecture. UML tools, such as *Sparx Systems Enterprise Architect* (EA) now include UML profiles and add-ins for requirements management, model transformations, eXtended Markup Language (XML) schema generation, code generation and other actions. UML tools also can support a distributed collaborative development environment based on readily available version control systems, such as *Subversion*. One of the main benefits of employing an UML-based standard development approach is the use of built-in document generation capabilities. Being able to generate standards product artifacts in an automated fashion can contribute greatly to both the maintainability and the usability of the standard, as described in the next section.

5.3.1. Centralized UML Repository

A centralized, distributable UML repository has been created using the EA distributed model configuration functionality and the Subversion collaborative software development tool. In this manner, requirements from various users can be collected and managed within the same model.

5.3.2. Automated Generation of Model Artifacts

The UML model environment includes a number of automated artifact generation capabilities. There are two areas where this has been used with success: 1) Documentation Generation; and 2) Model Transformation. The EA documentation generation allows the user to define templates for the automatic generation of documents such as requirements specifications, traceability matrices, model description documents and others. A similar function supports a HyperText Markup Language (HTML) output for increased browseability. The Model Transformation feature is of particular interest to C-BML since it allows for a set of XML schemata (i.e. PSM) to be generated from an UML model (i.e. PIM).

Thus the effort can be spent on the most important and difficult task: constructing and maintaining a PIM based on initial and subsequent requirements.

5.4. Preliminary Results

Figure 14 shows an example extract of the XSD PSM model transformation output that has been generated automatically from the PIM or logical data model, similar in form to the one shown in Figure 7. The start-symbol "Message" contains a header, a message-metadata and a message-body. Many of the MIM metadata elements have been reused, but some additional types have been added. Figure 15 shows an example of one type of *messageBody*, an *Acknowledgement*. In this case, the MIM distribution-acknowledgement type is reused directly, while the acknowledgement-category-code has been modified slightly to account for domain-specific requirements.

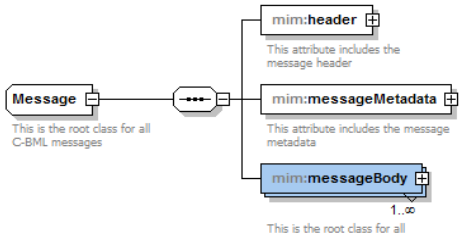

Figure 14 – MIM-based C-BML Message Structure

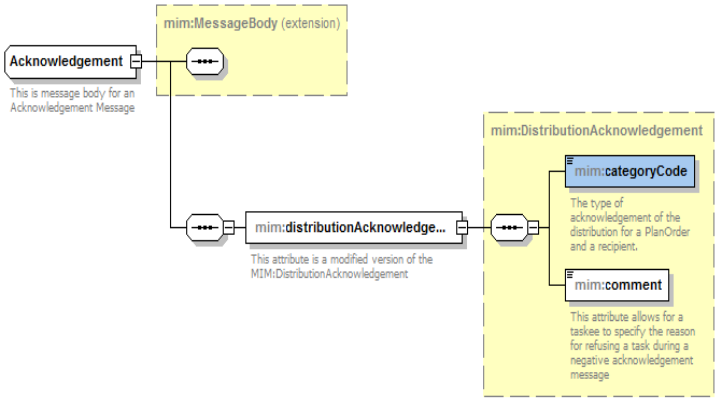

Figure 15 – MIM-based C-BML Acknowledgement Message

Since the C-BML Model Definition process is based on the MIM Change-Proposal process, any change or addition made as part of the C-BML development effort can be communicated directly to the MIM for consideration as a change proposal to the MIM itself.

6. Conclusions

In order to maximize the usability and achieve greater benefits of interoperability technologies, technical interoperability standards are required. These standards products must be derived from operational requirements that are elicited through stakeholder involvement.

The development of international technical interoperability standards for multiple domains and communities from the C2 and simulation worlds is a labor-intensive and complex endeavor. Recent standards development organizations have reported that applying Systems Engineering methodologies coupled with an Enterprise Architecture approach can provide a framework and assist in reuniting the necessary and sufficient conditions for producing a successful standard. One of the keys to ensuring that a successful standard is developed is to establish a requirements management process wherein requirements are grounded in stakeholder operational needs, properly organized, and traced to standards artifacts.

Past experience has shown that producing such standards can take many years unless such a dedicated process is established that ensures proper stakeholder involvement. Furthermore, a well-defined, well-documented, sustainable process is required to ensure that the standard can evolve in a timely fashion and in a manner that is consistent with stakeholder expectations.

This paper advocates reusing the MIP Information Model and latest Change Request Process and tools as the basis for a C-BML standard product process and production chain. The MIP has vast experience in developing and maintaining an interoperability solution for an international user group and has elaborated an efficient, modular, traceable change proposal process and associated toolset for modifying and extending their solution. This process and toolset also can be applied to creating MIM subviews that can form the basis for derived products that maintain partial compliance with the MIM. C-BML utilizes the MIP models as the basis or foundation for the C-BML vocabulary and therefore can be created as a MIM subview. The MIM Change Proposal process and tools have been adapted to meet the needs for the development of the SISO C-BML standard and now constitute a repeatable, iterative, controlled manner to evolve the C-BML standards rapidly and efficiently while meeting and tracking stakeholder requirements.

Although XSD representations of standard products are often necessary to perform system development and integration tasks, it is important that they be part of a larger reproducible process that includes traceability back to the operational and technical requirements and therefore can evolve over time. In the case of C-BML, it has been problematic to craft an implicit model as part of a set of XSD that are manually constructed and maintained. Therefore the approach described in this paper advocates the development of a normalized conceptual model or PIM from which the XSD are derived through automated model transformations. This approach has the additional benefit

of producing several equivalent model representations and documentation while avoiding human-induced errors. Since the model representations are equivalent, it becomes easier to integrate systems that use different forms. Automatic generation of the requirements specifications and the normative and informative standards artifacts also reduces the time between iterations. Building normalized conceptual models or PIMs and then generating PSM and documentation has many advantages. And although XSD do provide an implicit model, this is a PSM and does not replace the need for platform-independent conceptual models.

7. Remaining Challenges and Future Work

The C-BML standard development activity commenced with the formation of the C-BML Product Development Group in SISO in 2006. The fact that it has taken over seven years to reach the milestone of a balloted C-BML Phase 1 product reflects the numerous challenges that the PDG has faced. Many of these challenges and obstacles have been reported in [11] and have been addressed by the approach outlined in this paper. However, other challenges described briefly below still need to be faced by SISO, the standardization body and issues resolved through coordination with stakeholders.

7.1. Deconflicting and Prioritizing Requirements

Initial work conducted under the umbrella of MSG-085 has contributed to collecting and managing requirements via a formal process and has led to establishing an initial set of requirements for military scenario initialization and execution [18][25][26][27] for both MSDL and C-BML standards[3]. This process also has highlighted a number of requirements conflicts. For instance, the short-term sustaining requirements for more efficient Command and Staff Training calls for free-text elements as part of a machine-generated (simulation) acknowledgement message for the benefit of the human C2IS operator that is part of the primary training audience. At the same time, longer-term requirements for advanced mission planning systems to support self-synchronization and concepts like Integrated Dynamic Command and Control [32] impose the absence of any free-text element in C-BML. Similarly, some stakeholders require that data symbology elements be included as mandatory elements of C-BML types for defining such as *units*, *equipment* and *facilities* while other suggest that any symbology information should be made optional since the defining model elements should come from the normalized model.

On a positive note, it is much easier to resolve requirements conflicts and prioritize requirements once they have been properly collected, analyzed and presented back to stakeholders for feedback.

7.2. The Balance between Grammar, Ontology and Business Rules

As the detailed process is being defined and prototype production chain being implemented, it has become increasingly evident that a balance must be struck between the amount of rules that one places in the grammar, the ontology and the so-called business rules. The grammar is the set of production rules that are common to all expressions. They determine the set of valid expressions. The ontology adds semantics and additional constraints or restrictions, but these are not needed by all applications. The business rules are free-text or tabular guidelines that applications should follow in order to "make sense" or rather to avoid illogical combinations.

If one attempts to include too many business rules in the grammar in the form of production rules, then the grammar becomes complicated, difficult to express and associated parsers become cumbersome and difficult to construct. If one puts too few production rules in the grammar, then the language no longer represents a standard interface since it is too general. The ontology is useful to capture semantics and restrictions in a formal manner, but it may not be desirable to impose the use of the ontology on all users for simple C-BML exchanges.

These choices only can become clear as the Phase 2 C-BML model iterations commence and the resulting products can be tested through trial use.

---

[3] At the April 2013 face-to-face MSDL and C-BML PDG meetings held in San Diego CA, the presentation of the approach outlined in this paper contributed to the formation of a Tiger Team tasked with defining a way forward for merging the MSDL and C-BML standards, as planned by SISO since 2006. This already can be perceived as a measure of success of this approach.

7.3. Tools

The existing MIP tools have been developed for the MIP Change Request Process. The process outlined above must be implemented with a toolset that is adapted to this process. This calls for modifications to existing tools, and also for new tools. For example, the XSD generation tool requires slight modifications to account for C-BML naming and design rules and other style guide issues. Also, currently no Ontology is being produced as part of the MIM, yet in theory this is possible using stereotypes from the OMG ODM profile. Prototype tools also are available for this purpose. And the requirements management process as implemented by the MIP is not performed at the same level of granularity and does not allow for the traceability identified as part of the current process.

However, workarounds and temporary solutions are available while tool requirements are finalized and new tool solutions are sought out or developed.

7.4. Way forward

The collaboration between the C-BML community and the MIP Block 4 MIM group is continuing under the umbrella of the NATO MSG-085 Technical Group. The goal is to develop a draft process implemented with a prototype production chain that can produce a draft combined MSDL/C-BML Phase 2 Model and example products, including: a centralized UML PIM; a set of OWL Ontology modules; XML schemata; and HLA FOM modules.

The methodology and tools described in this paper are being applied to the development of a new model called the **S**cenario **I**nitialization and **EX**ecution (SINEX) model. SINEX unifies the MSDL and C-BML standards based on a common set of requirements. To demonstrate the viability of the SINEX approach, the principal SINEX outputs, derived XML schemas, HLA-FOM modules, will be utilized during the final demonstration of MSG-085 tentatively planned for April 2014. This work then will be provided to SISO as recommendations for the proposed unified MSDL/C-BML standardization methodology and process.

8. References

[1] C. Hallam: “An Overview of Systems Engineering: The Art of Managing Complexity”, MIT Engineering Systems Division-Research Seminar in Engineering Systems, October 2001.

[2] Argo, H., Brennan, E.J., Collins, M.W., Gipson, K., Lindstrom, C., and MacKinnon, S., “Level 1 Model for Battle Management Language (BML-1), TEMO Simulation Laboratory (TSL), Fort Leavenworth, KS, 23 March 1999

[3] P. Gonzalez, B. Christie, J. White: “Applying Systems Engineering Principles to the Development of Transportation Communication Standards”, RITA-Intelligent Transportation Systems Joint Program Office, Final Report, FHWA-JPO-089, April 2011.

[4] A. Bridgewater: “Interview with Jim Highsmith: From Apollo Mission to agility manifesto”, Computerly Weekly, December 2011.

[5] A. Mostashari, J. Sussman: “Engaging Stakeholders in Engineering Systems Representation and Modeling”, MIT Engineering Systems Symposium, March 2004.

[6] B. Lang, M. Gerz, O. Meyer, D. Sim: “An Enterprise Architecture for the Delivery of a Modular Interoperability Solution”, NATO RTO-MP-IST-101, Information Systems Technology Panel Symposium, Oslo 2011.

[7] NATO Architecture Framework, Version 3, 2007.

[8] K. Heffner, K. Gupton: “Implementing a Standards Development Framework for the Coalition Battle Management Language”, ICCRTS 2013, Alexandria VA USA.

[9] US Intelligence Community and Department of Defense-Content Discovery and Retrieval Integrated Project Team: “IC/DoD Keyword Query Language Specification”, V2.0-20121003, October 2012.

[10] C. Blais, K. Galvin, M. Hieb, "Coalition Battle Management Study Group Final Report", SISO-REF-016-20060V1.0, July 2006

[11] J. Abbott, S. Levine, M. Pullen: "Answering The Question Why A BML Standard Has Taken So Long To Be Establishes?", Fall 2011 SIW, Orlando USA.

[12] K. Heffner: "NATO MSG-119 C2-Simulation Interoperability Workshop Technical Evaluation Report", May 2013

[13] Multilateral Interoperability Programme: Overview of the Joint C3 Information Exchange Data Model (JC3IEDM Overview), Version 3.1.4, Greding, Germany, 14 February 2012.

[14] U. Schade, M. Hieb, M. Frey, and K. Rein. 2010. "Command and Control Lexical Grammar (C2LG) Specification". Technischer Bericht ITF/2010/02. Fraunhofer FKIE. July 2010.

[15] P. Gustavsson, "Operations Intent and Effects Grammar (OIEG) Specification", Version 1.4 July 2011.

[16] INCOSE: "Brief History of Systems Engineering", International council on Systems Engineering (INCOSE) Web Site, Last accessed 10-January-2013.

[17] J. Clark: "Systems of Systems Engineering from a Standards, V-Model, and Dual V-Model Perspective", Systems and Software Technology Conference, April 2009.

[18] K. Heffner et al., "A Systems Engineering Approach to M&S Standards Development: Application to the Coalition Battle Management Language", 13S-SIW-002, SISO Interoperability Workshop, San Diego CA, April 2013.

[19] F. Paetsch, A. Eberlein, F. Maurer: "Requirements Engineering and Agile Software Development", Proceedings of the 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, IEEE 2003.

[20] ISO/IEC 9126-1:2001, "Software engineering – Product quality – Part 1: Quality Model", 2001.

[21] E. Trejos: "Introduction to DO-178B" July 2010

[22] G. Zoughbi, L. Brian, Y. Labiche: "A UML Profile For Developing Airworthiness-Compliant (RTCA DO-178B) Safety-Critical Software", Carleton University, TR SCE-06-19.

[23] Field Manual (FM) 6-30, Tactics, Techniques, and Procedures for Observed Fire, Headquarters, Department of the Army, Washington, DC, 16 July 1991

[24] NATO MSG-048 (C-BML) Final Report, February 2012.

[25] H. Savasan, A. Caglayan, F. Yildiz, OM Mevassvik, G. Sletten, U. Schade, B. Haarmann, K. Heffner: "Towards a Maritime Domain Extension to Coalition Battle Management Language: Initial Findings and Way Forward", Spring 2013 Simulation Interoperability Workshop, 13S-SIW-022, San Diego, April 2013.

[26] A. Brook, K. Heffner: "Air Operations CIG", Spring 2013 Simulation Interoperability Workshop, 13S-SIW-009, San Diego, April 2013.

[27] B. Gautreau, L. Khimeche, T. Remmersmann, J. Martinet, D. Muniz, T. Serrano, E. Pedersen, J. Lillesoe, N. de Reus, H. Hendersen, D. Liberg: "Lessons learned from NMSG-85 CIG Land Operation demonstration", Spring 2013 Simulation Interoperability Workshop, 13S-SIW-031, San Diego, April 2013.

[28] T. Remmersmann, U. Schade, L. Khimeche, B. Gautreau, M. Pullen, R. El Abdouni Khayari: "Lessons Recognized: how to combine BML and MSDL", Spring 2012 Simulation Interoperability Workshop, 12S-SIW-012, Orlando, FL, 2012.

[29] NATO MSG-085 Land Operations CIG Interface Specification Document, December 2012

[30] MSG-085 C2-Simulation Operational Concept Description Document: Training, 2012

[31] MSG-085 C2-Simulation Operational Concept Description Document: Planning, 2012

[32] P. Gustavsson, M. Hieb, P. Moore, P. Eriksson and L. Niklasson, "Operation Intent and Effects Model", Journal of Defense Modeling and Simulation, Sept. 2010.