On the Building of a UML Profile for the Description of Army Architectures in the Context of Complex Systems

Mario Couture* and Antoine Duval**

*Systems of Systems Section
Defence R&D Canada - Valcartier
Val-Bélair, Qc, Canada, G3J 1X5
Mario.Couture@drdc-rddc.gc.ca

**Now at Canadian Meteorological Ctr
Environment Canada
Dorval, Qc, Canada
Antoine.Duval@ec.gc.ca

Abstract

This paper addresses the work surrounding the conception and building of a Unified Modeling Language (UML) profile to be used to model military architectures in the context of complex systems. The revolution in military affairs, which imposes fundamental change in strategic planning and management, causes a shift from the old stovepipe bottom-up threat-based planning to a new top-down Capability-Based Planning (CBP). CBP involves, for instance, the management of acquisition projects from a more global perspective (at enterprise level) and imposes the use of a more holistic and integrating approach. The UML and the proposed Military Architectures UML Profile (MAU-Profile) provide a language that makes it possible to model military systems and concepts by using such an approach. The MAU-Profile and modeling conventions, which guide the use of UML in the context of System Engineering and CBP, are presented in this paper. An important requirement expressed by military authorities about this profile was that it should be easy to be used by non-experts. In this regard, the structure of the stereotype list takes the form of a tree containing eight main entries from which stereotype names can be easily retrieved. This structure is based on our generic definition of the word "system". Some examples showing the use of the profile in a context of CBP are also presented.

1 Introduction

The Defence Management System is an evolving set of means and tools used in the framework of Canadian Defence Service Programs to deliver services to the Canadian Department of National Defence (DND). While it has experienced many changes for the last 40 years, the revolution in military affairs that started to occur in the nineties encouraged DND to effect fundamental change in their strategic planning and management approach. This change can be summarized as a shift from the old stovepipe bottom-up threat-based planning to a new top-down **Capability-Based Planning** (**CBP**). This shift is based upon the 1994 Defence White Paper (DND, 2004a), Canada's defence policy document. This last document prescribes the exploration of innovative ways of acquiring and maintaining equipment. Michaud (2004) and Corbett (2004) give an overview of CBP.

The Canadian military acquisition undergoes major transformations. These involve many changes in military domains, organizations, used technologies, and

processes. Once completely functional, CBP will make it possible to manage capability development by emphasizing on integration, option analysis and detailed trade-offs analysis between acquisition projects. It will also lead to the identification of optimum investments in a restricted governmental financial framework.

CBP involves the use of enterprise perspectives or views that should consider many different acquisition projects and domains at the same time. In this context, traditional System Engineering disciplines taken separately might not be sufficient to address this new complexity. The concurrent re-use of current mature disciplines (like those found in System Engineering, Software Engineering, and System Thinking) into a holistic and integrating approach seems to be a more appropriate solution to achieve CBP. Some works on this matter can be found in: Cook and Sproles (2000a), Cook and Sproles (2000b), Cook (2001), Sage and Cuppan (2001), Keating and al (2003), Moti (2000), and Chen and Clothier (2003).

Architecture descriptions should also be adapted for CBP. They should support the holistic approach by allowing the modeling and the linking of any relevant concepts that pertain to any domain and project of an enterprise. They should also ease and promote the exchange and sharing of this information between these domains and projects. There is thus a need for a modeling language that should be understood by stakeholders from all levels, that should allow the modeling of any concepts, and that should support the holistic approach. The object-oriented generic UML modeling language (UML, 2004) combined with its extention mechanism were chosen as a solution by military authorities.

This paper presents work currently in progress at DRDC Valcartier to conceive and build a UML profile for this context; it is called Military Architecture UML Profile (the **MAU-Profile**). If used with appropriate CASE tools and databases, the MAU-Profile will promote holism by allowing the modeling and linking of any concepts pertaining to any military domain. Some preliminary work was necessary to identify and understand the concepts that must be modeled. An understanding of the contexts within which the profile will be used is thus presented in Section 2. The MAU-Profile is then described in Section 3. The stereotype list, the use of the profile and some examples of its use are given. Section 4 concludes this paper. The reader will find a definition of important terms in the Glossary.

2 The Context of Utilization for the MAU-Profile

Figure 1 is a high-level and voluntarilly simplified schematic of a top-down military acquisition system. Even if this example does not necessarily reflect the exact future state of the Canadian military acquisition, it is used here to show some concepts that are of importance in building a MAU-Profile. The Government Strategic Direction can be viewed as a predefined set of high-level missions that the Canadian Government asks DND to accomplish at home and abroad. These missions are decomposed into Capability Areas, which are constituted of a set of predefined Capabilities.

Systems of Systems (SoS) that are put into action "in-the-field" (during exercises and real theatres) perform some patterns of actions, which allow the achievement of these capabilities. The management, conception, development and even the use of military systems are not anymore based on threat but on needed capabilities. This means among other things that systems constituting SoS may be re-used within different sets of scenarios (or SoS) to achieve different capabilities. CBP encourages thus the merging and re-use of complementary military systems in a collaboration mode to form operational SoS "in-the-field". All collaborating systems of a SoS are sharing the same common mission: to achieve the needed capability. The orientation of systems' actions should thus be aligned toward the accomplishment of this common mission. In this mode, the capability of the whole SoS to accomplish one mission is greater than the sum of individual systems' abilities (taken separately without collaboration) to achieve the same mission.

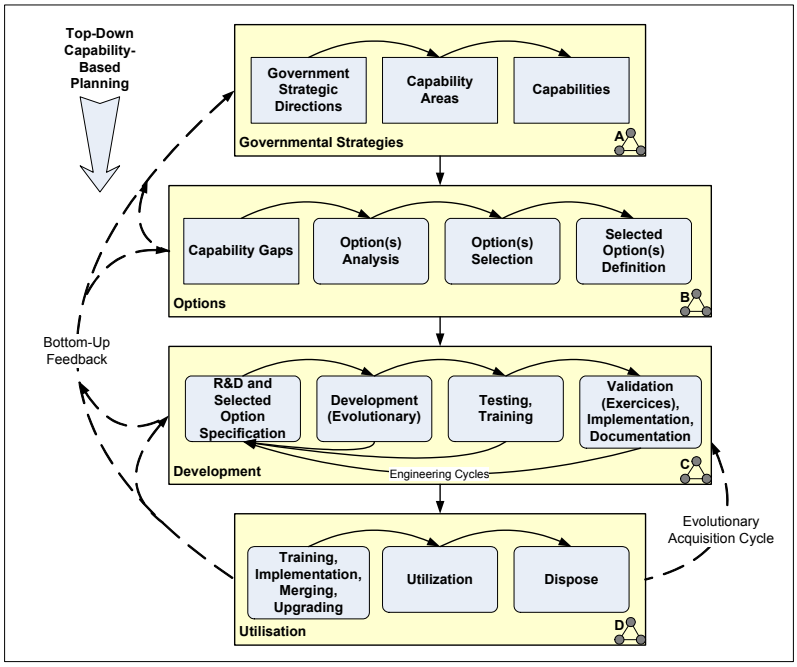

Figure 1. A generic view of the CBP approach in the context of DND acquisition

Once capabilities are identified (in Figure 1), Capability Gaps are then deduced by comparing these needs with what is available in terms of already owned systems to achieve the needed capabilities. Option Analysis and Option Definition processes identify, define and describe specifications for potential options (systems). The chosen options are then detailed and developed based on these specifications. They are tested, validated (by "in-the-field" exercises), and then accepted as "operational" for their use in theatres. Training and disposal are also part of acquisition.

While CBP is a top-down approach, different feedbacks are necessary for the global acquisition to be optimized. This insures that the developed options fit well "in-the-field" and meet concrete strategic, operational, and tactical requirements. As the

requirements for military systems often evolve during the lifecycle of systems, partial capabilities may be delivered using the evolutionary acquisition approach.

Military, industries and academic "in-the-labs" organizations that are achieving such military acquisition are relatively independent (operationally and managerially). They must collaborate to accomplish their common mission. They must perform DND acquisitions in the most effective and efficient manners in accordance with the Government Strategic Direction. Actually, they form SoS themselves; they are people using processes, technologies and materiel to collabotrate with other systems. Four different kinds of SoS have been identified in the example shown in Figure 1. They are represented by three-linked-dots icons A, B, C and D. Types A, B, and C correspond to "in-the-labs" SoS, while type D correspond to "in-the-field" SoS.

Enlarging this SoS perspective to a greater extent, Figure 2 shows that the whole acquisition system can be seen as an overall enterprise complex architecture (depicted as SoS W in Figure 2), which is made of other collaborating complex systems (SoS A, B, C and D). The consideration of both "in-the-labs" and "in-the-field" SoS together as forming itself an enterprise SoS suggests that the global architecture description should consider all its components as part of a whole sociotechnical system (with a focus on interrelationship; Maier, 1998). In this context, any cause/effect that may have influences on emergent behaviors of the whole should be identified, described and modeled in architecture descriptions, no matter the domain or the project (holism).

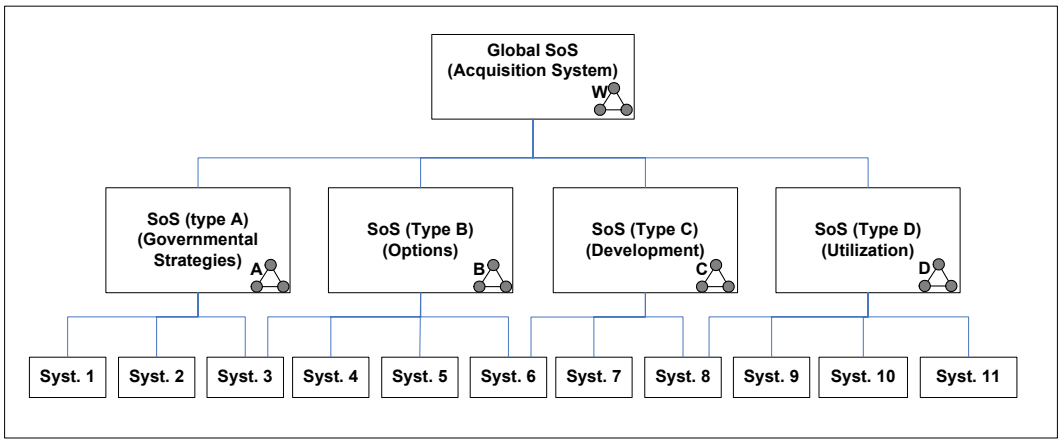

Figure 2. The global architecture of the whole acquisition system

Figure 3 depicts an instance of a simplified acquisition system in the CBP context. This interpretation gives high-level hints regarding how acquisition projects should be managed in this context and what concepts should be captured in architecture descriptions. In this figure, each acquisition project is associated with a predefined Capability, which in turn can be identified to one Capability Area (only two capabilities are shown). The end product of an acquisition project is a SoS of type D that will achieve capability when put in action "in-the-field". For instance, acquisition project 2 aims at developing systems 2, 3, 8, 9 and 13 (SoS 2). These systems will achieve Capability 2 (of Capability Area 1) when they will be collaborating "in-the-field".

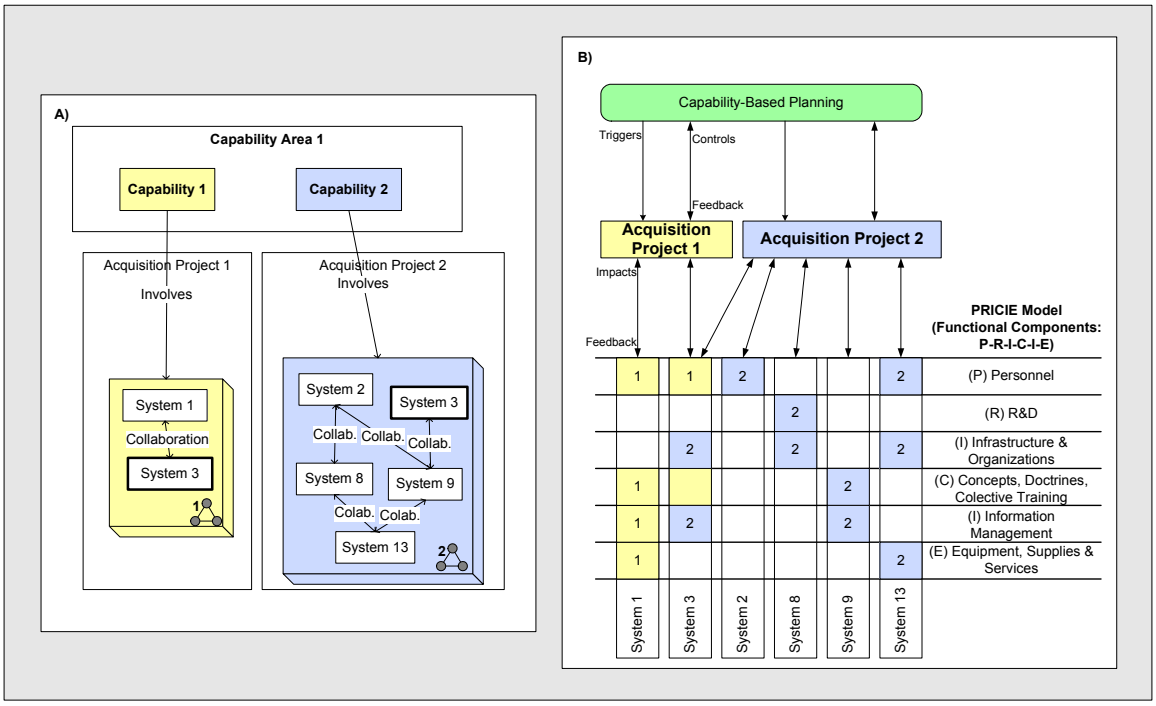

Figure 3. A hypothetical DND acquisition instanciation

As mentioned earlier, in the context of CBP, systems must be re-used within different SoS in order to contribute to the achievement of different capabilities whenever relevant and possible. Any relevant financial, managerial and engineering-related aspects may thus have to be managed or at least known at a higher level (higher than project level). For instance, Figure 3A shows that system 3 would be re-used in SoS 1 and/or 2 to achieve capabilities 1 and/or 2, according to the needs. The two involved acquisition projects (1 and 2) should thus be aware of all specifications regarding the functionalities system 3 should fulfill to achieve capabilities 1 and 2. They should in turn provide the higher level of management all the financial, managerial and technical information that are necessary to the global optimization of efforts (bottom-up feedbacks of Figure 1).

Figure 3B depicts another perspective of the same example. It shows that the CBP should trigger and manage acquisition projects according to needed capabilities (instead of threats), concurrently taking into account already owned and used systems plus the state of other systems under development. Any information about the development of shared systems (system 3 for instance) should be fed back to CBP level for cross-considerations between acquisition projets. Again, this would contribute to ease the global management and optimization at higher level (at CBP level). Figure 3B also shows that PRICIE functional components (DND, 2004b) and the links between them should be considered while addressing capability gaps. The Canadian Joint Task List (DND, 2004b) identifies which functional components should be emphasized while considering capability gaps.

Stakeholders and operators use processes, doctrines, technologies and materiel as means to manage, develop and operate complex systems like the ones shown in Figure 3. This suggests that these means should be used together; they should be interoperable to allow exchanges of information from one part of the architecture to another (that may pertain to another domain or project). For instance, the "on-the-fly" change of a specific requirement (associated with one specific system of one project) may have impacts on other projects involved in the building of a SoS. It may also have impacts on "in-the-field" operations, by making operational systems (or SoS) less efficient for instance. Teams from other relevant domains or projects should thus be aware of the requirement changes; they should validate and approve these changes before they are generally adopted (bottom-up feedbacks in Figure 1). During the "in-the-field" operation of a SoS, metrics should be used to monitor all its relevant behaviours. Measures of SoS complexity (Araujo and Caraça, 1999) of SoS inherent disorder (Hitchins, 2003) and deviation from its initial global mission boundaries are examples of new metrics that should be considered while operating SoS.

The traditional System Engineering that mostly deals with linear and deterministic stovepiped projects and that uses reductionism approach appears to be no more sufficient to deal with such complex systems at enterprise level. Architecture descriptions must also be upgraded in order to provide stakeholders and operators with complete integrated views of the whole (at enterprise level), with appropriate levels of details. These views should focus on components (and their interfaces; Maier (1998)) to measure or gauge the ability of the whole to fulfill its mission. Favoring holism should contribute to raise the synchronicity and homogeneity of the whole. It should also ease the orientation of systems' actions toward the accomplishment of the SoS global mission.

The MAU-Profile is proposed as one supporting solution. The next section gives a description of this profile.

3 The MAU-Profile

The UML modeling language was considered to model "in-the-field" and "in-the-labs" complex systems because of its popularity, its evolution (see SYSML for such evolutions; SYSML (2004); SYSENG (2004)) and its powerful extension mechanism. Using such languages, the process of modeling consists in iteratively creating and evolving models that represent domain elements in an accurate, understandable, consistent and modifiable ways. These models form an architecture description.

Figure 4 illustrates the four-layer hierarchy (M3, M2, M1 and M0 levels) used by the Object Management Group (OMG, 2004) to define modeling languages like UML. The Meta-Object Facility (MOF) is located at M3 level. It is composed of a minimal set of elements that make it possible to define UML at layer M2. UML has its meta-model elements defined at the M2 level and it is used at the M1 level. For instance, the UML model element "class" defined at M2 may be used to represent domain specific concepts or systems at M1. Instances of these model (or real-life objects) are represented at level M0. Figure 4 shows a class representing a "Brigade" at level M1; it may represent any

brigade of the domain of interest, it has no identity. One instance of this class Brigade is the 5th Brigade at M0. This specific instance has one identity. In Figure 4, tagged values (stating the country) are added to M1 and M0 UML elements to show how model parameterization can be achieved in UML diagrams. This four-layer meta-model hierarchy (using the MOF) is not limited to UML; it makes it possible to define other meta-models (at M2 level). For instance, it would be possible to define a new meta-model that would be dedicated to the modeling of business management.

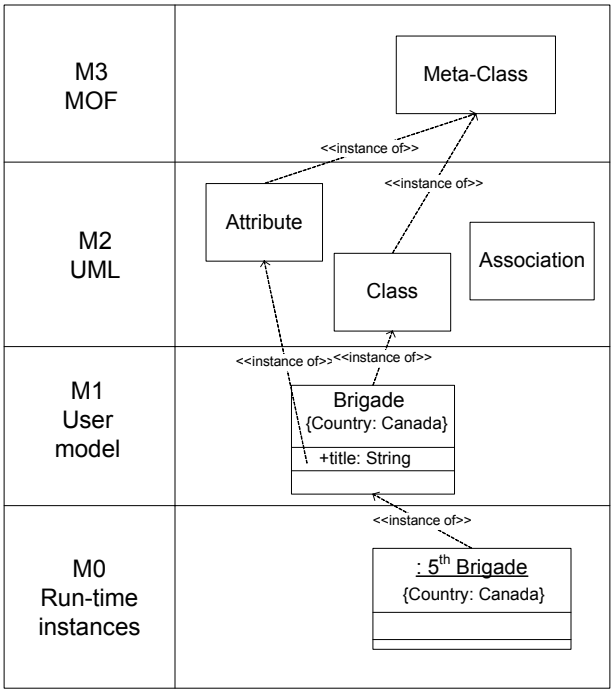

Figure 4. The four-layer meta-model hierarchy used by OMG

UML is mainly used in Software Engineering, but it can be used to model any concept, no matter the domain. Events, decisions, chronological, logical and reference links, computer, software, works, facts, causes and effects are examples of concepts that must be modeled and integrated into an architecture description.

The MAU-Profile is made of a list of **stereotype names**, of **modeling conventions**, and **tagged values**. Stereotype names extend the semantic of UML graphic elements to better characterize UML models for the domains. Modeling conventions guide modelers in their use of UML model elements. Tagged values are attached to model elements to hold some specific pieces of information. The next sections describe the elements of the MAU-Profile.

3.1 Stereotype Lists and Tagged Values

The foundation of the MAU-Profile is depicted in Figure 5. It is based upon our definition of system (see Glossary). All stereotype names are identified and defined on the basis of this definition. It can be summarized by the two following sentences:

To accomplish its Mission(s), a System (or a SoS) transforms Input(s) into Output(s) by doing appropriate Action(s), in specific Context(s), and following a set of Rule(s). All Characteristics that can be measured or evaluated should be captured.

The reader will recognize underlined words of these two sentences in the foundation of the MAU-Profile (Figure 5). These words form the structure of the profile.

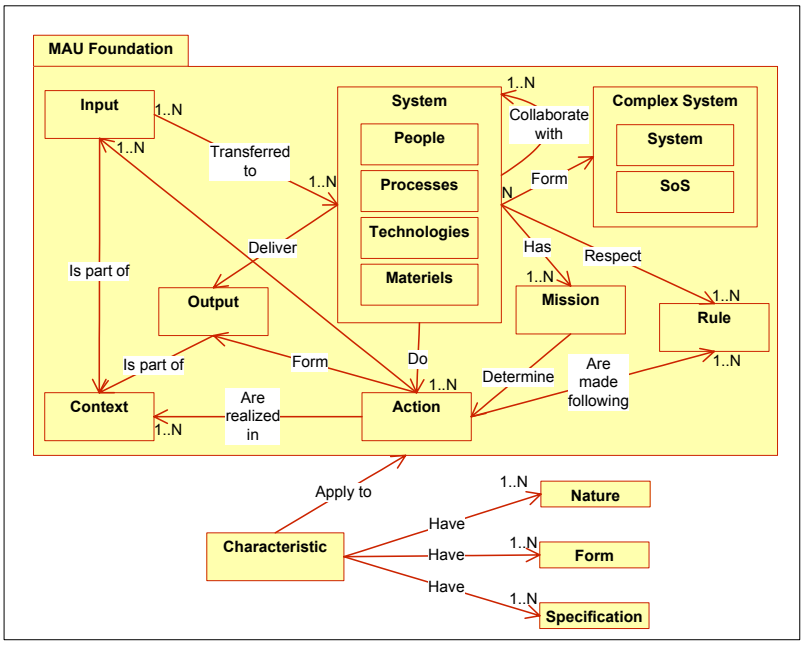

Figure 5. Foundation of the MAU-Profile

Based on this foundation, Figure 6 gives an overview of the structure of stereotype names used in the MAU-Profile. It has the form of a tree (note that only a limited number of stereotype names are shown for clarity purposes). The structure is based on eight high-level entries (System, Input, Action, Rule, Context, Mission, Output, and Characteristics) that are further decomposed into branches containing other stereotypes names. It is a generic classification that can be re-used in any domain; the concept of system can be re-used no matter the domain. It potentially represents a common language that can be used to holistically describe any domain of an enterprise. The complete list of stereotype names will be available in Couture (2005).

The first requirement expressed by military authorities for this work was that the modeling language should be easy to use by non-experts. Minimal training should be needed prior to its use. The lessons learned show that, in high stress operational situations where time is limited, operators prefer to use simpler tools which do a good job rather than hard to learn complex tools which do perfectly the same job. This simple structure (Figure 6) helps non-experts in finding stereotype names in tables by offering logic that is

based on association of semantics from the highest levels (the eight main entries) to the lowest levels (the leaves; not shown in this figure).

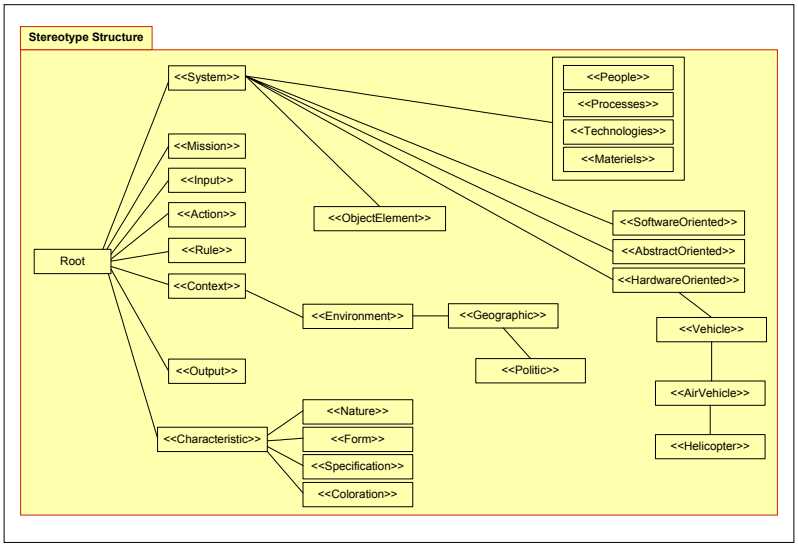

Figure 6 Overview of the MAU-Profile stereotype structure

The semantic of the stereotype name is always related to the semantic of its parents. For instance, the <<Context>> main entry (Figure 6) contains a branch called <<Environment>>, which in turn contains a branch <<Geographic>>, which in turn contains a branch <<Politic>>, and so on. One or many of these stereotype names can be used concurrently to express different flavors of <<Context>>.

A stereotype name is not exclusive to the branch it belongs to. Stereotype names may have different semantics, so the name can be re-used in different contexts. For instance, the stereotype name <<Area>> may be associated to space (<<Spatial>> <<Area>>), to mission (<<Mission>> <<Area>>) or to any domain (<<Domain>> <<Area>>). In the MAU-Profile, a stereotype name can be re-used as needed with different semantics. One or many stereotype names can be used together to make models' semantic more precise.

Some of the stereotype names were extracted from the C2IEDM (C2IEDM, 2005) and the CADM (CADM, 1998) data models. One of the main advantages of re-using such data models concepts is related to the fact that they are dedicated to the military domain. The DOD-AF (DOD-AF, 2003a; DOD-AF, 2003b; DOD-AF, 2003c) may be used to describe DND architectures. Re-using CADM's entity and relationship names will contribute to make the profile DOD-AF compatible (actually, this framework already uses UML diagrams).

The proposed stereotype list aims at providing names that will be added to UML elements for high-level classification purposes; they are not intended to make UML elements "very close instances". This constraint will prevent the stereotype list from becoming huge and unusable. For instance, the word "Apache", which is associated to a

specific kind of helicopter, will not appear in the stereotype list. Instead, the modeller must choose the stereotypes <<AirVehicle>> or <<Helicopter>> (or both) from the list to specialize the UML model element. The modeller should then use the word "Apache" to name the model. Instances or objects of this model would have their name adapted to specific object (having one identity), an example of a name of an instance is: XL25Z: Apache. The last part "Apache" is the name of the class and first past identifies the real-world instance.

Further parameterizations or characterizations of the UML model elements are achieved by adding tagged-values. It is suggested in this first version of the MAU-Profile to use CADM and C2IEDM data models' entities, relationships and attributes as tagged-values. They offer the advantage of being used by Canadian Forces organizations and they are relatively complete and stable.

Keeping the list of stereotype names as small as possible has many advantages when comes the time to use it. If all names representing military objects or concepts had been added to the stereotype list, it would have been huge and unusable.

3.2 The Use of the MAU-Profile

Modeling conventions are used to specify how to use the UML model elements and its profile in the context of System Engineering. They also contribute to standardize the ways architectures are modeled by different people. Questions like: "what does a UML component means while modeling systems other than software" are answered in this section.

In Software Engineering context, an **instance** of a class means "the runtime instance running on a processor and in memory". Military systems are not made of pure software only. Thus, an instance of any system represents "the instance that has a living in the real world". This definition of "instance" for system is more generic.

In the context of Software Engineering, an **interface** collects a set of operations that constitute a coherent service offered by a classifier. An interface is thus a collection of operations with a name. It cannot be instantiated. In the context of military systems, the concept of interface is more general as it has to include any kind of exchanges between systems of any nature. It is defined by the common limits and means that allow collaborations or exchanges between two systems. In this context, interfaces allow exchanges of fuel, energy and other kinds of physical and abstract flows like data or information.

Based on these definitions, modeling conventions that define the MAU-Profile are described in the following lines:

- **Naming UML element**. The name chosen to identify a UML element should be representative of the concept modeled.

- **Using stereotype names**: Any UML elements (classes, nodes, association, etc.) can be stereotyped (using the MAU-Profile stereotype names) to represent any domain elements. Stereotype names can be used at M0 and M1 levels. In the MAU-Profile, a stereotype name can be re-used as needed with different semantics. One or many stereotype names can be used together to make the models' semantic more precise;
- **Using tagged values**: Tagged-values can be used to further characterize UML elements for the domain. CADM and C2IEDM entities/relationships are proposed as tagged-values for this version of the profile. Tagged values can be used at M0 and M1 levels;
- **Using namespace**: A stereotype might be prefixed by a package name (using "::" to separate both names). The package name for any MAU-Profile elements is "**MAU**". That should solve naming conflicts if other profiles or stereotypes are used in the same diagram. For example, <<System>> and <<MAU::System>> are equivalent stereotypes for the MAU-Profile;
- **Using UML classes**: MAU-Profile rules for using classes are the same as those stated in UML 2.0 specifications. A class is the basic element or model element that can be used to represent any domain element;
- **Using UML components**: In Software Engineering context a component represents a modular, deployable and replaceable part of a system that encapsulates implementation and exposes a set of interfaces. In the military systems context, a component instance is a real world part of a system (or system, or SoS) that is modular, deployable and replaceable. It encapsulates implementation and exposes a set of interfaces. A component may include other UML components, nodes and classes;
- **Using nodes**: In Software Engineering context a node is a runtime physical object that represents a computational resource or platform, generally having memory and processing capability, upon which components may be deployed. In the military systems context, a node is a real world resource element or platform that supports other domain elements. A node may include other UML nodes, components and classes;
- **Using packages**: In Software Engineering context a package is a grouping and structuring of model elements. A package cannot be instantiated. In the military systems context, a package is a logical way of grouping UML elements that may represent any domain elements;
- **Using instances of stereotyped UML elements**: Instances of any UML elements may be stereotyped by the name <<instance>>. If a UML class is stereotyped with the name <<System>> then <<System Instance>> is the stereotype name for its instance;
- **Using "logical" UML 2.0 diagrams**: Stereotyped UML elements may be used within any standardized logical UML 2.0 diagrams (class, objects, state, sequence, collaboration and activity diagrams). Some diagrams may have to be modified to meet System Engineering needs. For instance, with MAU-Profile it should be possible to use node instances in sequence diagrams to depict systems' chronological activities;

- **Using "physical" UML 2.0 diagrams**: Stereotyped UML elements may be used within any standardized physical UML 2.0 diagrams (components and deployment diagrams). Some diagrams may be modified to meet System Engineering needs;
- **Generating computational code**: Stereotyped UML elements (like classes) will not become computational code if they are not pure software.

The object-oriented paradigm is a generic concept that is often applied in Software Engineering, but that is still valid while modeling any kind of systems. Objects will have a state, behaviour and identity. The behaviour may depend upon the state and the state may be modified by the behaviour. Inheritance is possible among hierarchical representation of domain elements (or systems, SoS). For instance, systems like military tank and jeep may inherit from the class Vehicle. Vehicle is a class showing the commonalities between tanks and jeeps (Figure 7). Polymorphism is also possible among hierarchical representation of domain elements (systems, SoS). As an example, Figure 7 illustrates a person driving a vehicle. When the drive operation is "called", the operation to be "executed" depends on whether the object used is a tank or a jeep.

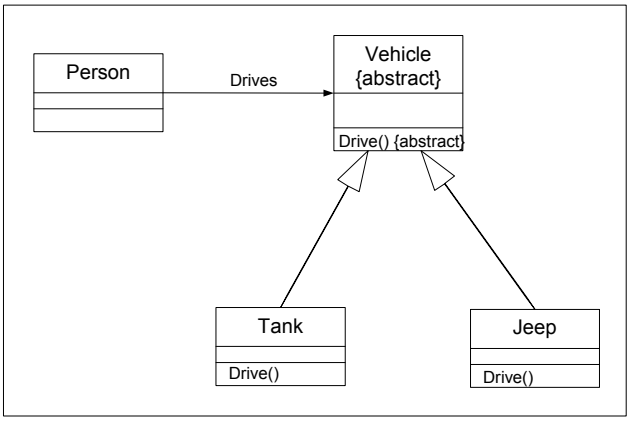

Figure 7 Inheritance and polymorphism in military systems context

Using UML, these modeling conventions, the list of stereotypes and tagged-values it is possible to model any kind of military system. Any stereotype name can be used and re-used as needed to characterize UML elements. Stereotype names should not be used as tagged-values.

3.3 Example of Utilization

This section presents three examples of UML class diagrams (Figures 8, 9, and 10) depicting complex military systems in the context of Capability-Based Planning. These examples do not express the exact reality of the present or the future of the Canadian military acquisition, nor are they complete. They are simple examples showing the use of the MAU-Profile. In these figures, a class is represented by a single rectangle for clarity purposes and stereotype names are used to give classes more specific semantic. The eventual parameterization of model elements would be achieved by the use of one or

many tagged values (tagged-values are not shown in these examples, for clarity purposes).

Figure 8 shows a strategic view depicting the architecture of an acquisition complex system. The top-down Capability-Based Planning is partly present in this diagram. High-level managers (like Ministers and Assistant Deputy Minister) define the Government Strategic Directions (recall Figure 1) that are used by military, industrial and academic organizations (performing acquisition) to identify Capability Gaps and then identify options. These architectures are composed of systems (people, processes, technologies and materiel) that form complex systems like SoS when they collaborate to accomplish their missions.

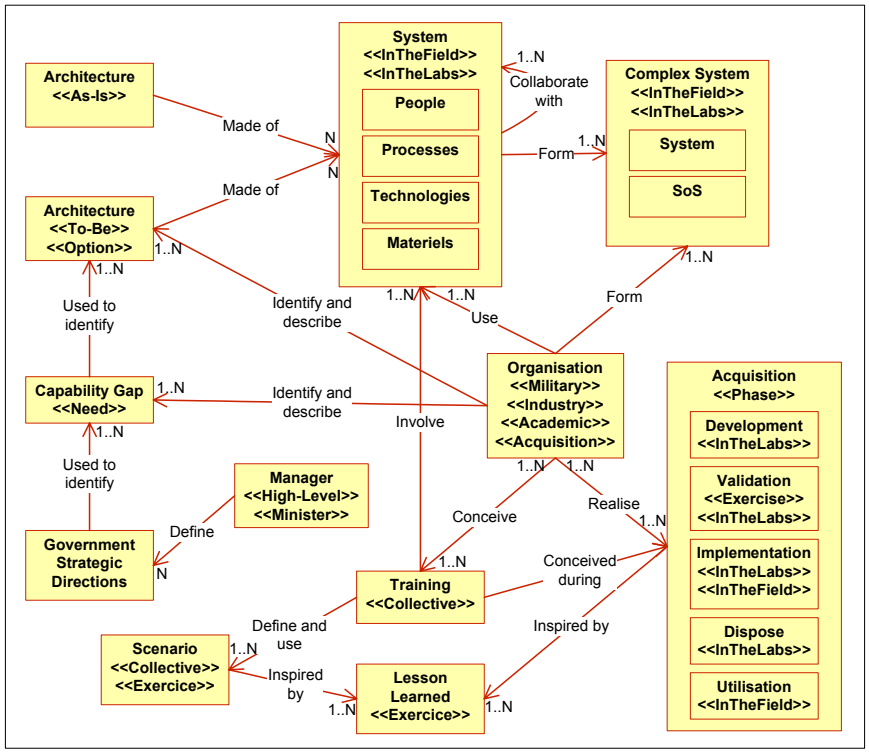

Figure 8. A hypothetical military acquisition complex system (strategic)

In this example, organizations performing the acquisition phases conceive the collective Training by defining and using Scenarios (or courses of actions); these are inspired by past Lessons Learned. The collective Training involves Systems; it is conceived during the Acquisition phases. This high-level (enterprise) diagram can and must be refined to further model SoS and its systems.

Figure 9 shows more details related to Organizations and collective Training elements (Figure 8). In this diagram, new stereotyped classes are used to model these elements with more details (and associations between them). Using a CASE tool, it should be possible to decompose one class into its many sub-classes using a hierarchical

structure. Figure 9 shows an example of this by decomposing the Scenario class (Figure 8) into other classes (that are contained in Course of Actions class).

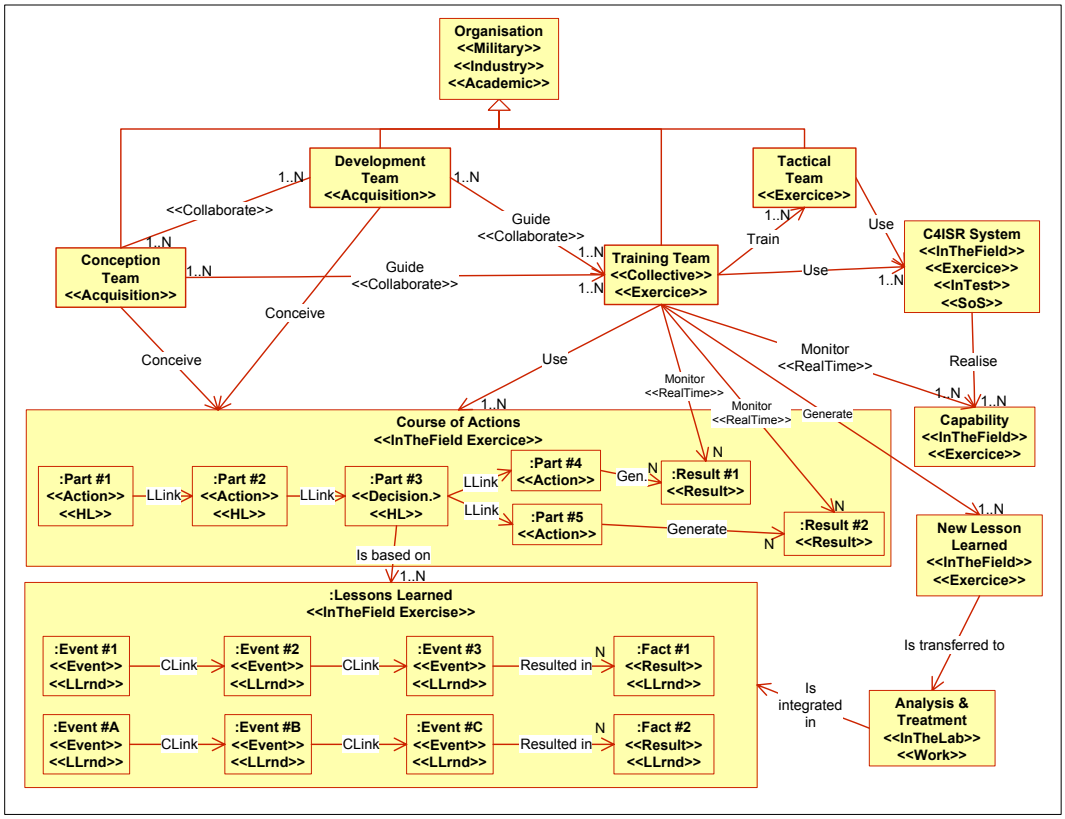

Figure 9. A hypothetical collective training architecture (strategic and operational)

Figure 9 gives a clear indication that Conception Team(s), Development Team(s), and Training Team(s) must collaborate to perform collective training; they form a SoS. The definition of the types of collaboration that must exist between teams could have been depicted in this figure by adding the appropriate tagged values to the associations. For instance, these tagged values could make reference to the standard(s) or frameworks that would have been used to standardize collaboration models and descriptions. These standards could have been defined by UML and the MAU-Profile. In this example, Conception Team(s) and Development Team(s) conceive the Scenarios (or Courses of Actions), and then Training Team(s) uses them to train Tactical Team(s). The conception of these Scenarios is inspired from the Lessons Learned from past experiences. When put into action, the collective training of Tactical team(s) involve the use of the developed SoS (here it is a C4ISR SoS) in field exercises. Delivered Capabilities are monitored in a real-time mode by Training Team(s). New Lessons Learned may then be generated, analyzed, treated and possibly kept for future uses.

In this simple example, Courses of Actions are made of a suite of high-level "Parts" (corresponding to sets of scoped actions) defining what should be achieved during the exercise. These are linked by logical link (LLink) associations. At some point

in time, a decision must be made for the exercise to be conducted toward the appropriate direction (Part 3 of Course of Actions). Lessons Learned inspire this decision from past experiences for better results. Note that the Lessons Learned takes the form of instances in this diagram. They are depicted as a concrete set of two past Lessons Learned involving events and final results.

When both Tactical Team(s) and C4ISR systems are fielded, they form a SoS of type D (recall Figure 1). Figure 10 gives more details of the C4ISR System from Figure 9. This class diagram shows that the C4ISR complex system is made of a Coordination Cell, a Head Quarter Communication Node and a Brigade HQ Cell. The Brigade HQ Cell is composed of a Plan Cell, an Ops Cell, a Command Cell and a HQ Server Cell. Further decompositions of these classes into new UML diagrams would provide more details of the architecture.

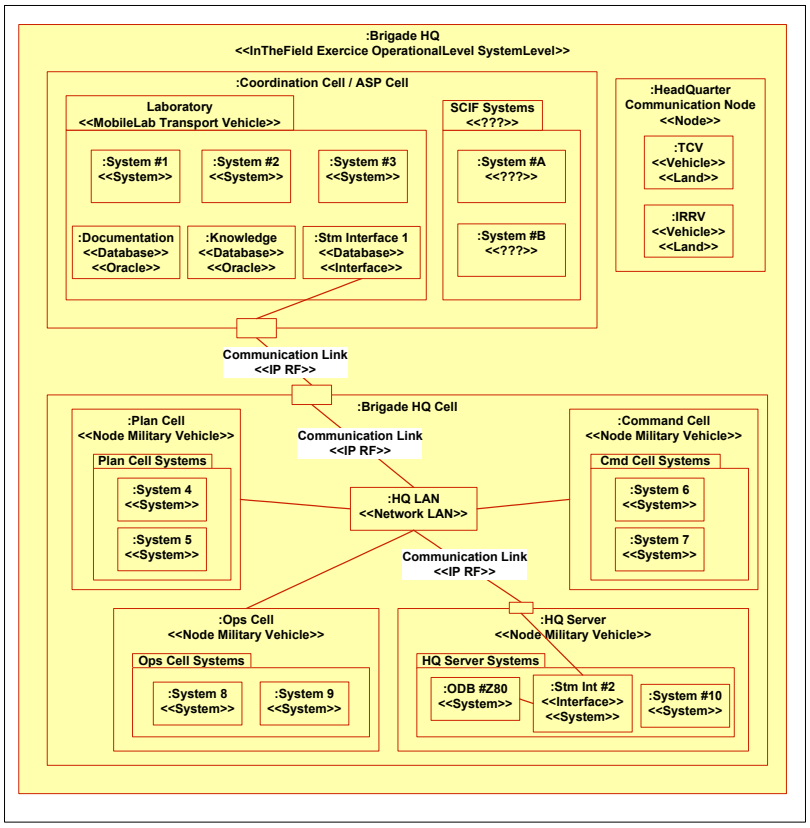

Figure 10. A hypothetical C4ISR architecture (operational and system)

Figure 10 makes use of the UML graphic element “package” to group model elements that are logically grouped in real life. The Laboratory package is, for instance, depicted as an assemblage of systems that are gathered together to accomplish specific missions or functions. Here, the stereotypes “MobileLab”, “Transport”, and “Vehicle” were added to this package to specify that it takes the specific form of a transport vehicle.

These three examples (Figures 8, 9, and 10) are related to the same hypothetical acquisition system, they should be part of the same architecture description. If associations between classes are not always shown in one class diagram, holism suggests that all associations between classes must be modeled and captured in the whole architecture description (within the CASE tool database). For instance, stereotyped associations or links between the 1- the Capability Gap class (Figure 8) for a mobile laboratory, 2- the detailed associated requirements (not shown), and 3- the Laboratory package (Figure 10) should be captured in the CASE tool database.

4 Conclusion

The work currently performed at DRDC Valcartier regarding the achievement of a UML profile for the description of military architectures in the context of Capability-Based Planning was described in this paper. Considering the transformation affecting military affairs and complexity associated to military systems, a review of contexts within which the profile will be used had to be done and presented. The management of military acquisition using the new Capability-Based Planning involves an enlargement of traditional System Engineering perspectives that was used in threat-based planning. An approach that considers mature engineering disciplines with new theories and that allows the production of holistic architectural descriptions at enterprise level appears as a potential solution in the context of Capability-Based Planning. In this context, architecture descriptions must allow the representation of any relevant concepts and links between them, no matter the domain or the project. The UML modeling language with the MAU-Profile was presented as a potential solution to this problematic.

There are many advantages of using our dynamic definition of system (and of SoS) and of using the structure of SoS that was presented (Figure 2). One of them is that some of the methodologies that will be identified and defined to address complex problems associated to a specific kind of SoS may potentially be re-used to address the same kind of problems for other kinds of SoS. The reason for this is that even if SoS are of different kinds, they are showing the same general characteristics. They are formed of independent collaborative systems, their behaviours are often non-linear, non-deterministic and they show hard to understand, to predict, and to control emergences. Actually, the Canadian acquisition system could be considered as forming an overall enterprise complex system that could be considered and optimized as a whole.

The language used to describe and model such business complex systems should be sufficiently generic to include any kind of concepts and links between them. The use of the UML and MAU-Profile will contribute to favour holism by providing a language that will allow the modeling (and the linking) of any aspects of complex systems at enterprise and project levels. It might contribute to lower risks that are associated to architectural changes by providing all stakeholders with complete, adapted and linked information. This linked information will contribute to make their decision processes more efficient and effective. The management and optimization of military acquisition projects at higher level (at CBP level; Figure 3) may be facilitated by the cross-exchange of any relevant information and knowledge among involved domains or projects. This should be

achieved by insuring completeness, coherence, consistency, and synchronization of the whole enterprise architecture description.

Often, human factors are not integral part of architecture descriptions. Even if people respect well-established processes and doctrine, they may give simple SoS strong nonlinear character. Such human-influenced systems that collaborate together still act as independent systems. They show interactions that give SoS emergent behaviours that are neither fully predictable nor easily repeatable (non-deterministic). They depend on initial conditions and environments that are often hard to measure and reproduce. The MAU-Profile should allow the modeling of such factors.

References

Araujo, T. and J. Caraca, 1999. Evaluating Complexity in Hierarchically Organized Systems, Proceedings of the III International Conference on Complexity in Economics, ISEG.

C2IEDM, 2005. All documents relative to C2IEDM can be found at MIP web site: http://www.mip-site.org.

CADM, 1998. C4ISR Core Architecture Data Model (CADM), Office of the Assistant Secretary of Defence, Department of Defence, US., Version 2.0.

Chen, P. and Clothier, J., 2003. Advancing Systems Engineering for Systems-of-Systems Challenges, Journal of The International Council on Systems Engineering, 6, 3, pp. 170-183.

Cook S.C., 2001. On the Acquisition of System-of-Systems, Proceedings of the INCOSE Annual Symposium, Melbourne, Australia.

Cook, S. C. and N. Sproles, 2000a. Synoptic Views of Defence Systems Development, in Proceedings SETE 2000, Brisbane, November 2000.

Cook, S. C. and N. Sproles, 2000b. Piecewise and Structural Views of Defence Systems Development, in Proceedings SETE 2000, Brisbane, November 2000.

Couture, M. and Duval, A.. 2005. On the Building of a UML Profile For the Description of Army Architectures In System of Systems Context. TR 2005-001 DRDC-Valcartier.

Corbett, D. W., 2004. Joint Requirements – It's Time To Pay The Piper. Canadian Forces College, MDS Research Project, CSC 30.

Couture, M., 2005. The MAU-Profile; a UML profile for the Description of Military Architectures. Document in preparation.

Davis, P. K., 2002. Analytic Architecture for Capabilities-Based Planning, Mission-System Analysis, and Transformation. RAND Corporation, MR-1513-OSD, ISBN: 0-8330-3155-4.

DND, 2004a. Department of National Defense of Canada, White Paper. http://www.forces.gc.ca/admpol/eng/doc/white_e.htm

DND, 2004b. Department of National Defense of Canada. Canadian Joint Task List and PRICIE related information can be found in web site: http://www.forces.gc.ca.

DOD-AF, 2003a. Definitions and Guidelines, DoD Architecture Framework Working Group, Department of Defence, US. Version 1.0, Volume I.

DOD-AF, 2003b. Product Description, DoD Architecture Framework Working Group, Department of Defence, US. Version 1.0, Volume II.

DOD-AF, 2003c. Appendices, DoD Architecture Framework Working Group, Department of Defence, US. Version 1.0, Volume III.

Hitchins, D. K., 2003. Advanced Systems Thinking, Engineering, and Management. Arthech House, Inc. ISBN 1-58053-619-0, 469 pages.

IEEE-1471, 2000. The Institute of Electrical and Electronics Engineers (IEEE), IEEE Std 1471 – IEEE Recommended Practice for Architectural Description of Software-Intensive Systems, New York, NY.

Keating, C., R. Rogers, U. Resit, D. Dryer, A. Sousa-Posa, R. Safford, W. Peterson, and G. Rabadi, 2003. System of Systems Engineering, Engineering Management Journal, 15, 3, pp 36,45.

Maier, M.W., 1998. Architecting Principles for Systems-of-Systems, Systems Engineering, 1, 4, pp. 267-284.

Michaud, S. M., 2004. Lost… but making good time: The Urgent Need for a Canadian Forces C^4ISR Framework. Canadian Forces College, MDS Research Project, CSC 30.

Moti, F, 2000. Engineering Systems Thinking and Systems Thinking, Systems Engineering, 3, 3.

OMG, 2004. All documents and specifications relative to UML, MOF, and other meta-models can be found at: http://www.omg.org

Sage, A. P. and C. D. Cuppan, 2001. On the System Engineering and Management of Systems of Systems and Federations of Systems, Information, Knowledge, and System Management, 2, 4, pp 325, 345.

SYSML, 2004. Many documents on SYSML can be found at: http://sysml.org

SYSENG, 2004. Many documents on SYSML can be found at: http://syseng.omg.org

UML, 2004. Unified Modeling Language™ (UML®), Version 1.5, (see OMG, 2004).

Glossary

Capability: The term capability is used in this paper as a property that expresses the ability of a SoS to perform a pattern of actions that will allow the accomplishment of a mission.

Capability-Based Planning (CBP): Paul Davis of the RAND Corporation (RAND, 2004) defines the CBP as: "*planning under uncertainty, to provide capabilities suitable for a wide range of modern-day challenges and circumstances, while working within an economic framework*" (Davis, 2002).

System: a system is made of people (person, group, association, organization) that use processes (doctrines, standards, methods), technologies (software, frameworks), and materiel (physical tools, vehicles, etc) to transform input into output within a specific context and under specific rules. The system provides functionalities the system must achieve when put in action. Systems are recursive in nature. Structures of system are allowed; system may contain or be composed of other systems. This definition considers systems as being dynamic rather than static. It involves all its contributing elements, which are acting in specific ways toward the realization of functions. Human factor is thus considered as being part of system's definition.

S**ystem of Systems (SoS)**: an assemblage of operationally independent systems that collaborate with each other in order to get the ability to achieve a mission-oriented set of actions that allow the realization of a global mission. This mission is understood and shared by all systems. A SoS may be dedicated to the realization of a few pre-determined missions. As for systems, SoS is an object of engineering in which collaborating systems need to be considered as part of a whole while evolving over time. SoS are recursive in nature, they are usually open as they are interacting with their environment(s), and they have life cycles that are determined by life cycles of their associated systems.

Bibliography

Mr. Couture received a B.Sc. degree in Physics and a M.Sc. in Physical Oceanography at the Université du Québec à Rimouski, Qc, Canada. After 8 years of M&S work at Fisheries and Ocean Canada, he completed a M.Sc. in Electrical Engineering at Laval University, Qc, Canada. In 2002, he joined Defence R&D Canada - Valcartier as a Defence Scientist in the System Engineering and Architecture (SEA) Group, which is part of the System of Systems (SoS) Section. His research interests are oriented toward the engineering and design of military architectures in the context of complex systems.