

A Review of the Proposed UML Profile for the Department of  
Defense Architecture Framework and Ministry of Defense  
Architecture Framework (UPDM)

Professor Dimitri Mavris

Co-Chair  
Architecture Working Group  
International Council on Systems Engineering

Director  
Aerospace Systems Design Laboratory  
School of Aerospace Engineering  
Georgia Institute of Technology

Dr. Patrick Biltgen  
Kristin Kelly  
Scott Brynildsen  
Andrew Howard

Reid Thomas  
William Engler  
Stephen Tongelidis  
Bradford Robertson

16 April 2007

## Introduction

This document reviews the proposed Unified Modeling Language (UML) Profile for the Department of Defense Architecture Framework (DoDAF) and the Ministry of Defense Architecture Framework (MODAF) - UPDM. As the primary goal of the review, this document will focus on examining UPDM to assess its compatibility with DoDAF and MODAF; compatibility with additional architecture frameworks is briefly addressed. Also included is an assessment for how UPDM enables systems engineering and subsequently how it follows industry standards and best practices. Finally, an extension is made to software engineering to examine the extensibility of the UPDM.

The proposal is in response to the Object Management Group’s (OMG) request for proposal dated September 16<sup>th</sup>, 2005 [1]. The merged UPDM submission, listed in reference 2, was a collaborative effort between Adaptive, Inc., ARTiSAN Software Tools, Ltd., International Business Machines, MEGA International, Telelogic AB, and THALES Group. Additionally, fourteen companies joined the submission as supporters of this proposal and the new UPDM framework. The proposed UPDM solution creates the framework and standards to allow software vendors to create tools enabling the creation of architecture frameworks in a model-driven architecture formulation.

## Compatibility with Defense Architecture Frameworks

The primary focus of this document is assessing the ability of the proposed UPDM standard to aid in the development and use of architectures within architecture frameworks. In its current state, UPDM builds upon OMG’s Model Driven Architecture (MDA) [3] to make required views and work products an extension of the created model. The top level approach that UPDM takes is illustrated in Figure 1 below. The supported views are indicated, along with the capability to add custom views and outside information and document. In addition, UPDM is designed to allow for the interchange of information. This is accomplished through a standard representation in XML Metadata Interchange (XMI).

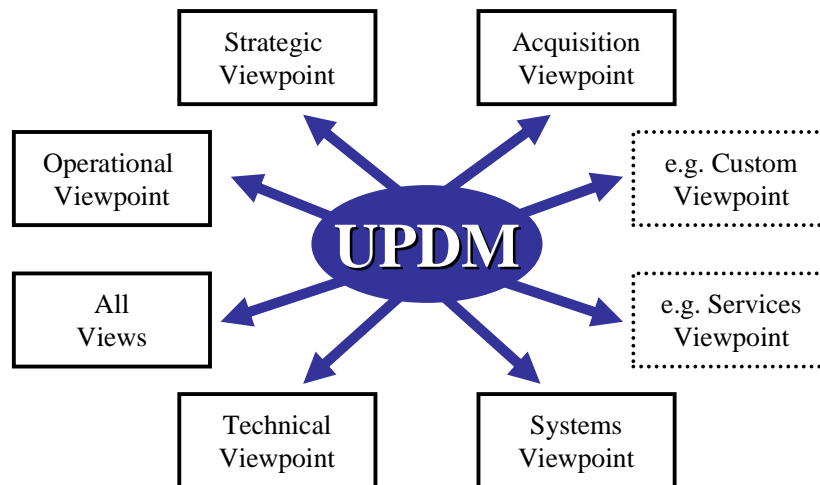


Figure 1: UPDM Viewpoint Support (Modified from reference 2)

## ***Department of Defense Architecture Framework***

The Department of Defense Architecture Framework (DoDAF), reference 4, is an evolution of the Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) architecture framework. The use of UML (Unified Modeling Language) is spelled out in reference 4, DoDAF Volume II: Product Descriptions. However, the UML examples in that document serve only as a reference point, and leave significant details to the tool vendor and end user to create. The purpose of the UPDM request for proposal [1], is to create a standard from which all software vendors and end users can work.

### **All Views (AV)**

All Views are a combination of overarching information common to the other three DoDAF views. The All Views products provide information that is pertinent to the entire architecture without capturing the architecture itself. All Views products provide the information that scopes the problem and provides context for the architecture [5].

#### ***One and Two***

The All Views consist of overview and summary information (AV-1) and an Integrated Dictionary (AV-2). These two views are used to provide an executive level summary of the architecture, and the assumptions and terms used in its creation. Because these are strictly text documents they are not easily represented in UML. UPDM handles these views by using its mechanism for referencing external links, and all other UPDM elements can indicate that they referenced an external taxonomy so this directs the viewer to the AV-2. A UML Class Diagram has the ability to capture the way things are put together. As a result, it could be used to represent the AV-1 in a more graphical form, though suboptimally.

			Good	Average	Poor
			●	◐	○
<b>All View</b>			AV-1		
			Overview and Summary Information		Integrated Dictionary
<b>UML</b>	Structure Diagram	Class Diagram	○		
		Composite Structure Diagram			
		Component Diagram			
		Deployment Diagram			
		Object Diagram			
		Package Diagram			
	Behavior Diagram	Activity Diagram			
		Use Case Diagram			
		State Machine Diagram			
	Interaction Diagram	Sequence Diagram			
		Communication Diagram			
		Interaction Overview Diagram			
		Timing Diagram			
		Model Library			
	<b>UML SysML Overlap</b>	Sequence Diagram			
State Machine Diagram					
Use Case Diagram					
Package Diagram					
<b>UML for SysML</b>	Block Definition Diagram				
	Internal Block Diagram				
<b>SysML Only</b>	Requirements Diagram				
	Parametric Diagram				
<b>UPDM</b>	External Reference	◐	◐		

Figure 2: All View Comparison

## **Operational Views (OV)**

The Operational Views are a series of products that lay out the tasks, activities, operational elements, and information exchange required for a given mission. The Operational Views combine graphical and tabular information to represent the architecture nodes and the communication between nodes, along with the type of information exchanged and the rate of exchange [5].

### ***High Level Operational Concept Graphic (OV-1)***

The OV-1 highlights the main operational nodes and other unique aspects of the operations. UPDM is able to model an OV-1 using several different UML diagrams. UPDM uses Class Diagrams, Use-Case Diagrams, and Interaction Diagrams to communicate effectively an OV-1. Class Diagrams are able to explain the actors and objects involved in the operation. Use-Case Diagrams will show what actions the actors and objects can have on the system. Finally, some version of an Interaction Diagram is needed to show the interactions between the actors and objects. With three different diagrams modeling a single view, a great deal of information can be conveyed. However, since one of the objectives of an OV-1 is to give a single image summary of the architecture, multiple diagrams may make it difficult to identify the most appropriate one for a summary view.

### ***Operational Node Connectivity Description (OV-2)***

The OV-2 describes the interactions and information exchange between the nodes. This is merely a description of the need to exchange information, not the physical connections. The diagram includes both the internal and external nodes as well as information about the necessary interactions. UML, through UPDM, can model an OV-2 using Class and Interaction Diagrams. The classes define the nodes and the Interaction Diagrams specify how the nodes interact with one another. SysML can provide further detail using a Block Definition Diagram.

### ***Operational Information Exchange Matrix (OV-3)***

The OV-3 is a detailed exchange of information between nodes. This includes what each node is expecting from every other node in the system and how the exchange will occur. The data itself is stored within the model library, but because the view is tabular in nature, it is not possible to represent this information within UML or SysML. The information required to generate this view is still included in the architecture model, but an external link or tool-specific solution must be used to generate the view.

### ***Organizational Relationships Chart (OV-4)***

The OV-4 relates different organizations together and illustrates the command relationships between them, as opposed to the business process relationships. UPDM uses UML Class Diagrams and Composite Structure Diagrams to represent an OV-4.

### ***Operational Activity Model (OV-5)***

The OV-5 is used to describe capabilities, Operational Activities, flows between activities, and flows to and from activities outside the scope of the architecture. This is the best tool for

describing the functional flow of activities to accomplish a mission or business goal. UPDM is able to model the connections in an OV-5 using a UML Use Case Diagram. The Use Case Diagram represents the high level organization used to complete the mission or business goal. It is also possible to represent this as an Activity Diagram.

### ***Operational Rules, State Transitions and Event-Trace Description (OV-6)***

The three OV-6 diagrams are used to capture constraints, operational conditions and sequences for the processes used in a mission or business process. The OV-6a describes operational or business rules as constraints on the architecture being examined. Due to the textual nature of an OV-6a, it is not possible to represent this within UML. This is one of the few views that the UPDM is unable to represent in any format or store significant information about. The OV-6b captures how an operational node or activity responds to various events by changing its state. UPDM utilized the UML State Diagram with StateMachines applying to an operational element. The OV-6c is a representation of the time-ordered information exchanges or interactions between nodes in a particular scenario. A UML Sequence Diagram is used to reflect the nodes that collaborate in each scenario.

### ***Logical Data Model (OV-7)***

The OV-7 defines the architecture domain's system data types and the relationship among the system data types. It is a key method for expressing the interoperability required. The OV-7 is modeled in UPDM using the UML Class Diagram to represent information elements inputs, outputs, and controls. The model can also be linked to an external source using UPDM's external reference mechanism to provide a database of information elements.

			OV-1	OV-2	OV-3	OV-4	OV-5	OV-6a	OV-6b	OV-6c	OV-7
			High-level Operational Graphic	Operational node connectivity description	Operational information Exchange Matrix	Organizational Relationships Chart	Operational Activity Model	Business rules that constrain operations	Business process responses to events	Traces actions in a scenario or sequence of events	Logical Data Model
			●	●	●	●	●	●	●	●	●
			○	○	○	○	○	○	○	○	○
<b>UML</b>	<b>Structure Diagram</b>	Class Diagram		●		●					●
		Composite Structure Diagram		●		○					
		Component Diagram				○					
		Deployment Diagram									
		Object Diagram									
		Package Diagram	●								
	<b>Behavior Diagram</b>	Activity Diagram					●		●		
		Use Case Diagram	●				●				
		State Machine Diagram							●		
	<b>Interaction Diagram</b>	Sequence Diagram							●	●	
		Communication Diagram									
		Interaction Overview Diagram									
		Timing Diagram									
		Model Library			●						
	<b>UML SysML Overlap</b>	Sequence Diagram									●
		State Machine Diagram							●		
		Use Case Diagram	●				●				
		Package Diagram									
<b>UML for SysML</b>	Block Definition Diagram		●								
	Internal Block Diagram										
<b>SysML Only</b>	Requirements Diagram										
	Parametric Diagram										
<b>UPDM</b>	External Reference			●			●				

Figure 3: Operational View Comparison

## **System Views (SV)**

The Systems Views provides information on the systems and interconnections which provide or support DoD functions. The information in Systems Views combines graphical and tabular data which are used to assign system resources to the Operational Views [5].

### ***Systems Interface Description (SV-1)***

An SV-1 depicts the systems needed to support operational nodes and the communication requirements between them necessary for architecture operations. In UPDM the SV-1 is represented by a Structure Diagram to represent a set of interconnected elements that exist to provide some piece of functionality. The type of Structure Diagram can be a Component Diagram, Deployment Diagram, or Composite Structure Diagram. It can also be represented in SysML using an Internal Block Diagram. The type of Diagram depends on the specific problem and is the architect's decision which type of Structure Diagram represents the system best.

### ***Systems Communication Description (SV-2)***

The SV-2 depicts information about the communications systems, communication links, and communications networks between the systems represented in the SV-1. UPDM is able to make use of UMLs Deployment Diagram to map pieces of a system to what is executing it or a Composite Diagrams to capture the interconnection of elements. If using SysML, an Internal Block Diagram can be employed to represent the communication paths to support each system

### ***Systems-Systems Matrix (SV-3)***

The SV-3 provides detailed information of the communication links between systems depicted in the SV-1 in a matrix form. Because this is a text matrix, UPDM references an external text source. Because the SV-3 is a more detailed version of an SV-1 the main interactions are captured by UML in the same manner as with the SV-1, but an external reference is required to fully define the SV-3.

### ***Systems Functionality Description (SV-4)***

The SV-4 describes the functions systems are performing and the flow of system data between those functions. UPDM uses a UML Activity Diagram to show the dynamic behavior of the various systems and capture the actions that make up the larger activities. UML dynamic diagrams can provide a timeline of systems needed to carry out the activities. A UML Object Diagram can be used to provide a snapshot of the relationships between systems and capture the data being exchanged.

### ***Operational Activity to Systems Functionality Traceability Matrix (SV-5)***

The SV-5 is a matrix that maps the relationship between Operational Activities and the System Functions that support each activity; it can also include the mapping of System Functions to Capabilities. The SV-5 is usually a spreadsheet matrix and no UML representation is needed for UPDM to depict this view.

***Systems Data Exchange Matrix (SV-6)***

The SV-6 is a matrix that captures the characteristics of the automated data exchanged between systems. This view does not have an associated UML diagram and UPDM makes use of its mechanism for external references. The UPDM Model Library includes templates that provide a starting point for creating the SV-6.

***Systems Performance Parameters Matrix (SV-7)***

The SV-7 is a matrix of the parameters used to quantitatively calculate the performance of systems which can be used to develop requirements and define specifications. The SV-7 is a text based matrix and for this reason UPDM does not use a UML view, instead it uses the UPDM mechanisms for external references.

***Systems Evolution Description (SV-8)***

The SV-8 is used to describe plans for modernizing system functions over time and when linked with other evolution products describes the total evolution plan or transition plan for the architecture. UPDM represents this view by using its standard external link mechanism and sends information to an external scheduling tool.

***Systems Technology Forecasts (SV-9)***

The SV-9 is a description of emerging technologies that are applicable to the architecture, and provides details about their availability and possible impacts. UPDM represents the SV-9 with a table and matrix that is exported from model elements.

***Systems Rules Model (SV-10a)***

The SV-10a explains the rules that govern the implementation of system in the architecture. UPDM uses text rules exported from the model to represent the SV-10a.

***State Transition Description (SV-10b)***

The SV-10b describes the System as it changes in response to various events. UPDM represents the SV-10b as a StateMachine Diagram. The StateMachine Diagram represents captures the internal transitions of an element given a stimulus and is well suited to represent and SV-10b.

***Event-Trace Description (SV-10c)***

The SV-10c provides the sequence of actions and exchanges and flows between systems. UPDM represents this view with a UML Sequence Diagram. The Sequence Diagram emphasizes the type and order of messages passed between systems and is a good representation of an SV-10c.

***Physical Schema (SV-11)***

The SV-11 represents the structure of architecture's domain data types and relationship of system data types. To capture these data types and relationships UPDM uses a UML Class Diagram. The Class Diagram is designed to represent the different types of elements that make up a system and can represent the different data types and their relationships as needed in the SV-11.

Good	Average	Poor
●	◐	○

Systems View		SV-1	SV-2	SV-3	SV-4	SV-5	SV-6	SV-7	SV-8	SV-9	SV-10a	SV-10b	SV-10c	SV-11		
		Systems Interface Description	Systems Communications Description	Systems-S systems Matrix	Systems Functionality Description	Operational Activity Systems Functionality Traceability Matrix	Systems Data Exchange Diagram	Systems Performance Parameters Matrix	Systems Evolution Description	Systems Technology Forecasts	Systems Rules Model	Systems State Transition Diagram	Systems Event Trace	Physical Schema		
UML	Structure Diagram	Class Diagram													●	
		Composite Structure Diagram	●	●												
		Component Diagram	●													
		Deployment Diagram	●	●												
		Object Diagram				●										
		Package Diagram								○						
	Behavior Diagram	Activity Diagram				●										
		Use Cads Diagram							○							
		State Machine Diagram										●				
	Interaction Diagram	Sequence Diagram												●		
		Communication Diagram														
		Interaction Overview Diagram														
		Timing Diagram														
		Model Library						○								
	UML SysML Overlap	Sequence Diagram												●		
State Machine Diagram												●				
Use Case Diagram																
Package Diagram																
UML for SysML	Block Definition Diagram															
	Internal Block Diagram	◐	◐													
SysML Only	Requirements Diagram															
	Parametric Diagram				◐											
UPDM	External Reference			●		●	●	●	●	●	●					

Figure 4: Systems View Comparison

### Technical Standards Views (TV)

The Technical Standards Views create the set of rules for the arrangement, interaction, and interdependence of system elements [5].

### ***One and Two***

The Technical Views consist of a Technical Standards Profile (TV-1) which documents the constraints and technical standards used in the architecture. The other TV is a Technology Standard Forecast (TV-2) which captures expected changes in applicable standards mentioned in the TV-1. Both of these views are text based and do not need to be represented in UML. UPDM handles these by using it's mechanism for external references.

			Good	Average	Poor
			●	◐	○
<b>Technical View</b>			TV-1		
			Technical Standards Profile		
			TV-2		
			Technology Standard Forecast		
<b>UML</b>	<b>Structure Diagram</b>	Class Diagram			
		Composite Structure Diagram			
		Component Diagram			
		Deployment Diagram			
		Object Diagram			
		Package Diagram			
	<b>Behavior Diagram</b>	Activity Diagram			
		Use Case Diagram			
		State Machine Diagram			
	<b>Interaction Diagram</b>	Sequence Diagram			
		Communication Diagram			
		Interaction Overview Diagram			
		Timing Diagram			
		Model Library			
	<b>UML SysML Overlap</b>	Sequence Diagram			
State Machine Diagram					
Use Case Diagram					
Package Diagram					
<b>UML for SysML</b>	Block Definition Diagram				
	Internal Block Diagram				
<b>SysML Only</b>	Requirements Diagram				
	Parametric Diagram				
<b>UPDM</b>	External Reference	◐	◐		

Figure 5: Technical View Comparison

## ***Ministry of Defense Architecture Framework (MODAF)***

The Ministry of Defense Architecture Framework (MODAF) from the United Kingdom builds upon the views found in DoDAF to extend the architecture framework to handle Ministry of Defense (MOD) processes and lifecycles. Specifically, MODAF adds Acquisition Views to further describe the lower levels of the implementation of the architecture, and Strategic Views to better describe the overarching views for the entire architecture [6].

### **Acquisition Views**

Acquisition Views have been added to the MODAF to capture programmatic details. These include the dependencies between projects and capability integration and are used to guide the acquisition and fielding process [6].

#### ***SoS Acquisition Clusters (AcV-1)***

The AcV-1 view describes a collection of systems that work together to create acquisition clusters. Within UPDM, this view can be modeled using a UML composite structure diagram to describe the relationships between systems to provide a greater functionality. The option exists for using a UML Deployment Diagram as well; this diagram shows how the architectures systems are executed and assigned.

#### ***SoS Acquisition Program (AcV-2)***

AcV-2 diagrams a project along a timeline to better show the phases and other aspects of the project. There is no UML diagram that can effectively recreate this diagram. UPDM can create external links to tools that make timelines and other views necessary to represent the AcV-2. The possibly exists to use a Timing Diagram to capture the timeline of the project, the Timing Diagram is intended to represent the timing specifications for messages between systems and would have to be modified or used creatively to show the timeline of a project.

			Good	Average	Poor
			●	◐	○
<b>Acquisition View</b>			AcV-1		
			SoS Acquisition Clusters		SoS Acquisition Program
<b>UML</b>	<b>Structure Diagram</b>	Class Diagram			
		Composite Structure Diagram	◐		
		Component Diagram			
		Deployment Diagram	◐		
		Object Diagram			
		Package Diagram			
	<b>Behavior Diagram</b>	Activity Diagram			
		Use Case Diagram			
		State Machine Diagram			
	<b>Interaction Diagram</b>	Sequence Diagram			
		Communication Diagram			
		Interaction Overview Diagram			
		Timing Diagram			
		Model Library			○
	<b>UML SysML Overlap</b>		Sequence Diagram		
State Machine Diagram					
Use Case Diagram					
Package Diagram					
<b>UML for SysML</b>		Block Definition Diagram			
		Internal Block Diagram			
<b>SysML Only</b>		Requirements Diagram			
		Parametric Diagram			
<b>UPDM</b>		External Reference			◐

Figure 6: Acquisition View Comparison

## **Strategic Views**

The Strategic Views are added to MODAF to augment the DoDAF views and support the capability management process at the top levels [6].

### ***Capability Vision (StV-1)***

The StV-1 defines the context of a group of capabilities in a way that is understandable by a general audience. StV-1s are usually text based documents and descriptions and are referenced by UPDM using its external reference mechanism.

### ***Capability Taxonomy (StV-2)***

The StV-2 is a structured list of capabilities and capability functions that are required to provide a capability. The StV-2 is a more structured representation of the StV-1. UPDM uses a Composite Structure Diagram to represent the StV-2 and the complex interactions among capability functions to provide a capability. A Deployment Diagram might also be used to show the configuration of capability functions to provide capabilities.

### ***Capability Phasing (StV-3)***

The StV-3 provides a representation of the capability at different times and shows anticipated capabilities as well. This is a text document or table and no UML document is needed so UPDM uses its external reference mechanism to incorporate this view.

### ***Capability Clusters (StV-4)***

StV-5 show groups of capabilities and the dependencies and interactions between different capabilities. UPDM uses a UML Composite Structure Diagram to show the functional dependencies between capabilities. An Interaction Overview Diagram could be used as well because it is designed to emphasize which elements are involved in performing an activity and can used to show which capabilities depend on others.

### ***Capability to Systems Deployment Mapping (StV-5)***

The StV-5 shows the mapping of systems to the capabilities that they support for a particular time period. The StV-5 is represented as a matrix and no UML is needed so UPDM references an external document.

### ***Capability Function to Operational Activity Mapping (StV-6)***

The StV-6 maps the functions to complete each capability to the high level operational activities. No formal UML diagram is needed and UPDM will reference a matrix that contains the relationship between Strategic Capabilities and Operational Activities.

			StV-1	StV-2	StV-3	StV-4	StV-5	StV-6
<b>Strategic View</b>			Capability Vision	Capability Taxonomy	Capability Phasing	Capability Clusters	Capability to Systems Deployment Mapping	Capability Function to Operational Activity Mapping
<b>UML</b>	Structure Diagram	Class Diagram						
		Composite Structure Diagram		●		◐		
		Component Diagram						
		Deployment Diagram		◐				
		Object Diagram						
		Package Diagram						
	Behavior Diagram	Activity Diagram						
		Use Case Diagram						
		State Machine Diagram						
	Interaction Diagram	Sequence Diagram						
		Communication Diagram						
		Interaction Overview Diagram				●		
		Timing Diagram						
		Model Library						
	<b>UML SysML Overlap</b>	Sequence Diagram						
State Machine Diagram								
Use Case Diagram								
Package Diagram								
<b>UML for SysML</b>	Block Definition Diagram							
	Internal Block Diagram							
<b>SysML Only</b>	Requirements Diagram							
	Parametric Diagram							
<b>UPDM</b>	External Reference	◐		◐		◐	◐	

Figure 7: Strategic View Comparison

## ***UPDM Summary for DoDAF/MODAF***

Examining the diagrams used to model DoDAF and MODAF through UPDM reveals certain diagrams appear more often than others. The various UML structure and behavior diagrams are used frequently, while only one type of interaction diagram is used, the sequence diagram. Because of the requirement for several tabular and textual views within DoDAF and MODAF, external references and tools-specific implementations are needed. This is even more evident in the System Views, Technical Views, and All Views. While the System Views use other types of diagrams to help create them, the Technical Views and All Views are very dependent on other non-UML formats. The main concern here would be widely varying formats between the implementations preventing an engineer skilled in reading the views created by one tool from reading the views of another tool. This concern cannot be addressed here and is beyond the scope of any UML profile.

In order for UPDM to effectively define a diagrammatic modeling language for DoDAF and MODAF, it must be possible to create each graphical view using UPDM. The Operational Views and System Views require the most diagrams to depict them, while the Technical Views and All Views require relatively few diagrams.

Evaluating which diagrams are used the most to model MODAF shows a small number of diagrams are utilized for multiple work products. The most often used diagrams are composite diagrams, deployment diagrams, and external references. The rest either do not appear or appear relatively few times.

The same scarcity is seen in how many diagrams are used to create each of the additional MODAF views. Unlike DoDAF, which may use several diagrams to model each of the views, the additional MODAF views are made of only a few diagrams each.

## ***Additional Architecture Frameworks***

The UPDM proposal focuses on applications to DoDAF and MODAF. However, additional architecture frameworks, such as the NATO Architecture Framework (NAF), are also relevant to UPDM. While it is difficult to accurately assess all the needs of future architecture frameworks, the flexibility of UPDM coupled with the links to external data sources provides the capability to handle foreseeable views or work products.

## **Conformity to Systems Engineering Best Practices**

The implementation and usefulness of UPDM will depend greatly on how well it merges into a systems engineering process. In order to fit well, it is important that the standard use terminology and tools that already are utilized in the current systems engineering best practices. One of the problems with such a statement is the large number and wide variety of defined systems engineering best practices documents. A brief survey of the best practices documents produced by

a cross section of professional societies is presented here. While the latest versions of the appropriate documents were used whenever available, it can often take some time for new methods and tools to penetrate different groups. This caveat should be kept in mind, particularly with regards to SysML.

### ***International Council on Systems Engineering***

As a group whose sole purpose is the promotion of systems engineering and systems engineers, the best practices presented by the International Council on Systems Engineering (INCOSE) are among the most important to meet. The third version of the INCOSE Systems Engineering Handbook, released in June 2006 serves as the parent document for INCOSE best practices [7]. At less than a year old, it is expected that this document would include some of the most current tools and methods available to systems engineers.

While the section for discussion of tools and methods is constrained, it is encouraging that SysML is included as part of Requirements Management section on page 7.7. Unfortunately, the discussion is limited to a brief overview and does not currently describe how SysML fits into the Systems Engineering (SE) process as a whole. This should not be seen as a criticism of the Handbook, but rather as a sign that SysML is still finding its place in SE processes and best practices. With time, indications are that it is likely that SysML will grow in popularity and will be utilized throughout the SE process. UML is also mentioned but only as the basis of SysML. This suggests that UML is not viewed widely as a tool for SE, despite being formalized nearly 10 years ago. While there is a certain amount of overlap between SysML and UML, the full set of UML diagrams are utilized for both UPDM Compliance Level 0 and Level 1. This disjoint suggests the possibility that it will be necessary for systems engineers to be as well versed in UML as they are likely to become in SysML in order to use UPDM as defined. It may be that UPDM is the driving factor for the military side of systems engineering to learn and use UML and SysML extensively, but for the time being, based on their limited presentation, neither language is presented as a common skill by the typical member of INCOSE.

The systems engineering processes presented in the Handbook already matches up very well to an implementation of both UPDM for the architecture and UML/SysML for the other aspects of engineering. Additionally, where defined, the terminology used by the Handbook is consistent with that used in the UPDM.

The two primary elements from INCOSE are that UPDM does conform to systems engineering best practices, however, a relative lack of knowledge and practice among the systems engineering community may limit the effectiveness of the broader application of UPDM. It should be noted, however, that the number of papers on UML, SysML, Object Oriented Engineering, as well as the various architecture frameworks suggests that these tools are beginning to saturate the field and will be rolled into best practices and standards soon.

## ***Office of the Secretary of Defense***

For the acquisition of military systems, the policy and practices defined by the Office of the Secretary of Defense (OSD) exist as underlying customer requirements, which are necessary in order to secure a military contract. Since UPDM is intended for use on military acquisition programs (through the use of the DoDAF and MODAF), the OSD standards for systems engineering are extremely important. While there are several documents that discuss systems engineering, one of the most complete is the Defense Acquisition Guidebook [8]. Surprisingly, the DoDAF is not mentioned in the Systems Engineering section, but rather in the section describing the Global Information Grid. Despite the limited discussion, the processes presented support several of the views specified by the DoDAF. The Department of Defense Systems of Systems Engineering Guidebook discusses the use of DoDAF as a method for fulfilling the requirement of CJCSI 6212.01d [9,10]. For any standardized method for describing DoDAF and MODAF views to become popular or useful to a systems engineer, DoDAF and MODAF need to be presented as a useful part of the systems engineering process and not just a requirement that must be filled. Having a defined standard such as the UPDM might encourage this, but it is up to systems engineers themselves to find and formalize its place.

As part of the specification, DoDAF gives incomplete suggested implementations of its views in UML [4, 5]. It seems reasonable to expect that a full implementation that still complies with the intent and spirit of the diagrams would be supported by OSD as a whole. The extension of the views into SysML appear to be a logical next step, but direct information supporting this as the view of the OSD has been difficult to locate. While clearly stated as unofficial and not representative of government policy, experts at the Defense Acquisition University have stated that for architectural modeling standards like SysML are the best hope, once widely accepted [11]. This appears to be a common belief both inside and outside of the Department of Defense, encouraging the acceptance of UPDM in order to allow modeling standards to become widely accepted. With wide acceptance, it seems clear that the Department of Defense would likely encourage standardization using the UPDM.

## ***Institute of Electrical and Electronic Engineers***

The Institute of Electrical and Electronic Engineers (IEEE) is the world's largest professional society. Combined with the amount of systems engineering required in electrical engineering, that has led to IEEE being one of the leading bodies for developing systems engineering standards. IEEE 1471 is the IEEE standard for describing the architectures of software intensive systems [12]. Much of the terminology definitions within this standard were used as the basis for DoDAF and, in turn, parts of MODAF. As a result, it also appears to have served as the dictionary for many of the terms within the UPDM. This dependency chain suggests that the language and basic concepts within the UPDM are in line with IEEE. It is difficult to prove this explicitly because the nuances in the definitions that often vary from source to source are not captured in a terminology section of the UPDM. IEEE 1220 defines the standard for the systems engineering process [13]. While this process has not been updated since 1998, several of the products described within are utilized in DoDAF. One would expect an updated version to further

overlap with more current systems engineering processes. While not entirely compatible, the specification defined in the UPDM does not conflict with the needs and processes shown in IEEE 1220. From a systems engineering standpoint, UPDM is expected to comply and assist in the implementation of the appropriate applicable IEEE standards and best practices.

### ***International Standards Organization***

The International Organization for Standardization (ISO) is the world's foremost body for technical standards. Within ISO, the systems engineering group is part of the information technology technical committee. Many of their standards are based on those created by other groups. For example, both the IEEE 1471 and the UML 1.4.2 Specification have been released as ISO standards [14]. Of the standards presented by the group, ISO 15288, ISO 15289, and ISO 19760 are the most appropriate for the systems engineering process. ISO 15288 defines the processes required for systems engineering over the product life cycle [15]. ISO 15289 describes each of the products created as part of the processes described in ISO 15288 [16]. ISO 19760 is a guide to implementing ISO 15288 [17]. These three standards define the processes and products necessary in generic terms without much reference to implementation. While they share some overlap with the processes associated with DoDAF and MODAF that are enabled by UPDM, much of what is described is beyond that scope. The other related standards are primarily intended for use for software engineering and require a certain amount of translation into systems engineering. The use of UML and SysML fit in well with some of these standards. If the software engineering standards are found to be useful to an organization's systems engineering processes then it would enable the process to reuse the knowledge of the two languages.

### ***Object Management Group***

The Object Management Group (OMG) is responsible for the original request for proposal for UPDM [1]. OMG is an open membership, not-for-profit, internal computer industry consortium. OMG creates integration standards for a wide range of technologies and industries [18]. In this context, OMG is providing specifications that enable systems engineering through the use of vendor created products, adhering to the UPDM specifications. The request for proposal released by OMG captures systems engineering best practices through reference to external standards such as IEEE 1471, and thus a submission meeting the request for proposal requirements will follow, and likely create, systems engineering best practices.

### ***Business Process Modeling Notation***

Another standard from OMG is the Business Process Modeling Notation (BPMN). Developed in 2006 prior to UPDM, BPMN shares several common features with UPDM such as XMI for data storage and UML diagrams to represent business processes [19]. A specific mapping between BPMN and UPDM is given in section twelve of the proposed standard [2]. With these features, BPMN could be considered to be complementary to UPDM, when DoDAF or MODAF views do not provide adequate capabilities for modeling processes or are not required explicitly. However, the lack of SysML and a limited extensibility for outside information prohibit the BPMN

standard from capturing the full range of elements within DoDAF and MODAF. Thus while BPMN shares some common elements with DoDAF and MODAF, it should not be viewed as an isolated alternative for modeling architecture frameworks.

### ***Systems Engineering as a Discipline***

Systems engineering is an ever evolving field. INCOSE, as the recognized governing body, has developed a series of best practices and published them as the Systems Engineering Handbook [7]. Through compliance with those best practices, UPDM should aid the evolution of systems engineering as a recognized discipline. A wide acceptance of UPDM would then expand the uses of systems engineering in turn. A wide acceptance may also transform defense architecture frameworks into design and engineering tools rather than a requirement. The overall impact of a larger community of systems engineers following the INCOSE standards specified by means of modeling languages and architecture frameworks could lead to broader acceptance of systems engineering methods and best practices.

### **The Assessment of UPDM**

The overall assessment of the proposed UPDM is that it is capable of supporting DoDAF, MODAF and additional architecture frameworks. It is the domain of OMG to determine if the proposed submission [2] is fully compliant with the mandatory requirements from the request for proposal, and addresses in full or partial compliance all of the elements within the original request for proposal [1]. However, it should be noted that one possible exception to compliance is the lack of representation of architectural patterns, which was called for in the OMG request for proposal.

As shown above in the evaluation figures, UPDM has sufficient coverage for all pertinent work products either directly through UML and SysML or through links to external models. UPDM also addresses additional concerns such as support for additional views, analysis of alternative models of an architecture, cross queries and analysis across related models, overlap and interface between different system architectures, scalability, relationship between architecture models and domain models, support for development methodologies, and the traceability of architecture models to requirements [2].

### ***Ease of Learning***

One of the largest hurdles for UPDM is the adoption of new software and tools and the learning curves associated with each. Those familiar with UML, SysML, or BPMN will find the transition straight forward. However, those users new to these elements may find the transition much more difficult. This is especially worth noting in the larger systems engineering context. For this to be widely accepted within the community, a familiarity level must be reached by a critical mass of all practitioners. Simply understanding systems engineering, or more specifically DoDAF/MODAF may not be sufficient for greater acceptance. It is largely left to the software vendors to provide examples and documentation for their specific tools. However, another approach may be the creation of an overarching guideline and reference manual, such as the DoDAF volume I [5]. Such a manual would provide official documentation for those wishing to

properly use UML for describing architecture frameworks without needing to wade through the more precise and less understandable language necessary in a specification.

### ***Implementation***

Specific implementations of UPDM are left to individual software vendors. The purpose of UPDM is to provide a common framework from which software tools to support system architectures emerge. The primary elements that each software vendor must address are portability, reusability, interoperability, and the ability to interface with external tools. The product to come from the individual software vendors will be available to industry for evaluation, and the ultimate selection will not relate to the specifics of UPDM, but to how the individual entities address the visual tools and user interactions [2]. Another item that will distinguish individual software applications will be the success and ease of model implementation and model querying. The ability and ease to generate DoDAF and MODAF work products through queries to the model will likely determine the overall effectiveness of the software.

### ***Extension to Software Engineering***

A consistent point of contention is the difference between systems engineering architecture frameworks, and software engineering architecture frameworks. Since the UPDM is built upon UML, a key tool for software engineering architectures, a logical assumption would be that UPDM extends to serve software engineering. However, this is not necessarily the case. As noted in reference [20], “The DoDAF and current software architecture approaches have been developed separately, by different organizations, with different purposes, and with little overlap.” In addition, systems engineering typically takes a more top level view than software engineering. Despite this, the capability to query a model and generate work products for DoDAF and MODAF is similar to the ability of current software engineering tools to create actual code from UML diagrams. Furthermore, the incorporation of the full UML 2.1 as the basis for compliance level 0 in UPDM creates a common base with software engineering. It will ultimately be up to the software vendors to decide if their products contain a full suite of capabilities to handle software and systems architecting.

### ***Broader Goals of UPDM***

The UPDM submission provides a comprehensive approach to exploiting a model driven architecture for application to DoDAF and MODAF. Current practices invariably feature one model for architecting, and another for creating work products. Fully realizing the benefits of a model driven architecture requires a flexible modeling environment that allows queries to the model to build the required work products. The process of creating work products should flow naturally as a result of the engineering process. Any proposed tools intending to link the two must fit into a unified methodology. The UPDM submission features these abilities, in addition to capturing systems engineering best practices and adhering to specifications from OMG and IEEE as defined in the associated RFP [1].

## Conclusion

The overall assessment of the proposed UPDM submission is that it is capable of supporting DoDAF, MODAF and additional architecture frameworks. Furthermore, the use of SysML in the submission should promote the use of these architecture frameworks in systems engineering. The proposed submission was in response to an OMG request for proposal [1] and this review of the UPDM submission is based on both the suitability of the submission as a modeling language for INCOSE systems engineering standards as well as a combination of considerations related to the broader systems engineering environment. The final conclusions that can be drawn for the UPDM are summarized below:

- The proposed UPDM submission provides the necessary elements to model and capture the different views of DoDAF and MODAF as well as providing the ability to create and add custom views.
- While it is the domain of the OMG to determine whether the UPDM submission meets all of the mandatory requirements, and partially or fully addresses all of the optional requirements, it should be noted that one possible exception to compliance is the lack of representation of architectural patterns, which was called for in the OMG request for proposal.
- The UPDM can be an enabler for the model driven architecture paradigm by creating the profile so that work products for DoDAF, MODAF, and systems engineering are not created independently, but can be managed automatically by querying the model.
- The proposed UPDM submission follows accepted best practices and standards as covered by INCOSE and IEEE, along with the specified OMG modeling specifications.
- The INCOSE Systems Engineering Handbook, released in June 2006, includes SysML as part of Requirements Management section (see page 7.7). However, the discussion is limited to a brief overview and does not currently describe how SysML fits into the Systems Engineering process as a whole.
- UPDM may allow for a closer relationship between systems engineering and software engineering through the use of the UML and SysML baseline, along with providing the query abilities to facilitate the needs of both communities.
- The success of UPDM will ultimately depend on the systems engineering processes and software products that are created around the UPDM standard, the features that these products possess, and the ease of use for the practitioners.

## References

1. Object Management Group, Request for Proposal, “UML Profile for DoDAF/MODAF (UPDM).” Document number dtc/05-09-12. September 16<sup>th</sup>, 2005.

2. “UML Profile for the Department of Defense Architecture Framework (DoDAF) and the Ministry of Defense Architecture Framework (MODAF).” OMG Document c4i/2007-02-01. March 2007.
3. Soley, Richard, OMG Staff Strategy Group. “Model Driven Architecture.” Object Management Group, White paper draft 3.2, November 27<sup>th</sup>, 2000.
4. Department of Defense Architecture Framework Working Group, “Department of Defense Architecture Framework Version 1.0.” Volume II: Product Descriptions, February 9<sup>th</sup>, 2004.
5. Department of Defense Architecture Framework Working Group, “Department of Defense Architecture Framework Version 1.0.” Volume I: Definitions and Guidelines, February 9<sup>th</sup>, 2004.
6. The Ministry of Defense Architecture Framework. Version 1.1, April 2007.  
<http://www.modaf.org.uk/>, Accessed April 13<sup>th</sup>, 2007.
7. Systems Engineering Handbook, Ed. Cecilia Haskins, v3, INCOSE, June 2006.
8. Defense Acquisition Guidebook, Defense Acquisition University, December, 2004.  
Retrieved from <http://akss.dau.mil/dag/> April 4, 2007.
9. Department of Defense Systems of Systems Engineering Guidebook, Version 0.9, Director, Systems and Software Engineering, Deputy Under Secretary of Defense (Acquisition and Technology) Office of the Under Secretary of Defense (Acquisition, Technology and Logistics), December 2006.
10. Chairman of the Joint Chiefs of Staff Instruction 6212.01D: Interoperability and Supportability of Information Technology and National Security Systems. Office of the Chairman of the Joint Chiefs of Staff, March 2006.
11. Ask a Professor – How do you cost estimate Model Driven Architecture Software?, Defense Acquisition University, October 2006. Retrieved from  
<http://akss.dau.mil/askaprof-akss/qdetail2.aspx?cgiSubjectAreaID=2&cgiQuestionID=1789>  
1 April 10, 2007.
12. IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems, Institute of Electrical and Electronic Engineers, September 2000.
13. IEEE Std 1220-1998 IEEE Standard for Application and Management of the Systems Engineering Process, Institute of Electrical and Electronic Engineers, 1998.
14. ISO/IEC 19501:2004 Open Distributed Processing - Unified Modeling Language (UML) Version 1.4.2, International Organization for Standardization, April 2005.

15. ISO/IEC 15288:2002 Systems engineering - System life cycle processes, International Organization for Standardization, November 2002.
16. ISO/IEC 15289:2006 Systems and software engineering - Content of systems and software life cycle process information products, International Organization for Standardization, April 2006.
17. ISO/IEC TR 19760:2003 Systems engineering - A guide for the application of ISO/IEC 15288 (System life cycle processes), International Organization for Standardization, November 2003.
18. The Object Management Group, <http://www.omg.org/>, Accessed April 13<sup>th</sup>, 2007.
19. Object Modeling Group, “Business Process Modeling Notation (BPMN) Specification.” Final adopted Specification, dtc/06-02-01. February 2006.
20. Wood, W. G., Barbacci, M., Clements, P., Palmquist, S., Ang, H., Bernhardt, L., Dandashi, F., Emery, D., Sheard, S., Uzzle, L., Weiler, J., Krummenoehl, A., “DoD Architecture Framework and Software Architecture Workshop Report.” Architecture Tradeoff Analysis Initiative, March 2003.

## Vita

Dr. Dimitri Mavris is the Boeing Associate Professor for Advanced Aerospace Systems Analysis at Georgia Tech and Co-director of NASA’s University Research Engineering, and Technology Institute in Aeropropulsion and Power. He received his doctorate from Georgia Institute of Technology in 1988 in the field of Aerospace Engineering. He joined the faculty of the School of Aerospace Engineering in 1995. Dr. Mavris is the Director of the Aerospace Systems Design Laboratory, which has 15 research engineers, 151 M.S. and Ph.D. students, and 60 undergraduate students with an annual budget of \$6M in sponsored research. Dr. Mavris has actively pursued vigorous ties between the academic and industrial communities. He has 228 refereed and conference publications. Dr. Mavris has pioneered the Robust Design Simulation methodology as well as a method for Technology Impact Forecasting (TIF), which lead to the development of Technology Identification, Evaluation & Selection (TIES) methodology. All of which is in support of decision making for complex system design. Dr. Mavris has made several significant accomplishments in the area of multi-disciplinary design, particularly in advanced probabilistic design methodology. The bulk of this work is focused on finding ways to account for uncertainty in the design process and to produce robust designs that are insensitive to changes in the design and/or operational environment.