2006 CCRTS
THE STATE OF THE ART AND THE STATE OF THE PRACTICE

C2 Architectures; C2 Modeling & Simulation; C2 Analysis

Validating DoD Architectures: The Promise of Systems Engineering

Joseph P. Reynolds

Point of Contact:
Joseph Reynolds
Vitech Corporation
2070 Chain Bridge Road, Suite 100
Vienna, VA 22182
703-547-7014 | 703-883-1860 (fax)
jreynolds@vitechcorp.com

Validating DoD Architectures: The Promise of Systems Engineering

Joseph Reynolds

Abstract

Today's needs for interoperability and portfolio management, across military organizations worldwide, lead to increased focus on architecture development. Architecture frameworks are sponsored by Defense Departments in Australia, Canada, France, Korea, United Kingdom and United States.

The majority of today's efforts in architecture development focus on generation of disparate architecture views, seemingly without the benefit and rigor offered by systems engineering.

This paper describes the DoDAF validator–a Model Based Systems Engineering (MBSE) approach to the architecture development process. The approach is an extension to a proven systems engineering process that creates a win-win scenario for executive oversight team as well as the operational and engineering communities. The process also ensures that interoperability based on executable design verification leads to program success and consistent and accurate architecture perspectives for further analysis.

1 Brief Discussion of DoDAF

The Department of Defense Architecture Framework (DoDAF) is a set of documents that provide guidance on architecture products that describe the war fighter operations, business process and the systems that implement the operations and processes. Depending on the purpose and scope of the architecture being created the products will either describe current capabilities and implementation using existing systems or describe improved capabilities based on implementation of systems underdevelopment. The goal of creating an architecture description for existing capabilities is to be able to identify operational gaps that the current system structure is not capable to perform. The goal of creating an architecture description for systems under development is to identify improved process and the satisfaction on missing capabilities. The bottom line for systems under development is that there should be traceability to an operational need to defend the money spent for such a system.

The products that describe department of defense architectures are organized into three major categories; operational, system and technical views. The operational views

describe what is to be accomplished and who is responsible. These views are constructed from a set of operational activity / event threads that are designed to achieve a mission or operational task. System views describe the systems, including components, the functions the components perform and the links between the components, which are currently or will be the actual implementation of the operational activities. The technical views describe the standards and conventions that govern the design of the system or impacts on the system in the future based on changes in technology.
Within each category there are products describing slightly different perspectives of the overarching architecture to accommodate multiple stakeholders and to assist them in decision making processes. The objectives for the architecture products as a whole is to develop a common denominator for understanding, comparing, integrating and ensuring interoperability between interacting architectures.

2 Challenges in Meeting DoDAF Objectives

Meeting the objectives of DoDAF is no trivial task. The development and information required is outside of established mainstream processes. Often special working groups are created to generate the products and these groups are often divided into smaller groups that develop specific views based on individual expertise, leading to the creation of inconsistent views, and data and semantic interfacing problems. Reinforcement of inconsistency comes from those who certify the views; in general they too only examine portions in which they are experts. Using an architecture product production method independent of mainstream processes is asking for unidentified integration and interoperability problems and management oversight nightmares, let alone meeting the DoDAF objectives between architectures. The DoDAF Volume 2 does provide a language which ties the products together however elements represented in multiple products have relationships with other elements that should be defined and consistent. The lack of consistent definitions makes the task of comparing and integrating architectures even more difficult. Even if all products are consistent and comparable between architectures, there still is the challenge of validating the products actually describing the actual architecture accurately.

3 Solutions Offered by Model Based Systems Engineering to Support DoDAF

Model Based Systems Engineering (MBSE) supports DoDAF by providing a well-structured schema for all the elements, a repository that is inherently consistent and a method to validate operational activities, system functions and requirements, and an adaptable methodology to get the job done.

The biggest challenge is to develop a schema to relate and describe the elements that make up the architecture products. Having an underlying language will allow the developers of the architecture to create graphical and textual descriptions in a consistent manner, it will also allow for automation of traceability between views to ensure consistency. Systems engineering terms such as, components, functions, interfaces, exchange items, and resources already defines many of the elements contained in the architecture products. The Model Based Systems engineering approach developed by the founders of Vitech Corporation has created a robust schema defining the relationship between these elements dating back to the early 1970's. To support the DoDAF elements that are less systems engineering and more operational concepts the schema and methods needed to be extended. Traditionally operational requirements are handed off to the system engineer to design a system and often do not carry the original context of those requirements as pertaining to the set of missions trying to be achieved. Extending the MBSE schema, every element has a distinct language tying it to other elements within the architecture. The verb pattern describing relationships between the elements allows for complete traceability. Elements that appear in multiple views can now be described once, ensuring consistency form multiple perspectives.

The schema is the meta-model that describes architectures in a consistent manner; a repository is needed to hold the complete model of the operations desired, systems that implement the operations and the technical standards governing the systems. In the MBSE approach a common repository, which maintains all information of the individual elements and relationships to other elements is used to generate any necessary architecture products. The repository then becomes the complete model of the architecture being developed. Each element and its relationships exist only once within the repository and the architecture products are never stored but they are generated from the repository. Subject matter experts can work on specific elements or views and because the architecture views are not static, changes will be automatically propagated to all view containing those elements. Consistency and accuracy is now inherent in the process for developing the products, more important the DoDAF product production becomes a bi-product of proven repeatable mainstream methodology.

Prior to any changes made to the schema, repository or anything to do with the MBSE approach, many of the architecture development activities are already supported. Traditional MBSE consists of four domains; Source Requirements, System Physical Architecture, System Behavioral, Verification and Validation (figure1).

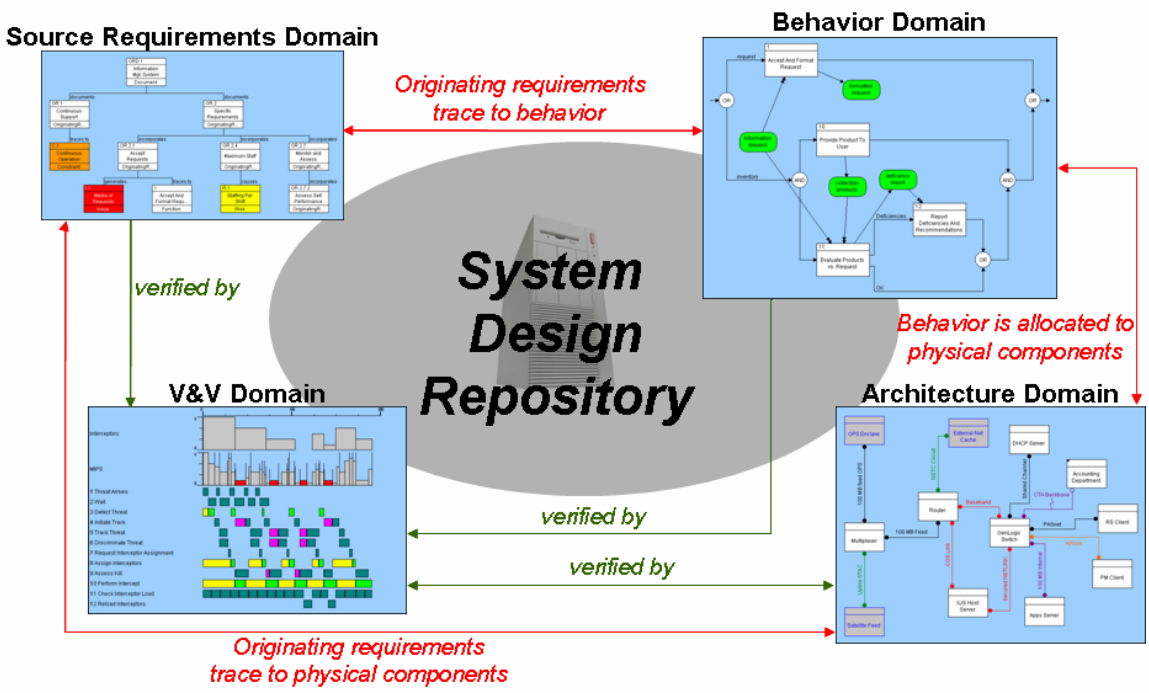

Figure 1 Four Domains of Traditional Model Based System Engineering

The source requirements domain consists of originating requirements from stakeholders, standards and conventions that govern how the system will be designed. The information for the technical views is already being captured in the repository. The Technical Standards Profile TV-1 view comes strait from the standards that trace to system functions, and the Technical Standards Forecast TV-2 is derived from function issues generated by emerging technology and standards.

The physical architecture domain consists of the components that perform the system functions and the links and interfaces, which carry items passed between functions. Graphically the systems links and interfaces are created using a physical block diagram with hierarchal decomposition. The information for System's Interface Description SV-1, System's Communication Description SV-2 and System-Systems Matrix SV-3 all will reside within this domain.

The system behavioral domain describes the functions that are needed to accomplish the stakeholder's needs as defined by the originating requirements. The systems functional behavior is modeled using Enhanced Functional Flow Block Diagrams (EFFBD), 'N squared' and IDEF-0 diagrams which are consistent with one another because the views are generated from the repository and created simultaneously as the repository is created. These diagrams contain many of the elements needed to produce all the system architecture views. More importantly, the EFFBD with its functions allocated to components in a physical block diagram is capable of simulating discreet time events controls, resources and characteristics of links between systems, Validating the System Functionality Description SV-4, System Data Exchange Matrix SV-6, Systems

Performance Parameters Matrix SV-7, States for the Systems State Transition Descriptions SV-10b, and the systems event trace description SV-10c.

The combination of the behavioral and physical domains contains all the data to produce a complete set of system architecture views, except for SV-5 Operational Activity to System Function Matrix. However, since the schema and repository has been extended to handle the operational views of DoDAF, mapping the operational activities to the functions that implement them can create this view. The extension of the schema to the operational view is mirror extensions of the MBSE approach. Operational Activities are derived from operational tasks and missions depicted in a High Level Operational Concept Graphic (OV-1), similar to system functions and their creation from the requirements domain. Like functions Operational Activities Model (OV-5) is created using the same techniques. The OV-5 model is also validated by simulating discrete time events controls and resources and automatically creates an Operational Event-Trace Description OV-6c as a result of the simulation. The Operational nodes and needlines contained in the Operational Node Connectivity Description OV-2 are models by using a physical block diagram similar to components and their links, however needlines do not have the capability of being simulated. Needlines drive the requirements of the system links and don't have physical sizes to be simulated. Operational Nodes perform activities just like functions are performed by components, thus once operational activities are allocated informally to their respected operational nodes the Operational Information Exchange Matrix OV-3 is created. The extension of the schema also includes who is responsible for the operational activities and system functions by using a hierarchal decomposable organization element to create an Organizational Relationships Chart OV-4.

One should remember that the architecture products or views are models describing the architecture and that in order to validate a model; the model must behave in a manner similar to what is being modeled to a degree to answer the appropriate questions. The operational activities and system functions are the only elements that exhibit behavior capable of being modeled in discreet time events, dependant on item exchanges and resources. Using an Enhanced Functional Flow Block Diagram (EFFBD) for the operational activity model (OV-5 view) and System Functionality Description (SV-4) will automatically create an Event Trace Description Timeline (OV-6c) and Systems Event-Trace Description (SV-10c) views when simulated. Validations between these views are automatic and ensure the actual behavior of the architecture is described in the logic of the EFFBD Diagram. Besides validating time sequences, the simulation capabilities of the MBSE approach takes into account a lack of resource that will delay an activity until it is available, validating system and operational resource requirement to perform a mission. Static Architecture products force someone to perform a trade study to obtain resource needs. All too often the need for such a study won't occur until after the damage of not knowing is done. The system functional models also have the capability of simulating capacities of links between components. Functions transfer items over links, the items have sizes and links have capacities, thus a function may be delayed until there is enough space available on the link.

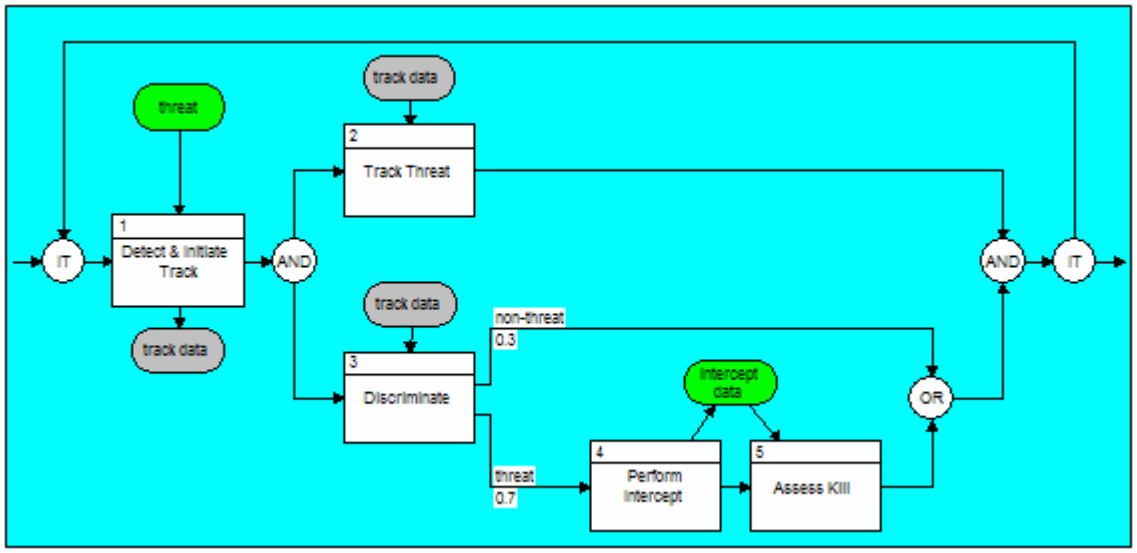

Figure 2 Enhanced Functional Flow Block Diagram

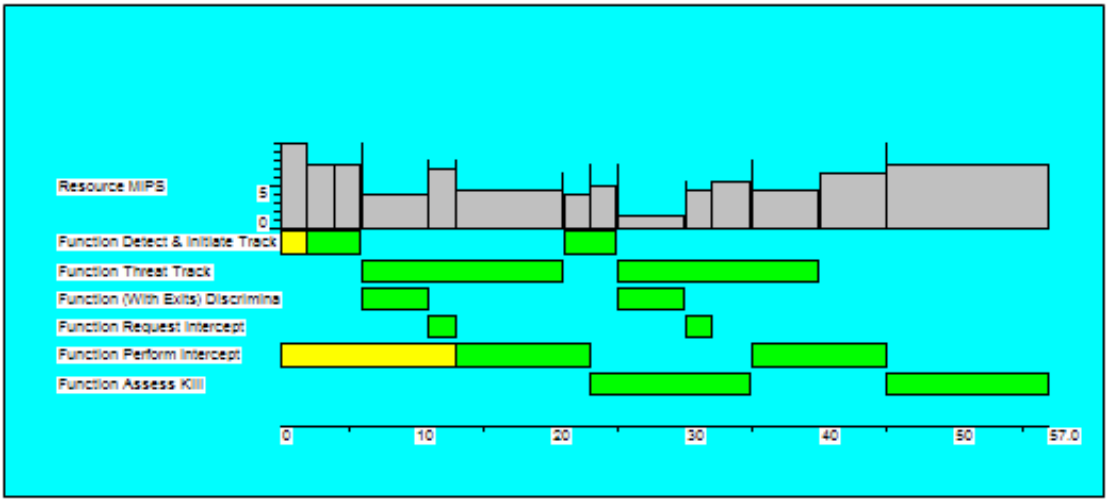

Figure 3 Simulation of EFFBD in Figure 2. Gray is amount of resources available, yellow is a delay caused by a control item, Green is the duration of the function. The y-axis is the type and name of the element and the x-axis is time.

Complete traceability is achieved by relating the operational elements to the system elements they implement. System functions implement the operational activities, system components implement the operational nodes, links between systems implement the operational needlines, and the items transferred by needlines are implemented by items carried by system links. The use of the schema and repository derived from the model based system engineering approach not only provides full automated traceability from an element in one view to a dependent element in another view, but also provides traceability to all the decisions tracked by issues and risks and the requirements driving the design of the systems.

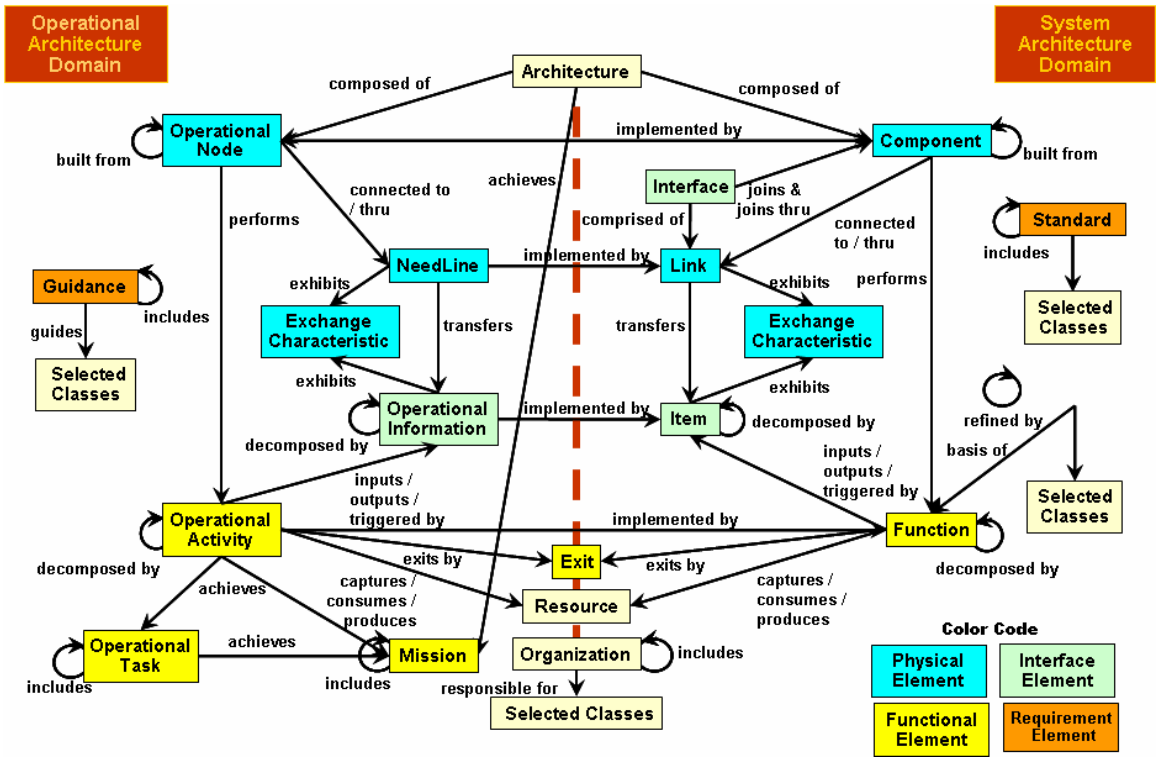

Figure 4 A Subset of the Extended Model Based System Engineering Schema Containing a Complete set of architecture elements needed to describe an architecture.

3 Summary

Working in an integrated environment built from a proven system engineering methodology resolves data and semantic interfacing problems associated with independent methods to create architecture descriptions. Reliability is achieved for management oversight and technical performance. Not only is integration between technical, operational, and system architecture descriptions, there is also integration with the decision making processes through system engineering verification and validation processes. The combination of the schema repository and simulation capabilities offered by MBSE creates a degree of intelligence; a change in a design element will automatically propagate to all architecture products containing the element. If a change in a design element affects a behavior element system operational or both, a simulation can be conducted to validate the change behavior is correct. Graphical and textual products are generated from the repository instead of being stored as a static view, which needs to be updated every time a change occurs, thus the architecture views are always consistent with the development life cycle, leading to significant savings in cost and schedules.

All of this is accomplished by extending the schema of the proven repeatable process of model-based system engineering approach.

References

Baker, Loyd, et al, “Foundational Concepts for Model Driven System Design,” published by INCOSE Model Driven System Design Working Group, 1996.

Childers, Susan and James E. Long, “A Concurrent Methodology for the Systems Engineering Design process”, *NCOSE 4th International Symposium*, 1994.

DoD Architecture Framework Version 1.0, (Volume I: Definitions & Guidelines; Volume II: Product Descriptions; Deskbook), 09 February 2004.

Crisp, Harry E., Naval Collaborative Engineering Environment, Presentation to INCOSE WMA, May 2002.

Long, James E., “Relationships between Common Graphical Representations Used in System Engineering,” *SETE 2000* (updated 2002).

Long, James E., “The Secret to Developing Systems of Systems: Model-Based Systems Engineering,” *SETE 2004.*

Long, Dinsmore, Spadaro, Alford, et al., "The Engagement Logic and Control Methodology as Derived, Defined, and Applied at TRW", unpublished notes, 1968 - 1972.

Maley, Joseph and James E. Long, “System Engineering and CORE®: A Natural Approach to DoDAF,” (updated 2005).