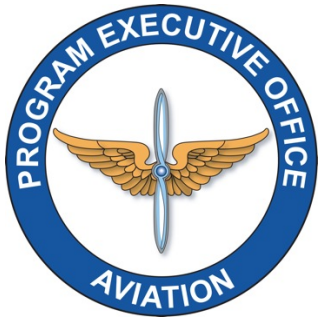

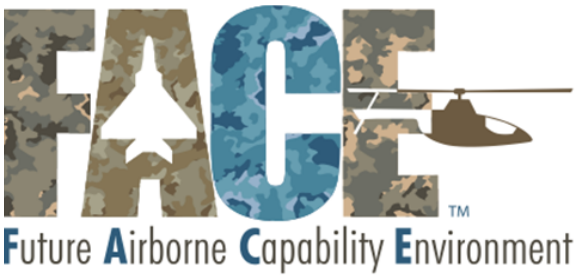

MBSE with Specificity - A Paradigm Shift in System Engineering

An order of magnitude improvement in System Development

US Army Aviation FACE™ TIM Paper by:

TES-SAVi, a subsidary of

Tucson Embedded Systems, Inc., FACE Member since 2010

Sean P. Mulholland, William G. Tanner, and Stephen M. Simi

Published 2 February 2016

US Army Aviation FACE™ TIM Paper

Document Title

Document No.: Doc. No.

Published by The Open Group, <Month> 2015.

Comments relating to the material contained in this document may be submitted to:

The Open Group
8 New England Executive Park
Burlington, MA 01803
United States

or by electronic mail to: ogface-admin@opengroup.org

Table of Contents

Executive Summary

Military and commercial aviation communities alike are hopeful for a "revolution" that springboards our industry from today's shortfalls in dealing with the complexity of Airworthy Open Systems platforms to a solution that allows for the rapid development of safe and secure systems. Is it such a stretch to imagine a set of models, once built, shuffled within systems diagrams, like LEGO™ blocks, forming a realistic, integrated representation of desired functionality and the end system? In such a world, systems engineers would be able to conduct open systems design trades, monitor internal data exchanges, analyze and correct performance characteristics, and ensure safe and secure operations in a systems architecture virtual or physical environment.

A solution may lie in the further advancement of leading edge methodologies and technologies in the areas of model-based engineering of systems, systems-of-systems analysis, design, development, and verification. New modeling languages such as AADL and the FACE Data Model Language have increased the specificity with which systems models can be built and may contain hints at the next "revolution" of system development. After all, are we likely to achieve such a leap using our current modes of thinking and continuing to press the limits of our current processes and technologies, or do we need a shift in thinking, a change in point of view, *a change in paradigm*?

Introduction

The need to build systems of greater and greater complexity has always existed. This is easily seen in the aerospace domain where over the past 20 years we have had a significant increase in the complexity of entire aircraft systems and subsystems. Within this timeframe, U.S. military aircraft have gone from independent avionics systems for each aspect of flight and mission management to highly integrated avionics subsystems comprising a highly functional, yet incredibly complex system. It has been predicted that overall system complexity will likely increase ten-fold within the next 10-20 years.

Such drastic numbers are due primarily to rapidly evolving and pressing capabilities supporting the warfighter: increased safety in extreme conditions such as Degraded Visual Environment (DVE) and Inadvertent Instrument Meteorological Conditions (IIMC), unmanned and partially-manned flight, mission and route re-planning and rerouting, cognitive decision aiding (CDA), data correlation and fusion, enhanced situational awareness (SA), non-traditional artificial intelligence algorithms, increased complexity and accuracy of missions, interaction between manned and unmanned teaming participants, and new cyber security requirements.

In order to achieve this order of magnitude increase in the complexity of aircraft systems and subsystems, new and pioneering methodologies and technologies, different than those used to build current systems, must be embraced. One area that is prepared for significant process improvement is in the specification of system requirements and design, using model-based system engineering (MBSE) processes and integrated tooling.

The current paradigm for systems engineering focuses on requirements decomposition in both raw textual and model-based textual formats. However they are still primarily textual. Natural language-based textual models of system requirements have proven ineffectual in describing the complete system. It is well documented that using natural language is typically ambiguous and difficult to adequately define the requirements in such a way as to be usable for analyses functions. [23]

An alternative paradigm is to unambiguously model *with specificity* all aspects of the effort, from user's needs, system requirements, and system design, through high-level hardware and software requirements, to low-level hardware and software requirements and design. Such an approach affords, with minimal input, auto-generation of engineering and process artifacts *at each subsequent level of the model hierarchy*, ensuring that the system we are building not only meets the correct requirements (verification), but also realizes the user's needs (validation) at every stage of development.

This paper's thesis is that such an approach has a very high likelihood of being an example of the desperately needed paradigm shift required to enable the aforementioned increase in complexity of future systems.

Requirements/Specification Model

Current System Requirements Techniques

The paradigm utilized for most systems engineering efforts focuses on requirements decomposition in both textual and pseudo model-based textual formats. Natural language based models of a system's needs are ineffectual at describing the complete system as the model carries with it the ambiguities of the language used. It is well documented that natural written language is typically ambiguous and difficult to adequately define the requirements in such a way as to not "imply" information that can easily be misinterpreted. To get around this limitation, data dictionaries are often employed to provide a universal description of terms used in the requirements and design of the system. The RTCA's DO-331 states "The following are NOT [emphasis added] considered as models within this supplement: Figures without syntax/semantics (these may be illustrative)" and "Equations related to natural language sentences."[5] In other words: simple pictures and natural language equations and statements are not models.

While DO-331 is primarily concerned with the software aspects of a system, as aerospace systems become significantly more software intensive, we purposely apply software modeling approaches to system requirements and design, thereby leveraging documents such as DO-331 to the systems engineering discipline.

Requirements Modeling Needs

Two types of models are introduced in RTCA's DO-331 [5]. The *Specification Model*—which represents high-level requirements that provide an abstract representation of functional, interface, or safety characteristics of software components that expresses these characteristics unambiguously to support an understanding and does not prescribe a specific software implementation or architecture except for exceptional cases of justified design constraints; The *Design Model*— which prescribes the software component internal data structures, data flow, and/or control flow including the low-level requirements and/or architecture at a level that it can be used to produce code.

DO-331 further states that a single model must be either a Specification Model *or* a Design Model. This poses an interesting problem, and represents and a departure from the current way in which many systems have been built and qualified as airworthy: models used for requirements are frequently very detailed functional block diagrams, sufficiently complete to support auto-code generation. We address this dilemma by identifying an appropriate modeling language that is unambiguous at a requirements specification level for the entire system under development, not just the software.

Also identified in DO-331, the specification and design models typically describe functional, performance, interface, and/or safety characteristics. A set of highly integrated well-defined models that can fully describe a system would require the specification of these as well as security, standards compliance, and architectural design.

A New Approach

System of Systems (SoS) Specification

Often when we model a particular system (or subsystem), we begin by describing the system itself. We focus our efforts on the description of the system in a system-centric perspective. Use Cases are often employed to create a loose description of the system and how it will interact with its external environment, be they interfaces, users of the system, or other external system components. Since the external interfaces are just that, external, we typically focus little attention on the parent system within which our new system will exist. In DoDAF a set of Operational Views (OVs) are typically created to identify a general picture of the System of Systems (SoS) and the interactions and operations of actors within the Concept of Operations.

System engineering activities focus primarily on the system itself. If we instead focus our attention initially on the problem domain of the entire system we would achieve a better understanding of the system we are developing. We could start with a definition of our System of Systems describing its components, behavior, performance of each system and their interfaces. Once a detailed model is created we would have described in non-ambiguous terms, the external interface requirements for the system and the validation framework for which our system must be measured.

One approach for the definition of the system components in the SoS, is to describe the "things" that exist within the SoS through a formal model process. A "thing can be thought of initially as the physical entities and concepts that are in the problem domain. We describe attributes of "things" and identify the relationships between each other. A description of the behavior of each "thing" can also be described in terms of its behavioral capabilities and interactions with other "things". This approach would abstractly define both the system of systems and each system component including the system we are developing. This system of systems definition can be used to extract the "needs" of the system we are developing. This approach reveals the essence of data modeling of components, systems, and systems-of-systems.

System Specification

As we follow the above-mentioned modeling approach, we identify the sub-components of the system we are developing within the context of the System of Systems. Our system specification can then be decomposed into its component pieces. Each of the sub-components is refined by adding additional details such as performance requirements, more detailed behavior details through functional definitions, states and modes definition, and error and fault handling. Additionally, interface definitions are refined by building on the subcomponent relationship definitions for both data and control.

The approach discussed implements a hierarchical SoS definition used to specify a system constructed of other system elements. Each system element is described conceptually by both its behavioral model and by the systems data model. This hierarchical modeling approach is also applied to the concept of a System of Systems (SoS). A SoS is a system-of-interest whose system elements are themselves systems; typically these entail large-scale inter-disciplinary problems involving multiple, heterogeneous, distributed systems.

An interesting point is that the FACE Conceptual Data Model [1] may be able to be leveraged and expanded upon to develop the Specification Model as defined in DO-331. By adding behavioral meta-model elements and "required" relationships to the conceptual model, we may be able to attain a Specification Model that is

both unambiguous in its definition of the requirements, does not rely on natural language and yet does not prescribe an implementation.

System and Software Design

Implied by the nature of decomposition of the system specification, the system design is a further refinement and decomposition of the system specification. Many modeling languages and tools currently exist which provide the specificity needed for System and Software design to be used to generate code and tests. Two of the more recent modeling languages, AADL and the FACE™ Data Model language, focus on defining models with a high-level of specificity that is sufficient for code generation, test generation, and formal analytical methods.

AADL focuses on defining an unambiguous architecture-centric model of system software and hardware with the goal of providing analyzable models. A major benefit of AADL is that it unifies the models into one system architectural model. This allows identification of mismatches that occur when different models of timing, fault tolerance, security model, etc. are integrated, reducing the time of analysis to detect system failures that are typically realized later during system operation [13].

The FACE™ Data Architecture provides a standard method for data sharing between software components. Data modeling is required to enable information sharing and interoperability between software components. A common data model enhances reuse by establishing a standard communication data definition between software components. The FACE™ data model specification [14] achieves significant benefits of describing all data elements with sufficient specificity in both semantics and measurements at a level that once combined with required processor hardware specification is sufficient to auto-generate code.

The end goal of the FACE™ Data Model Language [14] is to sufficiently define the data semantics for data and control messages (structures and fields) exchanged between two software components. The FACE™ Data Model Language, revision 2.1, provides for both a semantic description and an unambiguous measurement system specification for each datum. By providing for an unambiguous non-arbitrary semantic and measurement description of message fields, developers can fully specify messages and can therefore increase interoperability and ease of integration of software components.

The not yet ratified FACE v3.0 Data Architecture builds on the FACE data model and provides an integration model that describes the connectivity and data transformations that occur within the FACE Transport Services Segment that are required to interface two FACE applications or services.

Both AADL and the FACE™ Data Model Language represent significant technological advances in Model Based Engineering (MBE). AADL defines an unambiguous architecture of control systems design, and FACE™ Data Model Language defines unambiguous interaction between system software elements.

However, the current SysML, FACE Data Architecture, and AADL modeling tools are not integrated. While each serves a significant purpose, there has been relatively little work performed to define the integration points. There are several approaches for the integration of multiple modeling languages. One approach as identified in the Impact Study [23] is to develop a “model exchange/interchange and model co-simulation approaches that facilitate integration while allowing organizations to choose the right tools for the task”. Other approaches are often proprietary in the way that the models are integrated. The resultant models are specific and proprietary to a particular project, company, or tool chain. Another approach is the

AWESUM™ approach [24], which provides a System Unified Model (SUM) to integrate the various disciplines and tools in a non-proprietary model supporting open systems development.

Summary

In the next few years more experience in applying MBSE and MBE tools and simulations for obtaining certification under Military standards and DO-178C/DO-331 will be achieved. Expansion of existing toolsets in the areas of FACE Data Architecture, AADL, SysML, UML, and other newer modeling processes will have to be developed further in order to streamline modeling and simulation techniques. In addition, techniques, training and tools will likely be developed to support robust configuration management of large-scale system models. Also, the model languages and tools will be modified to support multidisciplinary engineering beyond just system requirements and design, computer aided design, software design and verification, to include power systems, system of systems, human factors, safety models, and security models.

While Model Based Engineering by itself is not a "silver bullet", MBE is an "enabler". If systems are modeled appropriately MBE can be leveraged to improve the system development lifecycle to build systems that are up to ten times more complex than current systems. In order to realize these improvements, the specification and design models must have the specificity to support unambiguous validation, verification, hardware and software generation, document generation, and analyses for systems integration activities such as timing, network load, memory utilization, and more. Therefore, while MBE is not a "silver bullet", it is essential to achieving the building of the ever-increasing complex systems of the future.

Further study is recommended for the integration of the FACE Data Architecture with SysML and AADL, as well as FACE Data Architecture support for a more complete system conceptual model to include: entity defining relationships (where relationships can be used for uniqueness of entities), behavioral specifications added to conceptual modeling, choreography of message interfaces, and integration of components to define a system.

***A closing note**. While the decomposition of system requirements to hardware and software requirements to hardware and software design is effective, it does not typically lend to reuse of software elements. This is the result of very specific requirements being imposed on the design of software components that often restrict the design of the component and does not consider the desire for portability and reusability of the component. While this is often true, it does not have to be. If the quality attributes of portability and reusability are "passed down" as system requirements, the designers of the software can use this as justification for the more robust and multipurpose functionality of their design.*

References

[1] "Technical Standard for Future Airborne Capability Environment (FACE™) Edition 2.1," The Open Group, May 2014.

[2] "Airworthiness Qualification of Aircraft Systems," US Army Regulation AR 70-62, Research, Development, Acquisition, HQ Department of the Army, 21 May 2007.

[3] "Advisory Circular AC 20-115C – Airborne Software Assurance," US Department of Transportation, Federal Aviation Administration, July 2013.

[4] "RTCA DO-178C – Software Considerations in Airborne Systems and Equipment Certification," RTCA Dec. 2011.

[5] "RTCA DO-331 – Model-Based Development and Verification Supplement to DO-178C and DO-278A," RTCA Dec. 2011.

[6] "Advisory Circular AC 20-148 – Reusable Software Components," US Department of Transportation, Federal Aviation Administration, December 2004.

[7] "RTCA DO-297 – Integrated Modular Avionic (IMA) Guidance and Certification Considerations," RTCA Nov. 2005.

[8] "Method and Apparatus for Interfacing with Multiple Objects using an Object Independent Interface Protocol," US Pat No 8,239,586, Tucson Embedded Systems' Capability Driven Architecture (TES' CDA).

[9] "Capability Driven Architecture: An Approach to Airworthy Reusable Software," Tucson Embedded Systems, American Helicopter Society 63rd Annual AHS International conference, Virginia Beach, Virginia, May 2007.

[10] "Reusable Automated Platform SIL Testing -- A Cost- Effective Risk-Reduced Process for Airworthy Reusable Software," Tucson Embedded Systems, American Helicopter Society 66th Annual Forum, Phoenix, AZ., May 2010.

[11] "Enabling Situational Awareness and Network Centric Operations for Systems utilizing FACE™ Open Systems Architectures," Tucson Embedded Systems, American Helicopter Society International Specialists' Meeting on Unmanned Rotorcraft and Network Centric Operations, Scottsdale, Arizona, January, 2013

[12] "Systems Engineering Vision 2020," International Council on Systems Engineering (INCOSE), TP-2004-004-02, 2007.

[13] "Model-Based Engineering with AADL," Feiler & Gluch, Pearson Education Inc., 2013.

[14] "FACE™ v2.0 & v2.1 Reference Implementation Guide," Open Group, 2014.

[15] "MIL-HDBK-516B – Airworthiness Certification Criteria US Department of Defense (DOD), Sept 2005.

[16] "Joint Software System Safety Committee (JSSSC) Software System Safety Handbook," 1999.

[17] "A Survey of Situational Awareness Requirements in Air-toAir Combat Fighters," Endsley, 1993, Department of Industrial Engineering Texas Tech University. Journal of Aviation Psychology.

[18] "Situational Awareness Information Dominance & Information Warfare," Endsley & Jones, 1997, U.S. Air Force Armstrong Laboratory, AL/CF-TR-1997-0156.

[19] "Revisions to the JDL Data Fusion Model," Steinberg, Bowman, White & Jones, 1999, ERIM International Inc.

[20] "Aviation 2050 Vision - Technology for Tactics," 2013, Dr. Bill Lewis, Director of the AMRDEC's Aviation Development Directorate.

[21] "Developer's Handbook for Airworthy, Reusable FACE Units of Conformance," Carter, Simi, Tompkins; 2014, US Army AMRDEC-SED.

[22] "System Architecture Virtual Integration (SAVI)," Aerospace Vehicle Systems Institute, Texas A&M University System.

[23] "INNOVATION AND MODERNIZATION PROJECTS AFFECTING CAPABILITIES AND TECHNOLOGY (IMPACT) STUDY: THE AIRWORTHINESS OF COMPLEX SYSTEMS - Final Report", Mr. David R. Arterburn, Study Lead - University of Alabama in Huntsville, 2015, US Army Aviation Development Directorate Contract # W31P4Q-10-D-0092 DO84

[24] "TES-SAVi AWESUM™ Model-Based Systems Engineering (MBSE) for FACE™ Applications", Stephen M. Simi, IEEE Aerospace Conference 2014

About the Author(s)

Sean P. Mulholland, is one of the co-founders of Tucson Embedded Systems, Inc. Sean has a B.S in Computer Science and Systems Design from the University of Texas at San Antonio. Sean currently serves as TES CEO and President. Sean has 27 years of experience in software intensive system development, design, integration and testing, especially as it relates to mission critical and safety critical systems. Sean has designed and built several product lines that produced significant advancements in the areas of Geographic Information Systems, Military Ground Systems, Unmanned Ground Vehicles, Unmanned Aerial Vehicles, and Manned aircraft systems. Sean is a contributing author to the FACE™ technical reference architecture and has been active serving as a key resource in FACE Data Architecture development. Sean's current work is focusing on the development of a process and supporting tool suite for optimizing the system development of safe and secure systems in military and commercial aircraft.

William G. Tanner has a B.S. in Computer Science Engineering from Northern Arizona University and over 20 years of embedded software and embedded software application development experience, 14 of which are in software engineering project and product management. Bill is a contributing author to the FACETM technical reference architecture, has been active serving in the FACE Data Model Working Group, and is the lead data modeler for the Joint Common Architecture (JCA) Data Model. Prior to joining TES in 2006, Bill worked for IBM as a software engineer in the disk storage system division, and Project Technology and Mentor Graphics as a data modeler and project manager for the division's UML modeling, verification, and code generation tools.

Stephen M. Simi, serves as TES-SAVI's Vice President. Stephen has 30 years of experience design and developing engineering and scientific applications, and managing multiple programs. Stephen is also very active in the FACE Consortium's Conformance, Airworthiness, and Outreach subcommittees. He is recognized as an industry innovator of agile technologies that can be applied to Joint forces across the common operating picture/battlespace of C4ISR assets, and an industry expert in lifecycle development of reusable software systems. He has several technical publications and presented to the AHS, AOC, AIAA/IEEE societies, to FACE and MITRE on areas of software development, reusable systems, and advanced modeling and simulations of those systems. Stephen currently manages 4 US army programs, JCA, MIS, R2C2, and MICD for TES.

Stephen has a B.S. in Physical Sciences (Math Computer Sciences, and Engineering) and a M.S. in Engineering from the University of Maryland. Before working for TES, Stephen served as the Director of Software Development, and Director of Software Business Development at world-renown optics company Breault Research. He also served as a technical fellow at the MITRE Corporation for the US Army, and served various other organizations designing, developing, and testing engineering and scientific applications over his 30-year technical career

About The Open Group FACE™ Consortium

The Open Group Future Airborne Capability Environment (FACE™) Consortium, was formed in 2010 as a government and industry partnership to define an open avionics environment for all military airborne platform types. Today, it is an aviation-focused professional group made up of industry suppliers, customers, academia, and users. The FACE Consortium provides a vendor-neutral forum for industry and government to work together to develop and consolidate the open standards, best practices, guidance documents, and business strategy necessary for acquisition of affordable software systems that promote innovation and rapid integration of portable capabilities across global defense programs.

Further information on FACE Consortium is available at www.opengroup.org/face.

About The Open Group

The Open Group is a global consortium that enables the achievement of business objectives through IT standards. With more than 500 member organizations, The Open Group has a diverse membership that spans all sectors of the IT community – customers, systems and solutions suppliers, tool vendors, integrators, and consultants, as well as academics and researchers – to:

- Capture, understand, and address current and emerging requirements, and establish policies and share best practices
- Facilitate interoperability, develop consensus, and evolve and integrate specifications and open source technologies
- Offer a comprehensive set of services to enhance the operational efficiency of consortia
- Operate the industry's premier certification service

Further information on The Open Group is available at www.opengroup.org.