AEROSPACE REPORT NO.
TOR-2017-01695

Mission Assurance Considerations for Model-Based Engineering for Space Systems

June 27, 2017

Marilee J. Wheaton
Systems Engineering Division
Engineering and Technology Group

Prepared for:

Space and Missile Systems Center
Air Force Space Command
483 N. Aviation Blvd.
El Segundo, CA 90245-2808

Contract No. FA8802-14-C-0001

Authorized by: Engineering and Technology Group

Developed in conjunction with Government and Industry contributions as part of the U.S. Space Programs Mission Assurance Improvement Workshop.

Executive Summary

Defense systems in general and space systems in particular, are characterized not only by their demanding missions, intricate and risky systems development, but also by the detailed documentation that must be created, managed, and interpreted throughout a system's lifecycle. The discipline of systems engineering was born out of these complex systems, dating back at least to the Apollo program, with the goal of improving the success of these complex systems; however, the document-heavy approach of traditional systems engineering seems out-of-step with our modern age, which saw the emergence and primacy of computing, the internet, and mobile technology. Moreover, defense and space systems customers are looking for systems with shorter development cycles and more iterative development styles than were common when the systems engineering practice first emerged while focused on mission success. However, the creation, management, and interpretation of documentation (much of it required for sound communication as well as the need of customers for mission success with manageable risk) can degenerate into churn and inconsistencies when subject to iterative and change-tolerant development models.

One methodological solution to more manageable iterative development is model-based systems engineering specifically for the systems engineering implications and model-based engineering for this same problem when extended across disciplines. With a model-based approach the truth is resident in the model, not in the documents. Systems models enable improved mission/requirement/design trades to detect potential defects and correct disconnects early, enable design decisions to improve systems performance reliability, safety, and improve the ability to conduct reviews in a more-effective and proactive manner. Documents may be generated as-needed but the accepted baseline is always in the integrated model which the team creates, modifies, and interprets during the evolution of the system design. Given the significant paradigm shift between document-based and model-based work, the mission assurance methods must change as well. In this work, we have identified six key mission assurance process areas which will likely be impacted by the shift to model-based work: requirements analysis and validation, reliability, system safety independent reviews, design assurance, and configuration management. In the body of this document we summarize our recommendations for systems engineering, program management, and mission assurance leaders to consider for model-based programs. We also describe a set of implementation recommendations to retain and enhance mission assurance during the transition from document-based to model-based work.

Appendices capture the detailed work for each mission assurance area along with detailed recommendations. For each area specifically, we define the current state of each area, the expected future model-based state, and describe both the risks and opportunities for mission assurance practitioners to consider when planning, and executing mission assurance activities on future model-based projects.

Acknowledgments

This document was created by multiple authors throughout the government and the aerospace industry. We thank the following contributing authors for making this collaborative effort possible:

Albert Hoheb	The Aerospace Corporation
Aliki Loper-Leddy	SSL
Chris Schreiber	Lockheed Martin Corporation
Dave Gianetto	Raytheon Company
Howard Gans	Harris Corporation
Marilee Wheaton	The Aerospace Corporation
Melissa Meyers	JPL
Myron Hecht	The Aerospace Corporation
Nadia El-Sherief	Raytheon Company
Michael Chory	MIT Lincoln Lab
Steve Martin	SMC

A special thank you for co-leading this team and for their efforts to ensure completeness and quality of this document goes to Dave Gianetto (Raytheon Company) and Marilee Wheaton (The Aerospace Corporation). Another special thank you goes to the team steering committee champions, Mark Baldwin (Raytheon) and Anne Ramsey (Harris) for their vision and leadership.

The entire team deeply appreciates the contributions of the subject matter experts (SMEs) who provided source material for and/or reviewed draft versions of the document:

Ryan Noguchi	The Aerospace Corporation
Donna Nystrom	The Aerospace Corporation
Norm Lao	The Aerospace Corporation
Kalyani Rengarajan	The Aerospace Corporation
Carter Wright	Ball Aerospace & Technologies Corporation
Robert Adkisson	The Boeing Company
Bill Sharp	The Boeing Company
Ed Moshinsky	Lockheed Martin Corporation
Ronald Mandel	Lockheed Martin Corporation
Anne Ramsey	Harris Corporation
Scott Gibbons	Harris Corporation
Frank Lombardo	Harris Corporation
Alan Zoyhofski	Harris Corporation
Martin Feather	JPL
John Evans	NASA
Larry DeFillipo	Orbital-ATK
Mike Violet	Orbital-ATK
Mark Baldwin	Raytheon Space and Airborne Systems
Spencer Studley	SSL
Harry DuRettle	Government

Contents

Figures

Tables

1. Introduction

1.1 Background

Increasing system complexity of national security space systems results in greater likelihood of systems engineering (SE) escapes and greater impacts of these escapes when they are not found and rectified early. Examples of SE escapes include ambiguous requirements, disconnects in requirement flow down or allocation, and unacknowledged requirement interdependencies. In addition, system complexity has increased from the growing amount of legacy systems and interfaces which can constrain system evolution and management complexity as increased with teams of organizations collaborating on the effort. Challenges of interoperability and enabling the operations of systems of systems (SoS) also contribute to increasing enterprise complexity. Also, incremental and spiral development of systems result in multiple requirement and design baselines.

Development by many needed specialists results in information stovepipes. These stovepipes can be organizational, structural, temporal, or project based. Key knowledge can be lost between lifecycle phases and projects. Inconsistent representations of the system in individuals' mental models exist, that is, a lack of a "common SE picture" and a degradation to mission assurance. Ambiguous specifications of requirements and design result in the increased risk of SE escapes not being discovered until later in the lifecycle.

Documents are brittle artifacts. They are difficult to keep synchronized and consistent, inconvenient for capturing complex and numerous interrelationships, and disconnects may be difficult to find.

Model-based systems engineering's (MBSE) relevance to government programs is established when the information captured in system models is proactively used to meet the success criteria of SE, mission assurance (MA) reviews, audits, and project management milestones.

Due to the fast pace of technology innovation, security threats and new standards and regulations during the lifecycle development, the requirements on a system may change. Having the requirements and design in one model enables the impact analysis of these changes on mission assurance.

Another relevant criterion is when MBSE practices improve the accuracy, efficiency, timeliness, and coordination of outcomes. Thirdly, that MBSE is integral to the acquisition strategy, contract implementation, and engineering activities and data across all stakeholders—government and industry. Data will promote consistency of vision and shared understanding among all stakeholders.

1.2 Mission Assurance Improvement Workshop (MAIW) Problem Statement

Mission assurance is founded on sound, repeatable, SE. Traditional SE methods where requirements baselines, hardware/software models, and product assurance criteria are all managed independent of solution architectures are insufficient to address the demands for resilient systems at lower price points and faster fielding. An approach for integration of these typically disparate elements is needed to leverage the developments in model-based engineering (MBE) into an effective execution framework. Any such approach, where models and software based systems and tools can replace documents in the verification and validation flow, must retain MA discipline through the program lifecycle.

This report focuses on how to adapt traditional MA practices to function effectively in an MBE setting. Figure 1 extends the traditional systems engineering Vee. The left portion of Figure 1 identifies program assurance that sets up the mission assurance program. Program assurance would include the planning activities necessary to apply MBSE/MBE to the work to be performed. The right most portion of Figure 1

shows that the results are mission success stemming from mission assurance activities that result from proper systems engineering with an emphasis on the tasks necessary for mission success. Figure 1 also shows how the engineering work extends past verification and validation through operations and sustainment, and finally how that flows back into defining new missions and the need for updated program assurance.

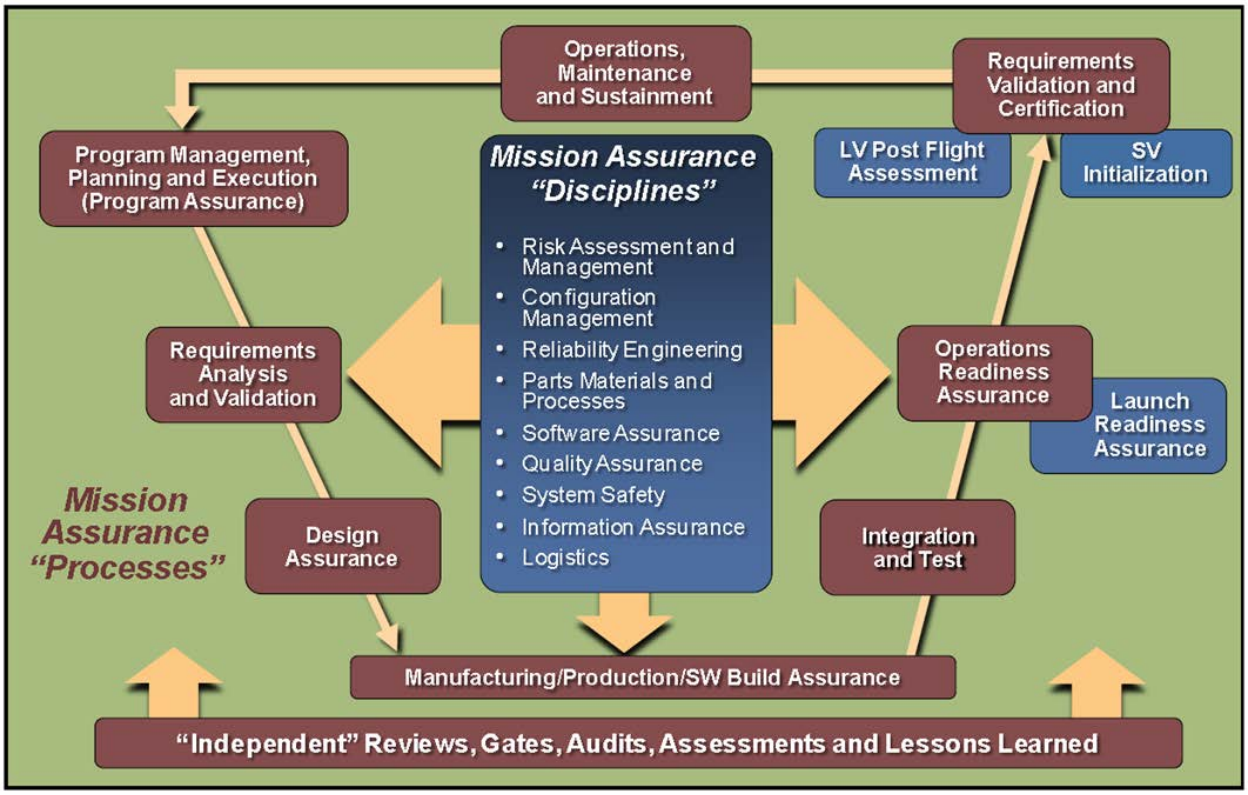

Figure 1. Mission assurance through systems engineering.

1.3 MBE Definition and Overview

Models are not new, as model-based approaches have been widely used in many disciplines, such as mechanical, electrical, and software for some time. However, these approaches primarily use models just to support analytical studies supporting performance analysis with no consideration given to other aspects of the system such as requirements, system structure, interfaces, and functionality. In MBE, models replace documents as the central focus, and provide a single source or truth, addressing all aspects of systems development throughout the lifecycle from requirements definition to verification. MBSE is often placed in the broader context of MBE. In this context, the system model is intended to be integrated with those models used by other disciplines. In this way, the system model serves as an integrating framework that spans the different disciplines [38].

MBSE is the emerging, more evolved practice of SE in which descriptive system models are at the center of the SE process. These models are built hierarchically and integrate many perspectives, are evolved and matured over time, and serve as the "single source of truth" for the system. MBSE facilitates a "common SE picture" in which everyone's mental models become more complete and consistent. Requirements and design can be validated and verified early and often to reduce risk of lingering SE defects. System modeling can provide graphic views that effectively convey necessary information at any level of abstraction. Customizable queries and reports provide comprehensive analysis of traceability and impacts

across the system or enterprise. SE overlaps with software engineering, requirements, design, interface, constraint, and other information contained in the MBSE tools should be seamlessly traceable to software tools. Figure 2 shows the systems engineering Vee as it would look with major steps/gates (grey boxes) and associated engineering work (red boxes) contributing to integrated digital systems models that provide a single source of truth to produce trusted views, reports, and documents such as requirement specifications, traceability analysis reports, impact analysis reports, interface control documents, training documents and provide outputs to other tools such as cost and schedule estimation tools or risk management data tools.

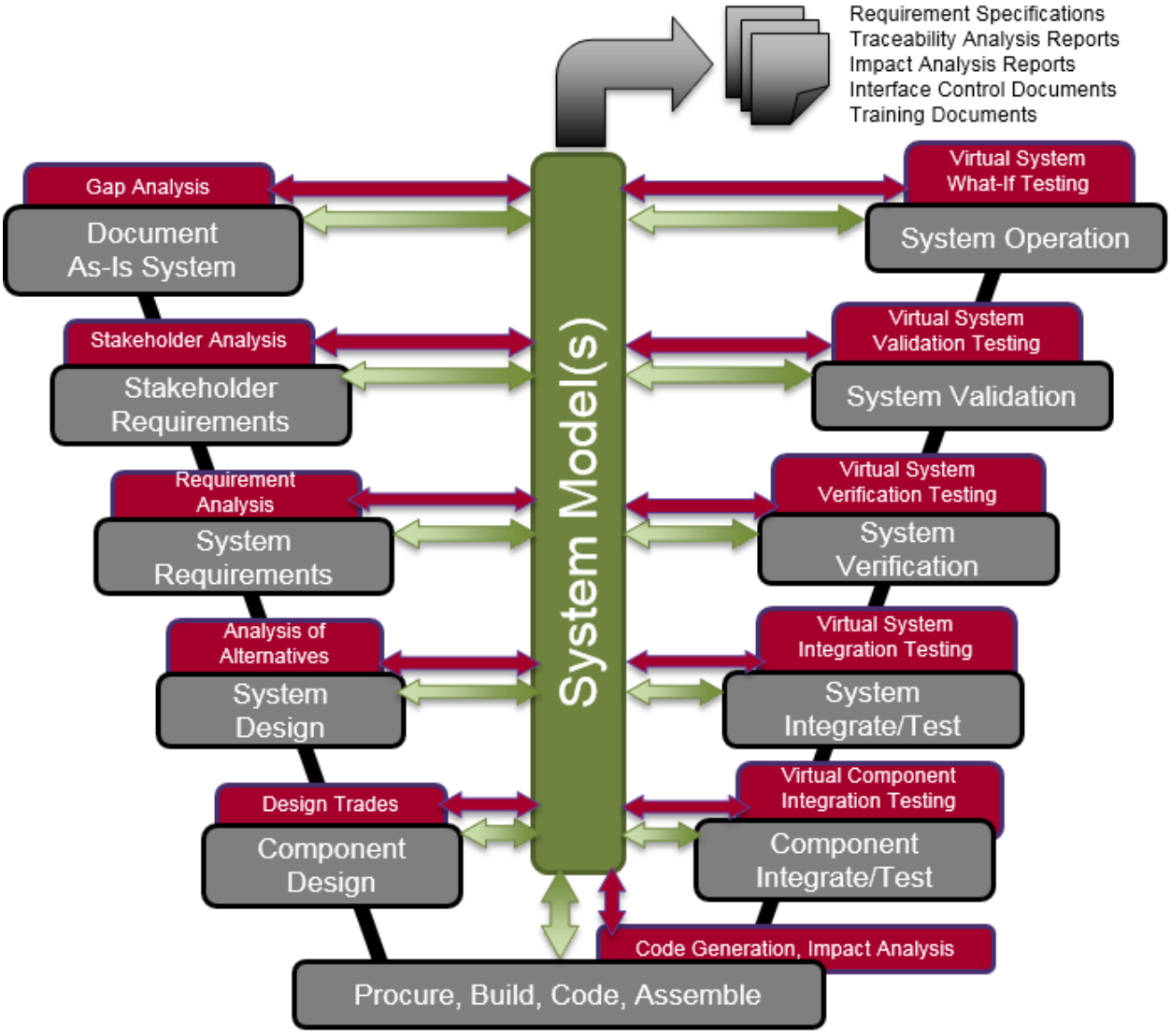

Figure 2. Concepts related to MBE.

Mission Success (MS): Defined as the achievement by an acquired system (or system of systems) to singularly or in combination meet not only specified performance requirements but also the expectations of the users and operators in terms of safety, operability, suitability, and supportability. MS is typically evaluated after operational turnover, according to program specific timelines and criteria, such as key performance parameters (KPPs). MS assessments include operational assessments and user community feedback.

Mission assurance (MA): Defined as the disciplined application of proven scientific, engineering, quality, and program management principles towards the goal of achieving mission success. MA follows a general SE framework and uses risk management (RM) and independent assessment as cornerstones throughout the program lifecycle.

MBE: An approach to engineering that uses models as an integral part of the technical baseline that includes the requirements, analysis, design, implementation, verification of a capability, system, and/or product throughout the acquisition lifecycle.
– *NDIA,* Final Report of the Model-Based Engineering Subcommittee*, 2011*

MBSE: The formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase, and continuing throughout development and later lifecycle phases.|
– *INCOSE,* Systems Engineering Vision 2020*, 2007*

Definition of Digital Engineering

The concept of digital engineering (another term for MBE) and digital systems models (another term for model data sets) has been a recent emphasis area for the Office of the Deputy Assistant Secretary of Defense (ODASD) for SE. Per their web page; http://www.acq.osd.mil/se/initiatives/init_de.html. ODASD (SE) is working to lead the Department of Defense (DOD):

> "Digital engineering (DE) (also known as MBE or model-based systems engineering) is an initiative developed and championed by ODASD (SE) to help streamline the way defense programs collect, retain, and share data. ODASD (SE) asserts that digital engineering has the potential to promote greater efficiency and coherence in defense programs by ensuring stakeholders have access to accurate, relevant, and consistent information throughout the life of a program. … Operating from a single source of truth and placing models as the lead digital artifacts move us in this direction. ODASD (SE) believes that the use of models, simulations, and digital engineering places a greater focus on the rigor and discipline needed in performing high-quality systems engineering."

Digital Artifact (DA): The artifacts produced within, or generated from, the digital engineering ecosystem. These artifacts provide data for alternative views to visualize, communicate, and deliver data, information, and knowledge to stakeholders.

Digital Engineering (DE): An integrated digital approach that uses authoritative sources of systems' data and models as a continuum across disciplines to support lifecycle activities from concept through disposal.

Digital Engineering Ecosystem: The interconnected infrastructure, environment, and methodology (process, methods, and tools) used to store, access, analyze, and visualize evolving systems' data and models to address the needs of the stakeholders.

Digital Model-Centric Engineering (DMCE): The application of engineering practices through the use of digital environments and tools. DMCE enables practitioners to engineer systems using digital practices and artifacts in a collaborative environment, creating a digitally integrated approach using a federated single source of truth to evolve complex systems. A primary characteristic of this environment and approach is the digital authority's ability to capture pedigree of all system-related data to facilitate and automate traceability, show dynamic relationships and changes to various aspects of the system development, and support decision-makers to make informed decisions.

Digital System Model (DSM): A digital representation of a defense system, generated by all stakeholders, that integrates the authoritative technical data and associated artifacts, which defines all aspects of the system for the specific activities throughout the system lifecycle [9].

Digital Thread: An extensible, configurable, and component enterprise-level analytical framework that seamlessly expedites the controlled interplay of authoritative technical data, software, information, and knowledge in the enterprise data-information-knowledge systems, based on the DSM template, to inform decision makers throughout a system's lifecycle by providing the capability to access, integrate, and transform disparate data into actionable information [9].

Digital Twin: An integrated Multiphysics, multiscale, probabilistic simulation of an as-built system, enabled by Digital Thread, that uses the best available models, sensor information, and input data to mirror and predict activities/performance over the life of its corresponding physical twin [9].

Technical Coherence: The logical traceability of the evolution of a system's data and models, decisions, and solutions throughout the lifecycle.

Technical Data: Recorded information, regardless of the form or method of the recording, of a scientific or technical nature (including computer software documentations). The term does not include computer software or data incidental to contract administration, such as financial and/or management information. (DFARS 252.227-7103 (a) (15))

2. Scope

2.1 Topic Team Charter

The charter for the MAIW topic team included:

1. Define an approach for executing MBE while retaining essential MA processes and deliverables.

2. Determine how MA processes execution and output/reporting may change when using an MBE approach. Consider several program execution MA processes in TOR-2011(8591)-21 as part of an essential set.

3. Identify a minimum set of model capabilities required to execute the MA processes.

4. Stretch task: Investigate how the roles of quality assurance and inspection might change in an environment where traditional paper outputs and physical inspection may be replaced by models and virtual inspection embedded into manufacturing processes.

2.2 Relationship to Other MA Documents

This document leverages the 2010 MAIW product, "Mission Assurance Program Framework," that defined 16 processes supporting mission success that were universally consistent across all organizations, and considered the essential set necessary to provide effective MA for US space programs. The "Mission Assurance Guidelines for A-D Mission Risk Classes" document contains guidelines to define characteristic profiles for MA processes for a given space vehicle risk class. For this 2017 MAIW topic "MA Considerations in MBE for Space Systems," we wanted to research and identify the MA impacts on a select group of these 16 processes for transitioning to MBE approaches on space system development programs. We further wanted to assess the impact to program execution mission success by transitioning some of the key processes to a MBSE approach. The expectation is that users of this 2017 MAIW product will use it in concert with both of these previous MAIW products.

2.3 How the User Should Approach this Document

This document should be viewed as a compendium document to the previous MAIW products identified in B. This document provides guidance to the MA practitioner on the changes affecting projects in going from a current state to a future state that employs MBE methodologies and practices. For each of the mission assurance areas addressed in this document, users of this document are provided recommendations for how to transition from their current practices to a MA enabled MBE approach, including risks and challenges that they can use to guide their efforts towards a successful outcome. This document also identifies changes in the responsibilities from the current state to future state for MA roles such as program management, SE, quality assurance, and other key roles. In addition, the document includes a summary of lessons learned and best practices collected from across the MAIW community, and it provides some recommendations for each of the process areas examined in this document. Detailed guidance is contained in the appendices for each of the included MA process areas.

2.4 INCOSE Vision

One of the SE imperatives in the SE Vision 2025 calls for advancing the tools and methods to address complexity. As documented in the INCOSE Vision, "MBSE is part of a long-term trend toward model-centric approaches adopted by other engineering disciplines, including mechanical, electrical and software. In particular, MBSE is expected to replace the document-centric approach that has been

practiced by systems engineers in the past and to influence the future practice of SE by being fully integrated into the definition of SE processes."

One of the SE challenges identified is that mission complexity is growing faster than our ability to manage it…increasing mission risk from inadequate specifications and incomplete verification. Knowledge and investment are lost at project lifecycle phase boundaries… increasing development cost and risk of late discovery of design problems. Applying MBSE is expected to provide significant benefits over the document centric approach by enhancing productivity and quality, reducing risk, and providing improved communications among the system development team [42].

3. Approach

3.1 How MA Areas Were Chosen

During the team kick-off meeting, the team members reviewed the set of MA processes contained in TOR-2011(8591)-18. During a discussion of all of the process areas (see Figure 3) the team identified, via consensus, which set of MA processes were a first priority essential set to consider for this year's topic team to investigate MA impacts of MBE approaches. The team also discussed topic team scope reasonableness given the MAIW schedule and resources. Based on the discussion, the following processes indicated by the arrows below in Figure 3 were selected for the 2017 topic team. Two other process areas were identified by the team as also being essential to investigate, but were decided to be recommended for a follow-on topic team – integration, test and evaluation, and environmental compatibility given the extensive scope of work represented by these two process areas. Detailed information in each of these selected six MA processes is in Appendix A – F. The other process areas not selected could be addressed as needed in follow on work related to mission assurance considerations for model-based engineering.

No.	Recommended Mission Assurance Process	Process Group
1	Requirements Analysis and Validation	1. Program Execution
2	Design Assurance	
3	Parts, Materials and Processes	
4	Environmental Compatibility	
5	Reliability Engineering	
6	System Safety	
7	Configuration/Change Management	
8	Integration, Test and Evaluation	
9	Risk Assessment and Management	2. Risk, Oversight, and Assurance
10	Independent Reviews	
11	Hardware Quality Assurance	
12	Software Assurance	
13	Supplier Quality Assurance	
14	Failure Review Board	3. Triage, Information and Lessons Learned
15	Corrective/Preventative Action Board	
16	Alerts, Information Bulletins	

Figure 3. MA process selection.

4. Impacts

With MBE, the MA roles will change from what is in the current processes. The following table details the differences between the current and future responsibilities. The future state are additional responsibilities to the current state responsibilities, so that the future state still contains everything in the current state. Some of the future state responsibilities are marked as N/A in the following table as those MA roles were out of scope for the process areas selected for this MAIW topic team charter.

Table 1. Current vs Future: MA Roles

MA Role	Current State Responsibilities	Future State Responsibilities
Government MA		
Program Manager Program lead that works with the contracting officer and staff the lead all the program work	• Defines the overall acquisition strategy and implementation • Creates and follows the program mission assurance plan (MAP) for internal government use • Defines the program acceptable risk level • Manages the request for proposal (RFP) and source selection processes	• Pre-award process discussions with industry on the strategic use of MBE • Instills MBE, as appropriate to the program needs and risk levels into the MAP • Instills MBE, as appropriate into the RFP and as a source selection criteria to ensure MA
System Engineer Defines the government SE program, approach, and resources	• Creates the program System engineering plan (SEP) that define the breadth and government implementation of SE, including modeling • Identifies the program specification and standards • Tailors SE/MA reviews and audits along with contract data requirements lists (CDRL)/data item descriptions (DiD)	• Plans for MBE work as part of the SE approach including setting the environment, infrastructure, training and readiness • Ensures specs and standards don't inhibit and do enable MBE tasks • Tailors SE/MA reviews and audits along with CDRLs/DiDs for MBE implementation
Project Engineer/Manager IPT member and counterpart of industry technical staff and cost-account managers	Follows the MAP, SEP, and other program documents to ensure MA • Tailors and uses the mission assurance baseline (MAB) to ensure MA tasks are completed	• Determines the MAB tasks that are MBE and conducts those tasks with the IPT and industry technical staff • Has MBE competency to ensure tasks are conducted
Industry MA		
Program/Project Manager	• Works with the government across the strategic, pre-award, and contract to ensure MA • Works with the government on the model contract that defines the specs/standards, CDRLs/DiDs including the IMP, IMS, and systems engineering master plan (SEMP)	• Ensures that MBE enables and improves MA by including it, as appropriate, into proposals, contracts, and the work • Program manager makes decisions based on MBE data to ensure MA based on the program's authoritative data source.

MA Role	Current State Responsibilities	Future State Responsibilities
Engineer System, requirements, configuration management, reliability, safety, design, integration, and test	• Conducts engineering activities resulting in products/services • Uses specifications, standards, guides, best practices, lessons learned to accomplish those activities	• Ensure that the inputs from their work come from the program data authority. • Outputs should go to the same repository, including any domains specific models.
Program Quality Assurance		
Program Quality Program quality management support to ensure contractual requirements are compliant	• Review program contract and flow down contract requirements • Develop program quality plans • Interface with customer on quality related issues	• N/A
Hardware Quality Hardware Quality management support to ensure known good products/services are delivered to customers	• Plan and implement inspection processes that ensure known good products • Perform physical configuration inspections/verifications • Perform product and STE verification and selloff • Disposition of nonconforming material in compliance with program requirements	• N/A
Software Quality Software Quality management support to ensure known good products/services are delivered to customers	• Plan and implement software inspection processes that ensure known good products/tests • Perform audits of software development processes • Perform software configuration inspections/verifications • Disposition of nonconforming software and test software in compliance with program requirements	• N/A
Material Quality Material Quality management support to ensure known good parts	• Interface with product quality • Develop supplier quality plans for programs	• N/A
Supplier Quality Assurance		
Supplier Material Quality	• Process compliance and resource for all material quality • Assist with development of supplier quality plans for programs • Assist with supplier quality portion of supplier selection	• N/A •

MA Role	Current State Responsibilities	Future State Responsibilities
Material Validation Ensure known good parts; material test and verification strategy for procured materials	• Engineering interface for obtaining key performance parameters (KPPs), translating into material verification plans by technology and/or part • Perform incoming inspections	• N/A
Infrastructure and Services	• Oversight for service providers	• N/A
Field Quality Engineering Field quality engineering, Inspection Processes	• On-site supplier quality oversight and insight • Source inspection/first article inspection – focal point for specification and process compliance at supplier • Supplier risk mitigation through execution of pre- and post-award reviews, product/process verification • Supplier control plan development/execution for key product characteristics and risk areas	• N/A

5. Recommendations

5.1 MBE Best Practices

1. MBE could enhance SE and MA efficiency and effectiveness by minimizing clerical tasks associated with integrating multiple document sets, providing a structured set of value-added engineering tasks that would potentially reduce risks earlier in the development, improve the confidence that the system is designed properly, minimize the need for "test" within increasing reliance on "analysis," potentially reduce needs for retest, and have greater confidence that the system will operate to stakeholder operational needs.

2. MBE would improve the engineering associated with requirements definition (model based documentation), requirements traceability, requirements analysis, and requirements verification and system validation.

3. DSMs that represent architectures, system, trade studies, risks, design, drawings, data, dataflow and other engineering data would be in a federated repository that is shared among stakeholders across the government and contractors. This provides a "single source of truth" that is kept current and aligned.

4. The DSM would provide a "digital twin" of the actual system so that as system variants are produced the digital twins reflect their "as built," "as delivered" design and configuration.

5. The DSM made up of a defined document taxonomy, data taxonomy, and set of digital artifacts can be shared and used proactively for engineering developments, independent product and process team work, reviews, audits, assessments, as well as verification, validation, and acceptance. This level of connected data enables improved engineering and program management confidence to levy go/no-go decisions to proceed with work, to test/re-test, and for acceptance.

6. Traditional paper based and scheduled CDRL documents are increasingly replaced by DSM views and reports enhancing engineering efficiency, effectiveness, and enabling improved risk assessment and handling, along with program decision-making.

7. For higher rate production, and product line management, such as CubeSats, MBE would likely provide a significant benefit by managing requirement sets, requirements traceability, enhance requirements analysis and validation while ensuring that verified digital twins exist for each product line item.

8. Decision-making starts on the modeler side, but transfers to the leader to understand the modelers information. Leaders need to make their decisions based on the model and need to understand the design completely. The modelers need to be able to effectively communicate their model to the decision-makers and not be disconnected from what is happening in the rest of a given program.

5.2 MA Recommendations

One common recommendation across all the process areas is to iterate and redefine the model throughout the program systems engineering vee, and establish a process to reevaluate or update the model throughout the lifecycle, and do so early and often.

Requirements Recommendations:
Consider reducing design verification tests (including retests) if model-based verification can be proven effective, benefits include reduced risk exposure during ground testing, lower cost and shorter cycle times, as well as reduced stress due to over testing. Build verification and workmanship checks should be retained irrespective of model-based practice maturity.

MA model-based projects should consider ways of verifying the quality and completeness of this model traceability and government stakeholder should drive accountability by requesting traceability metrics and reporting in milestone reviews. Best practices for verifying appropriate traceability and expected verification by mission class should be considered for a future TOR.

Recommend government consider ways of reducing document deliverables and replace with model deliverables. There are many challenges in this area but advantages include potential for the government to own more of the technical baseline and reduce vendor lock-in for long lifecycle programs.

Recommend programs that are transitioning from document to model-based work, review and evaluate their decision-making process in favor of model-based decision-making over document-based decisions. Challenges include establishment of go/no-go decision criteria and an efficient implementation through automation over manual review of models.

Design Assurance:
Consider using design data to determine adherence to process steps to provide better process rigor, rather than leaving process adherence up to engineers to understand and tailor, to enforce an assured process flow.

Consider implementing automated model checking and problem solving to flag small errors early for solution.

To improve MA, reduce likelihood of error, and reduce the burden on engineers, consider automated performance budget management that are updated with each design release. MA personnel should consider the pedigree of this automation, however, to reduce the likelihood of systemic errors in budget having consequences to the program and mission.

Reliability Recommendations:
Reliability plans and approach should consider leveraging model-based use-cases to deliver estimates of success probability of operationally-necessary uses rather than a single system-wide requirement like mean time between failure (MTBF) and availability.

Reliability plans should consider automated methods for failure modes and effects analyses early in the concept definition phase through detailed design—to give early reliability input to trades.

Consider linking FRACAS data, as it is collected, as part of the larger DSM to ensure timely identification and decimation of new failure modes and effects.

Reliability programs should utilize model-based use cases to develop more precise measures of effectiveness.

Build your reliability methods into your model. Have the data elements there, so you can build upon your model. Emphasize that this can be done early and continuously in the program. Take failure/test data and link it into your descriptive data of your model.

Safety Recommendations:
Involve safety engineers early in the design process as the model is being developed. Safety constraints and off-nominal paths can be added to use cases to help evaluate risk early on while the design is still fluid.

Recommend adding attributes to the system model that help safety analysis including: failure data and output produced when the failure occurs along with an estimate of the probability of failure.

Recommend all known single-point failures should be identified with dedicated attributes within the system model; this allows for automated lists of potential single point failure to be generated and thus mitigated.

Programs transitioning to model-based approaches should consider collaborating early with the system safety community including safety experts and range safety personnel to ensure the safety-oriented model outputs and their presentation meet the expectations and needs of this community.

Configuration Management:
SE, collaborating with stakeholders, should define the areas of the DSM that can be created, viewed, updated, and deleted to ensure model integrity and account for protection of intellectual property and information assurance.

MA personnel need to verify that the system model is under configuration control, even if relatively few users are involved with maintaining the model, to avoid model corruption, uncertainty, and data loss.

A model-based program needs to have a clear CM strategy to ensure baseline integrity. Artifacts may be in several locations (e.g., DOORS, Rhapsody, or Documents) and only one of these can hold the baseline that the others synchronize to.

Independent Reviews
Programs implementing model-based reviews should extend these reviews beyond the review of technical content to the completeness and validity of the model itself versus what is expected for the current milestone.

MA personnel need to verify that model checkers have been validated and are as effective as claimed, especially if manual review is reduced according to model checker maturity.

MA personnel should review model volatility as a risk indicator as well as the volatility of changes to model checkers. If models and checkers are continually changing together, version misalignment and errors are likely to occur and thus undermine the value of model checking.

5.3 Approach for Executing MBE

1. Adopt model-based tools with standard, open application programming interfaces (APIs) to allow for easier data exchange and reporting.
2. Invest in visualization and reporting technology development to maximize the value of data rich models. (Visualization is an investment because you have to try things, and sometimes fail until you get to what works.)
3. Invest in development of model-based tools that enforce rules for process execution and capture process tailoring and rationale.
4. Don’t try to model everything! Start small and model what you have, then expand as needed. Start capturing metrics on when the model was useful and needed.

5. Adopt automated audit and dash-boarding of model state and status.
6. Adopt single-source-of-truth paradigm for the system.
7. Consider the method to deliver the model to the government at the beginning of the contract. For example, adopt a baseline hosted on Government infrastructure.

6. Conclusions

6.1 Lessons Learned Summary

Table 2 below contains an abridged summary of all the lessons learned from each of the subsections in this document. These lessons learned were generated from the 2017 MAIW team discussion.

Table 2. Summary of Lessons Learned

	Lessons Learned
1	MA personnel need to verify that model checkers have been validated and are as effective as claimed. Especially if manual review is reduced according to model checker maturity.
2	Don't assume that just because you can get the data into the model that you can get it out.
3	Modeling helped identify both the requirement set and the UCs and stabilized the change of those fairly quickly.
4	The provided model views were insufficient to meeting the needs of the stakeholders; custom views were necessary.
5	With the lack of open system interconnects and system interfaces it's hard to have portable model data.
6	You need to define an intermediate format for the model data. It might not be the tool vendors you want to look to for that. You create a single authoritative source.
7	Consider the data exchange approach and define a tool independent (intermediate format) for your data.
8	Use the right tool for the job. This is in regards to showing a visual for the data. Sometimes is "X" software…other times it's excel…etc. Determine processes to define these forms but all come from the single source of truth.
9	Start small with what you have. (physical systems, UC's block diagram, etc.)
10	By modeling the systems functionality and CONOPS, identify disconnect between component requirements and correct mental models in the CONOPS.
11	Insure the processes are there to update the model to ensure feedback is happening and the processes are being executed.
12	Make sure that the models are not just within SE. The model integrates with discipline engineering models.
13	Model the process use cases and stakeholder interaction use cases.
14	Develop champions by demonstrating value early.
15	Capture ontology and style guide first and then add to it as you go. Plan for model organization and usability.
16	Need to make sure your modeling team has enough resources to be responsive.
17	It is important to collect metrics for value added of MBE such as number of engineering escapes identified.
18	There must be a single authoritative source for any individual piece of information. To reduce the likelihood of introducing errors that can lead to MA escapes.
19	When sources of truth are distributed, a disciplined process is needed to maintain CM of the technical baseline. This is needed to avoid CM escapes. Can use an issue tracking system such as ClearQuest or Jira.
20	Unit level SW test cases should verify alignment between code and model.
21	If you don't have the model up to date with the latest system information, then change assessment will be incorrect. Can lead to MA escapes.

6.2 Conclusions

In summary, we have identified six key mission assurance process areas which will likely be impacted by the shift to model-based work: requirements analysis & validation, reliability, system safety independent reviews, design assurance, and configuration management. In the body of this document we have summarized our best practices and lessons learned for systems engineering, program management, and

mission assurance leaders to consider for model-based programs. We also describe a set of implementation recommendations in order to retain and enhance mission assurance during the transition from document-based to model-based work.

The appendices which follow capture the detailed work for each mission assurance area along with detailed recommendations. For each area in-turn, we define the current state of each area, the expected future model-based state, and describe both the risks and opportunities for mission assurance practitioners to consider when planning, and executing mission assurance activities on future model-based projects.

7. Acronyms

AADL	Architecture Analysis and Design Language
API	Application Programming Interface
ATR	Aerospace Technical Report
CCB	Configuration Control Board
CDR	Critical Design Review
CDRL	Contract Data Requirements List
CI	Configuration Item
CM	Configuration Management
CMP	Change Management Program
COA	Courses of Action
CONOPS	Concept of Operations
COTS	Commercial Off-the-Shelf
DAU	Defense Acquisition University
DE	Digital Engineering
DiD	Data Item Descriptions
DoD	Department of Defense
DoDAF	Department of Defense Architecture Framework
DOORS	Dynamic Object Oriented Requirement System
DSM	Digital System Model
EIA	Electronic Industry Association
ETA	Event Tree Analysis
FCA	Functional Configuration Audit
FDIR	Failure Detection Isolation and Recovery
FFP	Firm Fixed Price
FMEA	Failure Modes and Effects Analysis
FMECA	Failure Modes and Effects Criticality Analysis
FRACAS	Failure Reporting and Corrective Action Systems
FTA	Fault Tree Analysis
GOTS	Government Off-the-Shelf
GPRA	Government Performance and Results Act
HW	Hardware
IBD	Internal Block Diagram
IC	Intelligence Community
ICD	Interface Control Drawings
IEEE	Institute of Electrical and Electronics Engineers
INCOSE	International Council on Systems Engineering
KPP	Key Performance Parameter
IMP	Integrated Master Plan
IMS	Integrated Master Schedule
IPT	Integrated Product Team
ISO	International Organization for Standardization
LDM	Logical Data Model
LRR	Launch Readiness Review
MA	Mission Assurance
MAB	Mission Assurance Baseline
MAP	Mission Assurance Plan
MAIW	Mission Assurance Improvement Workshop
MBE	Model-Based Engineering

MTBF	Mean Time Between Failure
MBSE	Model-Based Systems Engineering
MDA	Missile Defense Agency
MIL-STD	Military Standard
MoM	Manager of Managers
NASA	National Aeronautics and Space Administration
ODASD	Office of the Deputy Assistant Secretary of Defense
OMG	Object Management Group
OSLC	Open Services for Lifecycle Collaboration
PCA	Physical Configuration Audit
PDR	Preliminary Design Review
PES	Physical Exchange Specification
PHA	Preliminary Hazard Analysis
PHL	Preliminary Hazard List
PM	Program Manager
PPT	PowerPoint
RFC	Request for Change
RFP	Request for Proposal
RFV	Request for Variance
ROI	Return on Investment
SE	Systems Engineering
SEMP	Systems Engineering Management Plan
SEP	System Engineering Plan
SMC	Space and Missile Systems Center
SME	Subject Matter Expert
SRR	System Requirements Review
STE	Special Test Equipment
SW	Software
SysML	Systems Modeling Language
TOR	Technical Operating Report
TRR	Test Readiness Review
UML	Unified Modeling Language
VCRM	Verification Cross Reference Matrix

8. References

1. *Automating the Synthesis of AltaRica Data-Flow models from SysML*, Chapter 15: Reliability, Risk, and Safety, CRC Press, 2009.

2. Bernardi, S., J. Merseguer,and D Petriu, Dependability modeling and analysis of software systems specified, ACM Comput, Surv, 2011.

3. Bjorndahl, W. D., M. M. Simpson, L. J. Vandergriff, and H. Wishner, *Chapter 6 - Design Assurance (Mission Assurance Guide),* The Aerospace Company, 2007.

4. Blouin, Dominique and Holger Giese, *Combining Requirements, Use Case Maps and AADL Models for Safety-Critical Systems Design*, 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2016.

5. *Case Study of Successful Complex IT Projects*, British Computer Society, August 2006.

6. Costa, Bruno, Paulo F. Pires, Flávia C. Delicato, Wei Li, and Albert Y. Zomaya, Design and Analysis of IoT Applications: A Model-Driven Approach, *2016 IEEE 14th Intl Conference on Dependable, Autonomic and Secure Computing*, 8-12 Aug. 2016.

7. Committee On Science and Technology, *Investigation of the Challenger Accident,* Washington: House of Rpresentatives 99th Congress 2nd Session, 1986.

8. *Configuration Management for Defense Contracts, Technical Report EIA-649-1,* SAE International, 2014.

9. *Defense Acquisition University*, Glossary 11th Edition, Department of Defense, January 7, 2015, https://dap.dau.mil/glossary/pages/default.aspx

10. EIA, I., *15288, Systems and Software Engineering - Systems Life Cycle Process,* ISA ESA, 2015.

11. Enes, P., Build and Release Management, Supporting development of accelerator control software at CERN, *Master of Science in Computer Science*, Norway: Norwegian University of Science and Technology, Department of Computer and Information Science, February 2007.

12. Feiler, Peter and Julien Delange, Archgitecture Fault Modeling with the AADL Error-Model Annex, *40th EUROMICRO Conference on Software Engineering and Advanced*, 2014.

13. Fitzgerald, D., *MBE Update,* MIT Lincoln Laboratory, Graphics Adapted from Anark Corporation, 2016.

14. Freitag, Tom, B. H., *The Role of Independent Assessments for Mission Readiness,* Crosslink, 2007.

15. Gabsi, Wafa, B. Zalila and M. Jmaiel, EMA2AOP: From the AADL Error Model Annex to aspect language towards fault tolerant systems, IEEE 14th International Conference on Software Engineering Research, Management and Applications (SERA), 2016.

16. Guararo, S., G. Johnson-Roth, and W. Tosney, *Mission Assurance Guide, Technical Operation Report,* TOR-2007(8546)-6018 Rev. B*,* The Aerospace Corporation, 2012.

17. Harmann, R., Digital Environment and MBSE Progress at Airbus Space, *3rd NASA/JPL Symposium and Workshop on Model Based Systems Engineering,* Airbus, 2017.

18. Hecht, Myron and Daniel Winton, SysML Reliability Modeling of Ground Based Systems with Virtualized Architectures, *Manhattan Beach: Ground System Architecture Workshop*, 2014.

19. Hecht, Myron, A Model Based Systems Engineering Approach to Resiliency Analysis of a Cyberphysical System, *Gaithersburg: IEEE International Symposium on Software Reliability Engineering*, 2016.

20. *IEEE 15288.1 Application of Systems Engineering on Defense Programs,* IEEE, 2014.

21. International Council on Systems Engineering (INCOSE), *Systems Engineering Handbook, A Guide for Lifecycle Process and Activities, 4th Edition, INCOSE-TP-2003-002094,* Wiley, 2015.

22. Kostoff, R., *Research program peer review: purposes, principles, practices, and protocols,* Arlington: Office Of Naval Research, 2004.

23. Kolcio, Ksenia and Lorraine Fesq, Model-based off-nominal state isolation and detection system for autonomous fault management, *Big Sky Montana: 2016 IEEE Aerospace Conference*, March 5, 2016.

24. Liu. X., Z. Wang, Y. Ren, and L. Liu, Modeling method of SysML-based reliability block diagram, International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC), 2013.

25. MA-TOR-8591-21, *Mission Assurance Guidelines for A-D Mission Risk Classes,* June 3, 2011.

26. *Military Handbook: Configuration Management Guidance (MIL-HDBK-61A),* 2001.

27. *Mission Assurance Guidelines for A-D Mission Risk Classes.* TOR-2011(8591)-21, The Aerospace Corporation, 2011.

28. *Model-Based Systems Engineering (MBSE) Guidance for Government Acquired Programs,* ATR-2016-02402 Rev A, The Aerospace Corporation, 2016.

29. Okamura, Hiroyuki, Tadashi Dohi, Shin'ichi Shiraishi and Mutsumi Abe, Composite Dependability Modeling for In-vehicle Networks, ISBN 978-1-4577-0375-1/11: *Proceedings of the IEEE/IFIP 41st International Conference on Dependable Systems and Networks Workshops*, 2011.

30. Saunders, Mark, J. O., *Nobody's Perfect: The Benefits of Independent Review*, September 1, 2009, https://appel.nasa.gov/2009/09/01/nobodys-perfect-the-benefits-of-independent-review/

31. NASA, Space Systems Engineering Technical Reviews Module, 2008, Spring.

32. NASA, The Project Lifecycle Module (Space Systems Engineering, version 1.0) 2008, Spring.

33. NASA, *NASA Independent Verification & Validation Program Value Report,* NASA 2009.

34. Nelson, Nicola, G. A., *Independent Reivew Process -- Overview and Best Practices,* ATR-2009(9369), The Aerospace Corporation, 2009.

35. Nguyen, T., Model-Based Version and Configuration Management for a Web Engineering Lifecycle, *Proceedings of the 15th International Conference on World Wide Web* (pp. 437-446), Edinburgh, Scotland: ACM Press, May 23-26, 2006. doi:http://doi.acm.org/10.1145/1135777.1135842

36. NoMagic, Inc., MagicDraw Users Manual, Version 18.1, Chapter 12, 2015.

37. *Open Services for Lifecycle Collaboration (OSLC)*, 2017, http://open-services.net.

38. Oster, F. Applying SysML and a Model-Based Systems Engineering Approach to a Small Satellite Design, *Advances in Systems Engineering, American Institute of Aeronautics and Astronautics*, Reston, VA, 2016.

39. Pawlikowski, M., Mission Assurance--A Key Part of Space Vehicle Launch Mission Success, *High Fronteer*, 6-9, 2008.

40. Sauser, B. J., Attributes of Independent Project Reviews in NASA, *Engineering Management Journal, 18*(4), 11-18, (2006, December), http://www.boardmansauser.com/downloads/2006SauserEMJ.pdf

41. Schulte, M., Integrating Your Engineering Data, *Open Services for Lifecycle Collaboration (OSLC) Summit,* La Jolla, CA: Object Management Group (OMG), 2015. http://www.omg.org/news/meetings/tc/ca-15/special-events/OSLC_Summit_agenda.htm

42. *SE Vision 2025*, (n.d.), INCOSE: http://www.incose.org/AboutSE/sevision

43. Space and Missile Sysems Center, *SMC Systems Engineering Primer & Handbook, Volume 1, 4th Edition,* Air Force Space Command, 2013.

44. Space and Missle Systems Center, *Configuration Management (SMC-S-002),* Air Force Space Command, 2008.

45. Witt, Rouven, Andrew Kennedy, and Jens Eickhoff, (Boston, MA), *Implementation of Fault Management Capabilities for the Flying Laptop Small Satellite Project through a Failure-Aware System Model*, AIAA Infotech@Aerospace (I@A) Conference, August 19-22, 2013:

46. Zimmerman, P., A Framework for Developing a Digital System Model Taxonomy, *NDIA,* Springfield, VA, 2015.

Appendix A. MA Area: Requirements Analysis and Validation

A.1. Literature Search and Review

The intent of this MAIW TOR is to review the MA community accepted definition of the requirements process (including requirements analysis and requirements validation) then provide guidance on how a mission risk adjusted MA enabled MBE approach brings value to the enterprise. The following key documents were identified and will be discussed:

1. ISO-EIA-15288 "Systems and Software Engineering — System Lifecycle Processes," 2015-05-15 (IEEE) 15288.1 "Application of Systems Engineering on Defense Programs," December 10, 2014. This document is one of the accepted across the government as part of the standards profile necessary for mission assurance on government programs.
2. TOR-2011(8591)-21 Mission Assurance Guidelines for A-D-Mission Risk Classes, June 3, 2011. This MAIW written document is the basis for this TOR. This TOR provided the tenants and working title for the appendix "Requirements Analysis and Validation."
3. ATR-2016-02402 Rev A, "Model-Based Systems Engineering (MBSE) Guidance for Government-Acquired Programs", March 14, 2016. This document provides that more current thinking on how MBSE may be adopted.
4. A framework for developing a digital system model taxonomy
   Philomena Zimmerman, 18th Annual NDIA Systems Engineering Conference, Springfield, VA, October 28, 2015. This document provides relevant and current concepts related to MBSE from DOD ODASD (SE) that sets policy and regulation for systems engineering.

Implementing Requirements Analysis & Validation for Mission Assurance in a MBE Environment.

It's helpful to set requirements analysis and validation in context of mission assurance and then systems engineering and finally model-based systems engineering.

The Mission Assurance Guide (MAG) sets the concepts for mission assurance from which the DOD, intelligence community and civil agencies base their mission assurance program on. Figure 4 is from the MAG and illustrates the relationship between systems engineering tasks and mission assurance. This diagram is based on the systems engineering VEE that has been popular to illustrate decomposition of stakeholder requirements on the left side to manufacturing and production, and then integration of components on the right side, culminating in verification and validation of systems. Figure 4 shows all the processes identified in TOR-2011(8591)-18 [24] on which this document is based.

The MAG, Figure 4, extends the systems engineering Vee. On the left, it discusses program assurance that sets up the mission assurance program. Program assurance would include the planning activities necessary to apply MBSE/MBE to the work to be performed. The MAG diagram on the right then shows that the results are mission success stemming from mission assurance activities that stem from proper systems engineering with an emphasis on the tasks necessary for the mission to be successful. Figure 4 also shows how the engineering work extends past verification and validation through operations and sustainment, and finally how that flows back into defining new missions and the need for updated program assurance.

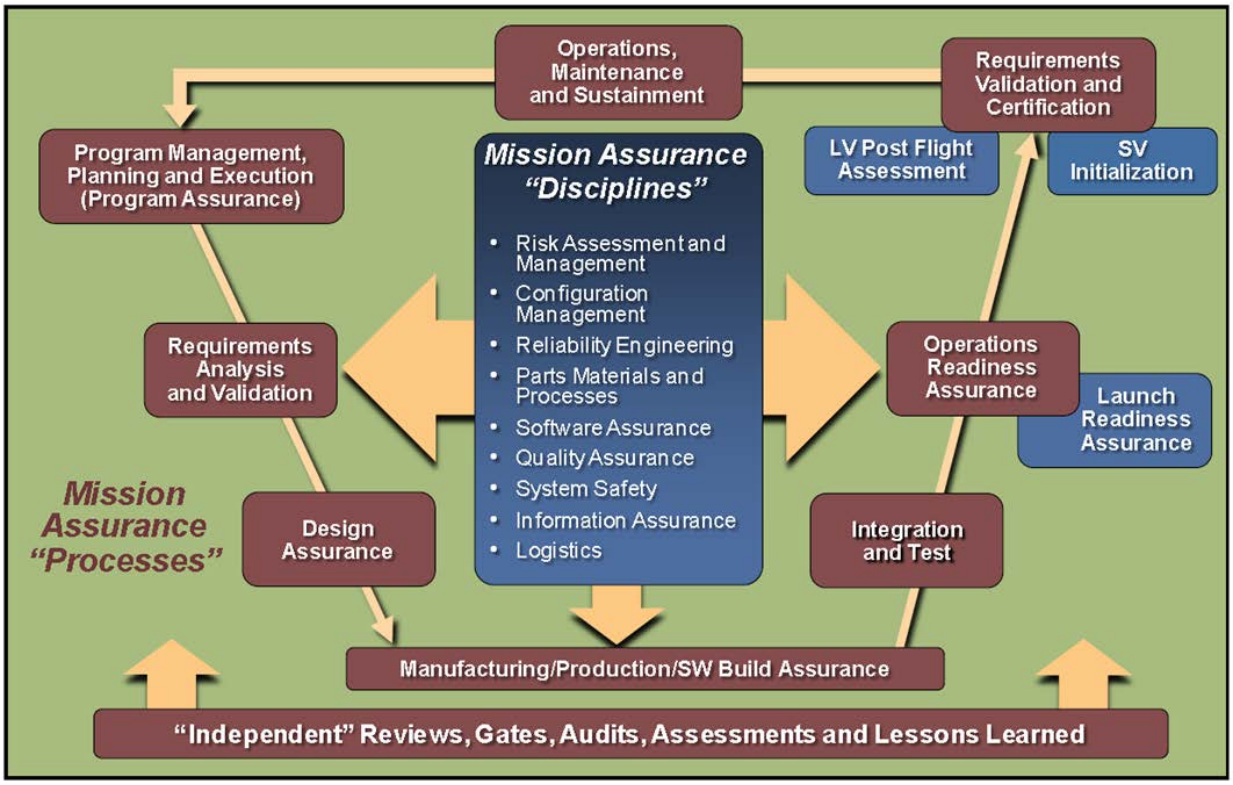

Figure 4. Mission assurance through systems engineering.

The Aerospace Corporation produced ATR-2016-02402 Rev A [28], that provides information on benefits and challenges for using MBSE in acquisition. Figure 5 shows the systems engineering Vee as it would look with major steps/gates (grey boxes) and associated engineering work (red boxes) contributing to integrated digital systems models that provide a single source of truth to produce trusted views, reports, and documents such as requirement specifications, traceability analysis reports, impact analysis reports, interface control documents, training documents and provide output to other tools such as cost and schedule estimation tools or risk management data tools.

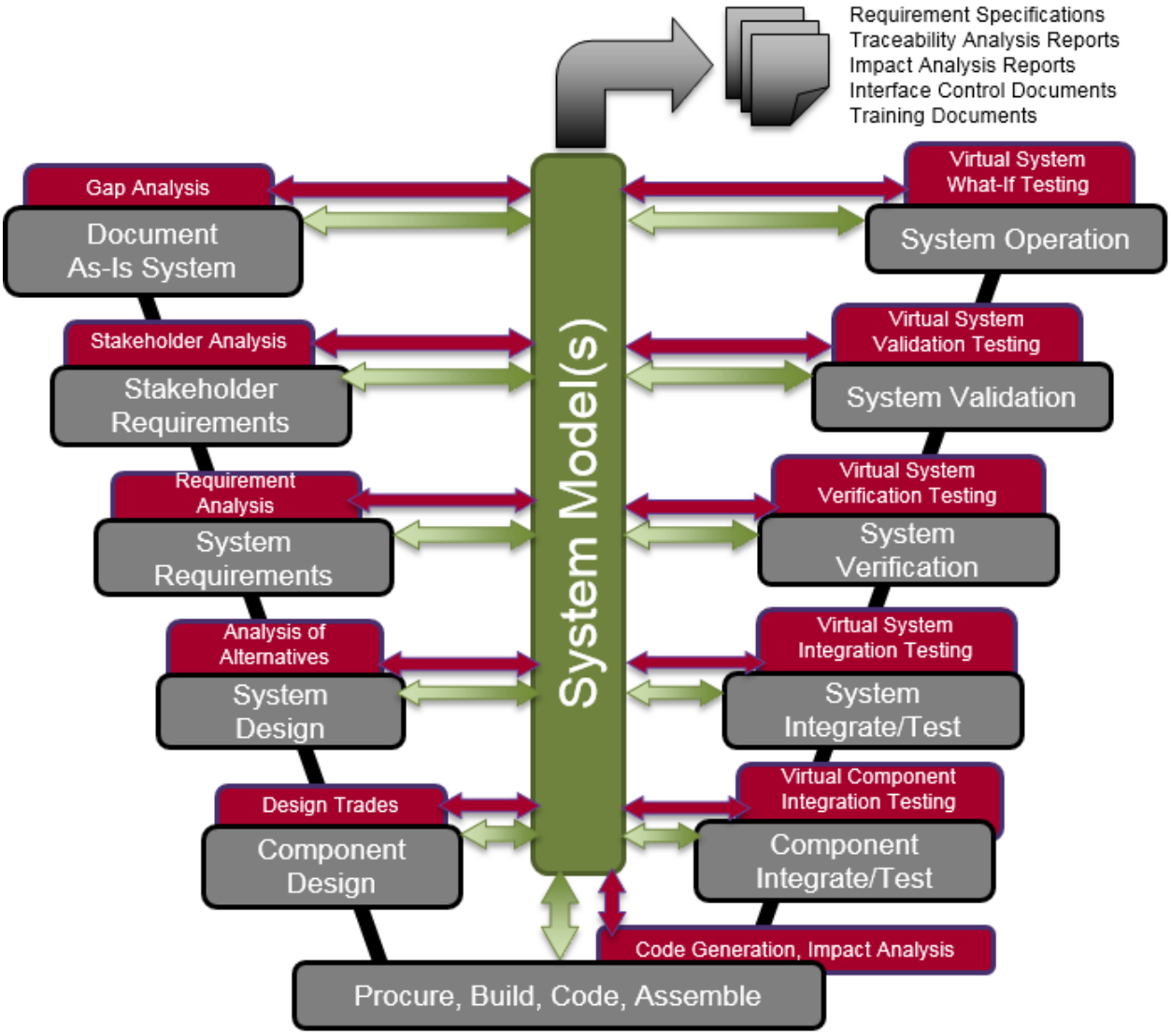

Figure 5. Concepts related to MBE.

Definition of Requirements Analysis and Validation

The most widely accepted definition of requirements, analysis, and validation appears in ISO EIA 15288 (International Standards Organization Electronic Industry Association) "Systems and software engineering — System lifecycle processes" and the IEEE 15288.1 [20] "Application of Systems Engineering on Defense Programs." These documents provide the requirements for conducting SE and are contractually binding on many space enterprise contracts since they've been adopted by the DOD, National Aeronautics and Space Administration (NASA), Missile Defense Agency (MDA), and the intelligence community as essential for MA. These agencies collaborated on a MA framework that has identified a profile of recommended specifications and standards for program application.

Figure 6 from IEEE 15288.1 [20] identifies systems requirements, systems analysis, and validation as technical processes within systems engineering. The IEEE 15288.1 [20] document identifies tasks to be accomplished for each process. Those tasks can be manual, assisted by automation, or automated with engineering oversight. The requirements analysis and validation tasks were examined for MA impact if conducted by MBSE

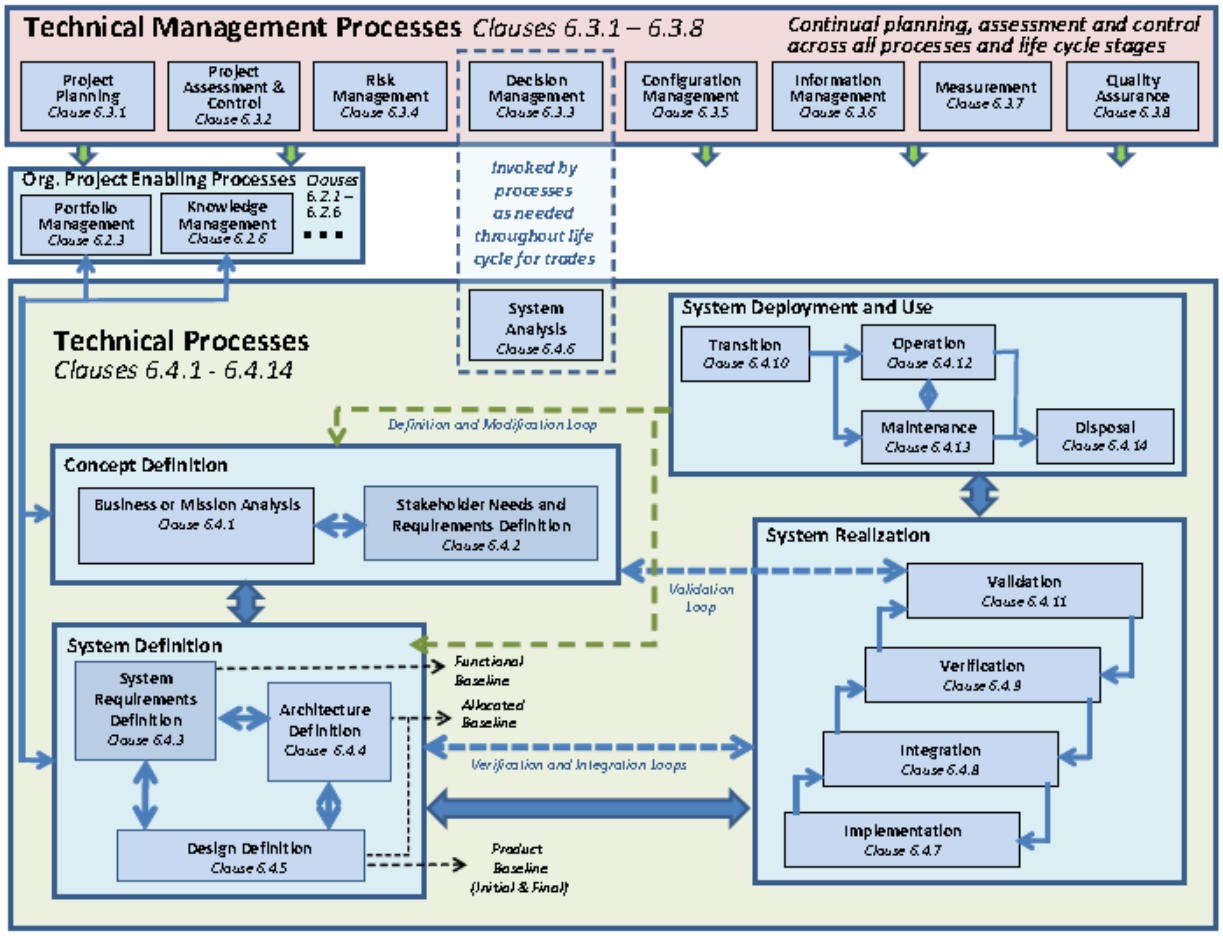

Figure 6. SE planning, assessment, and control interfaces with the SE technical activities.

Requirements Analysis and Validation Processes – Key MA Components

A previous MAIW group produced TOR-2011(8591)-21 [27], which defined key processes to MA and their tailoring for mission risk. The requirements, analysis, and validation process was identified as key MA process. That TOR provided a series of matrices that defined tasks tailored by risk levels shown in Figure 7 as an example below. Class A missions are those that have the least amount of acceptable risk while the Class D mission allows the greatest amount of acceptable mission risk. The risk class definitions are provided in TOR-2011(8591)-21 [27].

The TOR groups the "Requirement" column in Figure 7:

1. Requirements development process
2. Evaluation of requirements quality
3. Requirement traceability
4. Mission analysis validation

Requirement	Class A	Class B	Class C	Class D
Requirements Development Process	Required by contract with deliverables customer approved. Contract requires prime contractor to flow to subs for major elements.	Same as Class A	Contract requires prime contractor's best practices. Customer reviews evidences.	Recommended (Not Required by contractor) Discretion of SV/Payload developer that accepts risk. Level of effort determined by developer commensurate with program and best practices.
Evaluation of requirements quality (Measurable, verifiable, etc.)	Independent customer assessment conducted to the unit level	Same as Class A	Contractor assessment usually performed to the system level.	Recommended. Discretion of developer based on requirements fidelity
Evaluation of requirements traceability conducted	Independent customer evaluation conducted down to the unit level	Same as Class A	Contractor conducted in accordance with their best practices mitigating development risk.	Discretion of developer. Not always conducted. Dependant on fidelity of flow down
Mission effectiveness evaluation	Contractor evaluation; independent assessment conducted by customer	Same as Class A	Recommended, but not required. Dependant on performance expectations.	Discretion of develop based on performance expectation
Cost and schedule evaluation	Contractor required to conduct evaluation; independent assessment conducted by customer	Same as Class A	For CP contract contractor required to conduct evaluation and customer performs independent evaluation at higher level. For FFP Contractor best practices.	Not usually performance, as contract is almost always FFP. Dependant on contract best practices.
Mission Analysis Validation	Required by contract and customer approved	Same as Class A	Negotiated at discrete intervals during development	Recommended. Develop Discretion to conduct and evolve concept and conduct trade for mission success

Figure 7. A1-3 Matrix-requirements analysis and validation.

Reading through the TOR-2011(8591)-21 [27] document:

1. The requirements development process is discussing requirements definition and documentation
2. Evaluation of requirements quality is discussing requirements analysis
3. Requirement traceability is discussing up and down tracing of
4. Mission analysis validation is discussing validation of user requirements

Each of the four components will be further examined by comparing them to the relevant tasks defined in IEEE 15288.1 [20] and will then be examined to see to what extent they can be MBE enabled. The mission risk levels will be addressed as the four components are examined with an overarching idea that MBE improves MA for Class A and B programs and may enable improved MA for lower risk Class C and D programs, if MBE is implemented.

A.2. Expected Benefits and Outcomes

Requirements Traceability

The IEEE 15288.1 [20] tasks under Paragraph 6.4.2.4 System Requirements Definition process outputs has several tasks that require bi-directional traceability from operator/user desired capabilities to system requirements and constraints through systems design, down to the lowest design levels. The untraced requirements are detected earlier lowering the cost and design risk. While IEEE 15288.1 prescribed specific traces be performed to ensure appropriate design and implementation, MBE enabled SE allows more traceability to ensure that the design closes with requirements and operations.

TOR-2011(8591)-21 [27], in matrix A1-3 illustrated above requires traceability as well for MA for Class A and B risk postures. MBE tools and methods may enable routine requirements traceability for Class C

and D missions. This may be a great benefit to system product lines with increased standardization; for example, CubeSats.

ATR-2016-02402 Rev A [28] provides guidance information and mentions the benefits of requirements traceability using MBE tools and methods. Quick response to adapt to changing missions and threats demand an agile response to needs and impact requirements definition and change. Changes in requirements can be traced to the system elements impacted, the mission objectives affected, and the resources used to provide them. MBE tools may then improve cost and affordability attributes and lifecycle trade space. MBE can assist in developing the system concept of operations (CONOPS) by generating operational views and use case reports to ensure they are traced to system and interface requirements. Tracing requirements to use-cases and operational scenarios which provide a context for the requirement, enables better understanding of the requirement. Tracing requirements to system elements and suppliers allows impact analysis in replacing a component of a system (e.g., what if a spacecraft reaches end-of-life, a supplier goes out of business). It can also ensure that use case components trace to items within the system configuration. MBE enables traceability among requirements, reference models, and design risks, assisting in risk management. Leveraging model features such as systems modeling language (SysML) parametric and constraint checking, and satisfy relationship, and the constraints on allocation of performance parameters may be enforced to meet the system requirements. MBE enables references to an external document that may provide additional information, e.g., an algorithm document that provides theoretical analysis basis for allocation of performance parameters may refer to an external document using a link in the model.

Requirement Document Generation

IEEE 15288.1 [20] tasks. This standard lists many items that are required to be created, defined, and documented under the requirements review exit criteria. For organizations implementing this standard, "create," "define," and "document" are key exit criteria tasks/products that could be greatly facilitated by MBE and would provide a more trusted set of views or documents since they would step from the MBE "single source of truth." In addition, specific queries, beyond those required, can be conducted to add to engineering confidence and enhance overall mission assurance work and results.

TOR-2011(8591)-21 [27] does recommend differing levels of documentation for risk classes to ensure that MA tasks were conducted commensurate with accepted mission risk. This postulation is based on the existing, mostly manual and paper based approach to performing engineering and MA tasks. By having the model as the single source of truth and requirements documents generated from the model, everyone knows they have the current requirements documentation with no risk of inconsistency across development efforts. While examining the matrix A1-3 required tasks, many of these could benefit from MBE automation and may allow more MBE enabled MA tasks for Class C and D satellites and especially for higher production rates such as CubeSats.

ATR-2016-02402 Rev A [28], shows that models can be used to generate specifications, interface documents, and configurations directly from the model. An integrated set of model data provides a unified view of the system and components and maintains that consistency as the data evolves. Documents generated from modeling tools result in systems definitions that are capabilities/constraint/ performance/algorithm based and permit the refinement of requirement statements to more clearly clarify the intent, particularly for functional requirements. Requirements documents generated from MBE tools provide documents that use "shall" specification language. Since the resultant document is based on the MBE data sets it's intended to:

1. Reduces risk of misinterpretation of textual requirements.

2. Reduces risk of ambiguous or inconsistent specification of requirements or design between documents or even within a single document.

3. Enables frequent and early detection of requirement defects, circular dependencies, or capability shortfalls.

4. Enables "what-if" exercises and facilitates more effective impact analyses of potential requirement changes.

5. Define the "digital twin" of the system and maintain its currency and accuracy across all forms of representation and report.

MBE may be used to generate DODAF (Department of Defense Architecture Framework) views or other views to ensure that designs are traced to requirements. Changes are propagated through the models ensuring consistency and enabling impact analysis. Modeling tools can generate spreadsheets, populate dashboards, and populate web pages that are consistent with the modeling data which provides a single source of engineering truth. Stakeholder-level deliverables and viewpoints may be made available, and will be consistent-with, all other documents generated from the MBE environment providing a "single source of truth."

Requirement Analysis

IEEE 15288.1 tasks [20]. Paragraph 6.4.3.4 System Requirements Definition process outputs. This paragraph lists outputs that are necessary to complete system requirements definition and include such items that would be facilitated or more efficiently performed with MBE:

1. Ensuring the requirements included all requirements that also includes all functional, non-functional, interface and performance requirements and constraints imposed by each specialty function.

2. Ensure that lower-level requirements satisfy higher-level capabilities, requirements and constraints.

3. Ensure system interoperability needs.

4. Document decision trade studies.

In TOR-2011(8591)-21 [27], the properties of a requirement change from the paper-based "Shall" with criteria such as unambiguous and specific to a model-based definition that is capabilities, constraint, performance, and algorithm based ensures that the MBE requirements are appropriate. The collection of MBE requirements and design can be more easily reviewed for mission effectiveness via the use cases and requirement validation. As the requirements and configuration change then the model provides the information necessary to maintain the mission and system performance baselines. MBE based system definitions could benefit cost and schedule evaluation in that it provides a consistent and coherent definition making it easier to estimate/evaluate associated costs and schedules for a paper-based system definition. For higher rate production, such as CubeSats, MBE would likely provide a significant benefit by managing requirement sets.

ATR-2016-02402 Rev A [28], reveals that MBE improves effectiveness and efficiency across the system development by ensuring that the contributors have the same information at the same time and enables a distributed team to accomplish the work. It enables more efficient management of trade exploration ensuring:

1. More implementation trades can be conducted—producing a better product. MBE permits integrated design and implementation trades and impact analysis in a more complete and faster fashion than using paper-based analysis.

2. More risk mitigation trades can be conducted—reducing cost and schedule risk. From an aspect of risk management, MBE permits integrated design and implementation trades and impact analysis in a more complete and faster fashion than using paper based analysis.

3. Improved systems assurance and enhanced system design integrity.

The effects of planned and unplanned changes are more easily and accurately determined and are readily observable.

1. Changes in requirements can be traced to the system elements impacted, the mission objectives affected, and the resources used to provide them.

2. The cost and schedule impacts of changing test schedules or venues due to resource conflicts can be determined.

3. Changes in budgets can be supported by trade studies showing the impacts of different approaches to reduce cost.

Automated consistency checks on the data can be implemented to ensure the quality of the information. The MBE data sets capture the history of the program in addition to the current status.

1. Enhances the ability to capture, analyze, share, and manage the information.

2. The rationale for decisions can be captured so that they can be revisited, if conditions change.

3. Requirements, trade studies, designs, and rationale are captured with their relationships for possible reuse.

4. The data element pedigree is captured and accessible to all model users.

Requirement Validation

Requirements validation ensures the proper set of system requirements by comparing them against the mission and system concept of operations. It can be viewed as driving SE, by taking the end in mind, and thus driving both the MA approach and MBE tasks that would result in positive validation.

This document, IEEE 15288.1 tasks [20], provides requirements for both validation and verification. Validation provides objective evidence that the capability provided by the system in the intended environment, complies with stakeholder performance requirements. Early validation activities provide confidence in the system's ability to achieve its intended mission or use under specific operational conditions. Final validation involves operational testing on a production-representative system in an

operationally realistic environment. MBE would enable early validation activities improving the systems design and confidence that would meet operational needs.

Regarding TOR-2011(8591)-21 [27], MBE enhances validation planning allowing it to occur earlier in MBE based engineering since the requirements and configuration are defined earlier with more cohesion. If the models contain and decompose/allocate the KPPs to the configuration, then the requirements validation becomes more straight-forward. Having the MBE use cases ensures that the proper test environments and scenarios are used for validation.

In ATR-2016-02402 Rev A [28], this ATR shows the MBE provides the traceability among requirements, models, test cases test articles, test activities, analyses, and validation reports. This ensures consistency across the tasks and enhances the engineering specialists work by relieving them of administrative accounting of ensuring traceability so they can devote their energy to the effort of creating the appropriate design, analysis, integration and verification.

Model-based system development is a systematic approach for documenting, analyzing, and validating the system and lower tier requirements against the architecture/design and concept of operations (CONOPS). In the past, the most common approach was based solely on performance modeling, which can be used to confirm the feasibility of meeting a performance requirement and to validate the allocation of lower-level derived requirements but did not have the capability to validate operational requirements and capability.

Model-based validation can also be used to validate the established functional requirements and ensure consistency between the established CONOPS, requirements, and baseline architecture/design throughout the product development lifecycle. This validation method reduces the risk that operational issues will be detected during the system/product operational test and evaluation (OT&E) phase after system delivery.

A highly effective approach to demonstrate consistency is to document the time-ordered sequence of functions performed by the system via behavior modeling (also known as activity diagrams, sequence diagrams, event diagrams, event scenarios, etc.), and associate requirements to functions that have been assigned to elements or components of the baseline design within the model. These diagrams, which are derived from the system CONOPS, show the time-ordered interaction between system elements (including the functions performed) and the information communicated; and include schematics, such as those defined in System Modeling Language (SysML), to control the activity flow. This provides traceability between the requirements, CONOPS, and the product design. Requirements that do not map to a function assigned to the design or functions that have no requirement mapped may indicate an inconsistency between the established requirements, the baseline design, and the system/product CONOPS. An additional benefit of this approach is that the developed models can serve as the basis for the scenario-based testing (e.g., mission critical events (MCE) execution, day in the life test) used to test the as-built system. When linked with the verification and validation test plans and procedures, they provide end-to-end validation throughout the product development lifecycle.

A.3. Current versus Future Practices

Table 3, below, contrasts current practices to potential future MBE based practices for requirements documentation, analysis, and validation.

Table 3. Current Practices versus MBE Practices

Process Tasks	Current Practices	MBE Practices
Develop the system specification after performing requirement analysis	• Create the system specification as a paper document using paper architecture diagrams • The requirements are captured in a Word document or requirements management tool. May not show relationship among requirements and architecture • Manually ensure traceability and synchronize among the system CONOPS, architecture diagrams, system specification, external interface documents, and all plans • Requirement risk management is a manual process of counting and tracking requirements, their change, and their validation	• Generate the preliminary system specification artifact (e.g., document) from the system model set and generate other documents, as needed • Functional requirements in textual form can be refined into functional models that more clearly convey the required behavior. • System models ensure traceability and synchronize among the system CONOPS, architecture diagrams, system specification, external interface documents, and all plans • Link requirements in the models to reference model (satisfies, refines relationships), to show relationship between requirements (derive relationship). Include technical requirements in the models as requirements elements • MBE tools and methods may enable routine requirements traceability for Class C and D missions • CONOPS and operational concepts are defined by model use cases • Requirement risk management may be supported by models in the counting and tracking of requirements, their change, and their validation can be generated to support decision making
Manage system risk	• Risk assessment workshop held and risks collected, analyzed, and finalized into a document or risk assessment tool	• Include risks in the models and link to model elements associated with the risks • Model elements may be linked to external risk management tools • The model can be interrogated to identify risks associated with each model element, or the model elements associated with each risk, to help with risk mitigation planning and execution.
Develop budget and cost estimate document	• Use paper system architecture and paper preliminary system specification as input to costing models • Use the costing models to develop budget and cost estimate • Manually ensure that the costing models are up to date with changes to system architecture and preliminary system specification	• Integrate the system model set with the costing models either through software interfaces or through file sharing • Automatically retrieve inputs from the system model set to the costing models • Updates to the system model set are automatically propagated to the costing models
Develop the validation strategy	• Use the paper system architecture and preliminary system specification as input to the validation strategy development • Manually create the validation strategy	• Update the system model set to include views for validation • Add the validation requirements to the system model set • Automatically generate the validation strategy from the system model set • Use the models to provide early and frequent validation

A.4. Transition and Implementation Recommendations

This section first defines an MBE vision for how it may benefit systems engineering, requirements analysis and validation, and ultimately enhancing mission assurance. It then discusses what it takes to transition to MBE from current practices both in the near term and then the long term. While this information applies across systems engineering it sets the context for requirement analysis and validation.

MBE Vision

1. MBE could enhance SE and MA efficiency and effectiveness by minimizing clerical tasks associated with integrating multiple document sets, providing a structured set of value-added engineering tasks that would potentially reduce risks earlier in the development, improve the confidence that the system is designed properly, minimize the need for "test" with increasing reliance on "analysis," potentially reduce needs for retest, and have greater confidence that the system will operate to stakeholder operational needs.

2. MBE would improve the engineering associated with requirements definition (model-based documentation), requirements traceability, requirements analysis, and requirements verification and system validation.

3. DSMs that represent architectures, system, trade studies, risks, design, drawings, data, dataflow and other engineering data would be in a federated repository that is shared among stakeholders across the government and contractors. This provides a "single source of truth" that is kept current and aligned.

4. The DSM would provide a "digital twin" of the actual system so that as system variants are produced the digital twins reflect their "as built," "as delivered" design and configuration.

5. The DSM made up of a defined document taxonomy, data taxonomy, and set of digital artifacts can be shared proactively for engineering developments, independent product and process team work, reviews, audits, assessments, as well as verification, validation, and acceptance. This level of connected data enables improved engineering and program management confidence to levy go/no-go decisions to proceed with work, to test/retest, and for acceptance.

6. Traditional paper based and scheduled CDRL documents can be replaced by DSM views and reports enhancing engineering efficiency, effectiveness, and enabling improved risk assessment and handling, along with improved program decision-making.

7. For higher rate production, and product line management, such as CubeSats, MBE would likely provide a significant benefit by managing requirement sets, requirements traceability, enhance requirements analysis and validation while ensuring that verified digital twins exist for each product line item.

Transitioning to the Near-Term MBE Vision

1. Implementing MBE now improves both SE and MA by creating a single source of truth and a digital twin. It provides a series of structured, repeatable approaches that provide SE and MA rigor. These steps may change/alter the definition and use of the program's MA baseline set of tasks as well as the contractor's internal process documents and their execution.

2. Various languages and approaches can be used to define beneficial models but integrating various models that have been created using different approaches may be problematic. This may cause model integration and interchange issues. Many system modeling tools support interfaces to requirement management systems. As tools evolve there will be duplication of tool functions as requirements management tools grow to add modeling and modeling tools are enhanced to handle requirements management.

3. Integration of various models across stakeholders may not be possible but it is recommended to have a single source of truth at the system level to facilitate government – contractor coordination.

4. The system-level single source of truth will be useful to generate documents that may still be required, enhance traceability from operational use cases, through requirements, design, verification and validation. This is projected to enhance SE and MA by reducing clerical tasks and enabling engineering staff to identify and solve engineering problems earlier than when using document sets.

5. MBE may enhance analysis over document centered engineering by enabling improved designs earlier and enable earlier risk-based decision-making. This may also improve the ability to perform higher fidelity cost estimates sooner than traditional approaches.

6. The MBE single source of truth models would also permit initial validation sooner in the lifecycle to improve design decision-making, and improve stakeholder satisfaction.

7. MBE may permit more verification by analysis, reducing costly tests and perhaps reduce some needs for retest.

Transitioning to the long-term MBE vision

1. The concept of a DSM has been adopted by ODASD (SE) as its term for MBE that covers the emerging taxonomy, tool interface and interoperability, modeling languages and conventions. Additionally, workflow and engineering practices require updating to allow MBE data is acceptable as a replacement for documents.

2. The MBE tools will need to find solutions to complex data security marking and intellectual property management problems; especially in a federated approach with more than one tool and type of user.

3. The government SE, program management, and contracting approach require updates to accept digital information to replace documents that are used in progress decision-making across reviews, audits, milestones, and progress payment decisions. The acquirer needs to define guidelines on what elements should be included in the model for these reviews and audits.

A.5. Risks and Challenges

Requirements Traceability

Model-based requirements traceability removes clerical errors by providing status accounting type of traceability reports but it does not remove the risks involved with engineering judgement needed to relate elements.

Risk: Requirements traceability is an engineering activity and there is human engineering judgement involved with tracing system CONOPS, architecture diagrams, system specification, system configuration elements, external interface documents, and all plans. Inherently, traceability involves engineering judgement to determine if pieces are wholly satisfied, partially satisfied, and involve one-to-many or many-to-one elements. As engineering moves away from English-based requirements statements to requirements characterized by performance and constraints, the engineering judgement induced traceability risks will be reduced but not eliminated.

Requirement Document Generation

Model based document generation is the concept that the model data as the single source of truth can generate complete documents for SE and program management (PM) reviews and audits. Eventually the documents themselves may not be necessary as SE and PM accomplishment criteria allows the use of digital data.

Challenge: SEs are used to writing and interpreting textual requirements, and there may be some transition problems with the shift in paradigm to model-based requirements. The added expressiveness the model language can provide could lead to overly solution-centric expressions, and the challenge is to avoid this temptation.

Challenge: Models to generate the functional requirements, interface, and verification sections of a traditional specification and may/may not provide the other parts of the specification document that sets the background, purpose, use, and notes. The challenge, while documents are still necessary, is to produce a coherent document with models only auto-generating portions.

Challenge: The supporting engineering and program management processes will need to change to allow non-specification document-based evidence to be used to support the decisions. The challenge is to have the SE and PM process permit the use of modeling information instead of specifications for decision-making. SE and PM data review and audit accomplishment acceptance criteria will need to be established so that model data can be used in go/no-go decisions for SE and PM reviews and audits.

Requirements Analysis

Requirements analysis ensures that the requirements are "good" requirements (unambiguous, unique, verifiable, etc.) that also include all functional, non-functional, interface and performance requirements and constraints imposed by each specialty function.

Risk: SEs are used to writing and interpreting textual requirements, and there may be some transition problems with the shift in paradigm to model-based requirements. The added expressiveness the model language can provide could lead to overly solution-centric expressions, and the challenge is to avoid this temptation.

Challenge: Models help ensure requirements are "good" requirements through requirement numbering, boundary checking, and other means but models don't capture non-functional requirements well (e.g., human-machine-interactions, security, safety, reliability, maintainability, scalability, and usability, robustness, fault tolerance). The challenge will be to characterize the non-functional requirements in a manner so that they are unambiguous and may be verified/validated.

Requirements Validation

Requirements validation ensures the proper set of system requirements by comparing them against the mission and system concept of operations. It can be viewed as driving SE and thus both the MA and MBE tasks.

Challenge: As models define missions and CONOPS into use cases, validation becomes easier since requirements analysis and requirements traceability have been performed and are part of the modeling data. If there are engineering judgement flaws creating the use cases, system requirements, or traceability, then those flaws are preserved in the modeling data and may be more difficult to detect.

Appendix B. MA Area: Design Assurance

B.1. Literature Search and Review

Definition of Design Assurance

From the MA Guide, design assurance (DA) is ". . . a formal, systematic process that augments the design effort and increases the probability of product conformance to requirements and mission success. It independently assesses the development of specifications, drawings, models, and analyses which are necessary to physically and functionally describe the intended product as well as all documentation required to procure, manufacture, test, deliver, operate and sustain the product." (Bjorndahl, Simpson, Vandergriff, & Wishner, 2007)

Further, the objectives of design assurance are detailed below:

1. Verify design-to-requirements compliance
2. Ensure design accuracy and completeness
3. Validate documentation, configuration management, and change control processes
4. Ensure producibility
5. Ensure designs are testable and tests are valid demonstrations of design intent
6. Ensure designs are supportable
7. Ensure lessons learned are captured and communicated (Bjorndahl, Simpson, Vandergriff, & Wishner, 2007)

Design Assurance Processes

Per the design assurance guide, there are 15 process areas to be considered when applying design assurance practices. These process areas are outlined below:

1. Examining the technology readiness level of the program design elements.
2. Reviewing program planning for eliminating and mitigation or technology readiness-level risks.
3. Examining resource allocation including current and proposed staffing profiles, process, design, supplier, operational employment dependencies, etc.
4. Reviewing program planning with respect to coverage of key design assurance enterprise attributes relevant to the program phase and design attribute under evaluation.
5. Examining key technical performance metrics against margin requirements.
6. Reviewing analysis products and the program incorporation of those products for managing DA risks.
7. Monitoring test results throughout the design lifecycle especially test failures (e.g., engineering models, failure review boards, etc.).
8. Reviewing any other potential design shortfalls against initial requirements allocation as the design matures.
9. Analyzing negative trends, reduced margins, schedule slips, funding shortfalls, engineering changes, audit findings, customer feedback, etc.
10. Analyzing signification issues that are active or open.
11. Reviewing lessons learned database.
12. Review best practices.
13. Reviewing results of design assurance enterprise attributes assessment.
14. Evaluating risks from quality, functional, programmatic, cost/schedule aspect.

15. Interviewing key business and functional leaders and asking what concerns them about the program (e.g., if not enough resources, what on the work breakdown structure is not getting done).

To summarize, design assurance is concerned with ensuring a technical solution through an evaluation of that technical solution in the following high-level areas: quality of the technical solution against requirements and constraints, quality of process implementations used to arrive at the design solution (process adherence), and understanding and management of risks associated with the technical solution and its implementation.

Implementation of Design Assurance in a MBE Environment.

With this understanding, it is apparent that design assurance is directly related to both the development process and product of technical solutions. Design assurance's purpose is to minimize the risk associated with those technical solutions through reviewing and analyzing processes followed to accomplish design. Engineering practices have been in a transition over the last decade from document-centric to model-centric to model-based practices. At the heart of these practices, models are the record of authority for the design.

Current/Future Graphic:

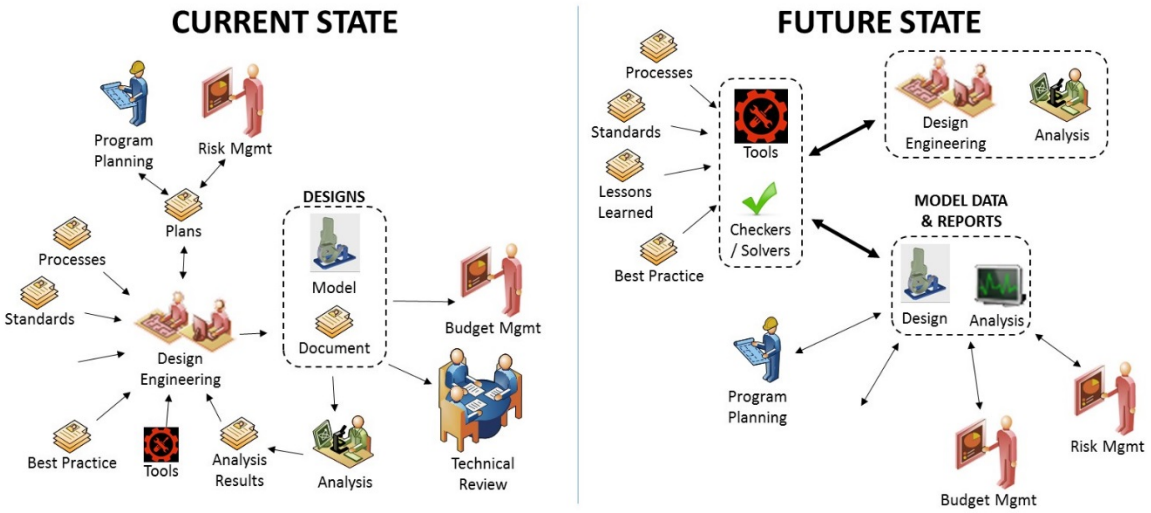

Figure 8. Design assurance current versus future.

Figure 8 (above) describes the difference between design assurance in the current state and a future state. In the current state, engineers produce both documents and models to capture system design. These engineers must follow processes and standards, and incorporate best practices to generate those designs. Other engineers analyze the designs and generate results that may be fed back in to the design process. Technical experts review system design models and documents to ensure that the right processes have been adhered to, standards have been applied, and best practices have been incorporated as appropriate. In a future state, tools will present best practices and lessons learned to designers. Tools will also be able to enforce standards and processes. Model checking features will help engineers quickly evaluate their designs using reasoners and solvers. Analysts and designers will work closely together with the same or integrated tool sets and stakeholder groups, like program management which will also be able to interact with design data with more usable interfaces and viewers, providing better data accessibility.

Summary Discussion of Key Differences

Today, interactions of organizational elements responsible for design assurance activities and program/project organizations happens mainly through interviews with, and reviews of technical and process documentation. In the current document-centric state, this interaction occurs through human-intensive efforts to read and understand how the solution and processes have been implemented to provide that evaluation. In a model-based future state, this interaction will change in a number of ways. First, interaction with design and process documentation can occur through the evaluation of model data, toolsets and environments used to capture this program effort. Not only will design assurance actors need to be familiar with design data captured in models, tools and tool environments, but they will also need to be familiar with how those tools and environments are able to enforce process performance. Additionally, design assurance actors will need to evaluate the application of tools for the design purpose, understand embedded assumptions and best practices, and how those practices may be enforced by using those tools. Also, design assurance actors will need to have new mechanisms for understanding and managing risks associated with the use of those tools and environments, similar to the way they are evaluated today, only using document-based practices. In the graphic above (Figure 8), you can see that these interactions will place a heavier dependence on the review of model data. This may require additional, or perhaps more tool-specific expertise, to effectively evaluate those implementations. Further in this section, we will explore each of the Design Assurance process areas to understand those differences at a more "localized" level, and provide guidance on what those key differences are.

B.2. Expected Benefits and Outcomes

The following are the expected benefits of migration to a future state for configuration management (CM) in an integrated MBE environment. They have been aggregated in to four key benefit areas.

Data-Integrated Design and Analysis

Automated configuration and pedigree capture – design processes employing model-based techniques will provide easier mechanisms to understand how modeled data of the solution works with analysis groups to better capture configurations and pedigrees of data used in those analyses. For example, design to analysis toolchains, because they will be more integrated in a model-based future, can capture (at the time of running a specific analysis) the pedigree and assumptions for data used. This can eliminate the need to additionally capture that configuration data, and allow for easier implementation of additional analysis runs, given design iterations.

Design, CM/DM and Workflow Tools Implement Process

1. Process enforcement – The implementation of process-integrated tools (that is tools that enforce a particular process' steps, assumptions and conventions), can provide better process adherence. Rather than leaving process adherence up to engineers to understand and tailor, tools can enforce a purer process implementation and better capture process tailoring for purposes of a particular solution. This has an added benefit of making this process adherence information more specific for design assurance review.

2. Best practice/lessons learned integration – Model-based toolsets can also provide for direct attribution of lessons learned and best practices with respect to a particular design step or concern. This allows engineers to easily understand those practices at the point of design decisions, rather than leaving that as a separate activity before engaging in design, or after reviewing designs.

3. Automated checking/solving – Model-based tools may also provide a platform for automated checking with respect to the quality of designs as they are being captured. An example of this is static code analysis as a part of software design development. Many tools for software engineering today provide capabilities for automatically checking for common errors and concerns, such as unused declared variables or mis-defined method definitions. These techniques can be applied to other engineering domains providing the same effects on improving quality quickly during the design phase.

4. Automated audit and dash boarding – Today, domains that rely on document-centric approaches require manual approaches to gathering data in metrics with respect to in-process work and design. Using model-based approaches, these metrics can be automatically aggregating, using many of the same tools employed for design to provide the data needed to understand progress and programmatic/schedule risk associated with the progress toward that work.

Architecture-Integrated Risk and Budget Management

1. Easier identification of allocation and impact – The use of model-based approaches for all elements of design, including SE, can provide an easier, less-manual platform for understanding the allocation of performance budgets to elements related to design. They can also provide an easier mechanism to understand the potential impact of design changes with respect to performance, and their impact on meeting design requirements related to those performance budgets.

2. Data-driven update and reporting as design matures – While employing model-based tools to capture design, especially when design is integrated with SE modeling tools, the ability to update performance against budgets can be automated. As design matures, these technical budget performance metrics can be automatically updated as new designs are released and performance is refined. This eliminates much of the manual process of aggregating performance data and places a smaller burden on engineers to update performance data for management and reporting.

Technical Rationale/Intent Capture for Reviews

1. Attribution of design features and decisions to applicable standards – A key component for design assurance actors' evaluation of the quality of a given technical solution is related to the ability to understand how a given design adheres to technical standards. Model-based approaches make it easy to capture relationships and associations between design features and relevant information (e.g., standards necessitating the feature, or constraining factors for the feature). Review of these relationships are more easily accomplished by querying and reporting on model data.

2. Association of design features to internal best practice – In the same way as above, references to internal best practices associated with design features can be easily reported with model data.

3. Identification of program-tailoring associated with design – In some cases, programs may decide to accept risks related to tailoring the application of a technical standard for design requirements. Design assurance actors need to understand the rationale behind these decisions and evaluate if the associated risks have been properly captured and managed. Employing model-based approaches, a design assurance actor can be easily led from a decision like this to the rationale and appropriate documentation or notation related to the acceptance of that risk.

B.3. Current versus Future Processes

Table 4, MBE and DA Process: Current and Future States

DA Process	Current	Future
Examining the technology readiness level of the program design elements.	Separate management of TRL. Manual association/allocation of TRL with architecture.	Technology maturity digitally capture and associated with architecture allowing easier reporting and management.
Reviewing program planning for eliminating and mitigation or technology readiness level risks.	Manual association/allocation of TRL with architecture.	Technology and design maturity managed together with SE and program planning tools.
Examining resources allocation including current and proposed staffing profiles, process, design, supplier, operational employment dependencies, etc.	Manual allocation/analysis of performance against margin requirements.	Automated collection of budget performance data. Process-integrated tools allow for better estimating and management.
Reviewing program planning with respect to coverage of key DA enterprise attributes relevant to the program phase and design attribute under evaluation.	Additional program activity required to understand and implement DA elements.	DA elements flow directly to program tools and processes reducing the need to plan additional program DA effort.
Examining key technical performance metrics against margin requirements.	Separation of architecture and system design with performance and system-level analysis parameters.	Automated collection of budget actual data enabling timelier, accurate budget performance data capture.
Reviewing analysis products and the program incorporation of those products for managing DA risks.	Manual review of analysis and associated assumptions with performance metrics. Slow design-to-analysis cycles.	More automated integration between design and analysis tools. Analysis and optimization driven design processes.
Monitoring test results throughout the design lifecycle especially test failures (e.g., engineering).	Separate management of test/failures and association with architecture.	Earlier and more frequent virtual verification of designs.
Reviewing any other potential design shortfalls against initial requirements allocations as the design matures.	Manual association of requirements with design performance.	Design tools integrated with requirements via architecture. Virtual presentation of requirements compliance.
Analyzing negative trends, reduced margins, schedule slips, funding shortfalls, engineering changes, audit findings, customer feedback, etc.	Manual trend analysis of budget and margin erosion throughout design process.	Designing tool integrated with workflow and data management tools allowing automated process metric collection and dashboard views.
Analyzing signification issues that are active or open.	Separate management of TBDs and active/open trades and issues manual impact analysis of open issues.	Active/open issues tracking integrated with design tools allowing automated impact analysis.

B.4. Transition and Implementation Recommendations

1. Adopt model-based design tools with standard, open APIs to allow for easier data exchange and reporting.
2. Invest in visualization and reporting technology development to maximize the value of data rich models.
3. Invest in training for design assurance actors to understand model-based design tools and environments to minimize the time required to understand design process execution, and datasets.
4. Invest in development of model-based design tools that enforce rules for process execution, and capture process tailoring and rationale.
5. Application of standard/common data definitions across all programs.

B.5. Risks and Challenges

Ability to find relevant technical data – Design assurance activities require review of technical information relevant to the given design. Traditional approaches currently have standard ways for capturing and relating this information. For example, mechanical engineers use notes and callouts on 2D drawings. When using modern mechanical modeling techniques, the information in those notes or related by those callouts are captured as model data and metadata. This information may be more difficult to find for a reviewer and could potentially introduce risk into the design. Engineers need standard techniques and approaches to provide relevant data, and design assurance personnel will need to understand how to find this data in the models they review.

Design assurance activity in a transition state – It is certain that transitioning to the described future state of MBE will take time. While transitioning, engineering data necessary to perform design assurance may be captured in different forms and in different locations. At one point, process data required for assuring process adherence may still exist in a document-based form, while information proving adherence to technical requirements in industry standards may be mapped and related in the model. Design assurance performers may be challenged to find the information they need to perform their jobs.

Integration of engineering modeling tools and environments – Some of the benefits of moving to a MBE approach require that engineering tools and the environments in which they perform become more integrated. The interfaces between tools, interoperability data transfers between tools, and the engineering environments themselves will need to be validated to ensure that these systems provide expected results in an expected fashion. Especially in early stages and during transition, tools and environments will not be seamlessly integrated and moving data will require transformation. This will require design assurance performers check the data handoffs to ensure the integrity of the design process and their products.

Appendix C. MA Area: Reliability Engineering

C.1. Literature Search and Review

The use of MBSE for reliability engineering and analysis has been demonstrated in many recent research publications. Examples include use of the object management group (OMG) SysML parametric diagrams for system reliability and availability calculations [24, 18], failure detection, isolation, and recovery design [45, 23], resiliency modeling [19], model checking [6, 1], and Network Modeling, [29]. A survey of dependability modeling and analysis of software systems specified with unified modeling language (UML) describes several different approaches to integrated reliability and dependability modeling with the OMG UML Model Driven Design processes [2]. The society of automotive engineering architecture analysis and design language (AADL) has been used for requirements management and mapping into designs [4] describing modular recoverable systems, and modeling of safety critical systems [15, 12].

Motivation and Benefits:

Management of system reliability has been mandated by Congress in the 2009 Weapons Systems Acquisition and Reform Act. However, successfully implementing the congressional mandate has been difficult because of the sporadic involvement of reliability engineers until late in the design process at which point many programmatic decisions are irreversible. As a result, operational availability and reliability requirements are not met. MBSE and associated tools enable reliability engineering involvement throughout the design process from early concept exploration to operations and maintenance.

As will be described below, MBSE allows the reliability engineering function to be performed throughout the design process from concept exploration through operation and maintenance. When the concept of operations is being defined, the operationally necessary success probabilities can be declared for each of the use cases, which in turn, can be transformed into reliability parameters that can be analyzed and measured to assess conformance. This allows for a much more precise definition of requirements that will result in a more suitable system design and verification methods (analysis and test) that are more relevant to the system end use. As the design is defined in greater detail, failure behavior can be modeled to ensure that the necessary diagnostics, detection, and recovery mechanisms are in place. With such modeling, analysis can be simplified and because the resultant system will be more reliable, maintainable, and available, lifecycle costs will be reduced. As the system changes are made, due to either new or different requirements or implementation decisions, impact analysis can be immediate thereby enabling better decision-making. Finally, a system model that incorporates analysis, detection, recovery, and maintainability provisions can be adapted and reused thereby enabling a process of continuing improvement in system acquisition.

C.2. Expected Benefits and Outcomes

MBSE enables better reliability engineering through:

1. Better requirements statements: Multiple, operationally meaningful reliability requirements stated in terms of the probability of mission success for key use cases rather than arbitrary units such as MTBF or a monolithic operational availability that are directly traceable to the operational concept.

2. Improved conceptual design: System-level reliability and availability requirements with traceability to the higher level operational concept success probability requirements to enable impact assessments, tradeoffs, and analyses of alternatives during the conceptual design phase.

3. Improved traceability: Detailed reliability and availability requirement that are derived as the functional and physical architectures are established and lower level design elements are established.

4. Automated analysis: Automated qualitative reliability analyses such as failure modes and effects analyses and quantitative reliability analyses using reliability logic diagrams (also known as block diagrams) can be generated immediately whenever design changes are made (or contemplated).

5. More complete impact analysis: Traceability from reliability requirements to verification artifacts and implementation model elements to enable immediate impact analyses as design changes are contemplated.

6. Propagation of changes: Immediate updates when changes are made to the requirements, verification artifacts, or the design requirements to enable reliability engineers to easily repeat analyses and make other changes.

7. Better documentation for maintenance: Maintenance documentation that can be automatically generated from the design based on attributes stored in model elements such as symptoms, diagnostics, and maintenance procedures.

The net results will be to reduce development costs; reduce the likelihood of unanticipated development problems due to inadequate impact analysis; increase reliability, maintainability, and availability; and reduce lifecycle costs.

C.3. Current State vs Future State

The table below reflects the current state of reliability engineering and how it would change with the implementation of MBSE.

Table 5. Current State Process vs MBSE Process in Reliability Engineering

Process	Current State	Future State
Definition of system-level requirements	Defined as part of the concept of operations – typically a single quantity	Defined through use cases with multiple probabilities of success defined on the basis of key use cases
Allocation of reliability requirements to lower-level elements	Performed from high level design documents using tools such as spreadsheets; input data for artifacts must be manually collected – not always updated with design	Use cases with success probability parameters are linked to requirements which are linked to requirements which are then linked to blocks representing subsystems that implement the requirements and artifacts that will later be used for verification
Design for reliability: failure modes and effects analysis (FMEA), fault tree analysis (FTA)	Labor intensive analyses that are generally done once based on a design representation at a fixed point in the design process. As the design changes, the design representation used for the analysis does not change	Automatically generated FMEAs and FTAs created any time as the design changes with the ability to immediately assess the impact on requirements

Process	Current State	Future State
Design for reliability: software implemented failure detection, isolation and recovery (FDIR) provisions	Software design for FDIR occurs in several layers of the architecture and in different modules. Often difficult to track because responsibility is dispersed among several different software design groups or teams	FDIR design defined in cross-cutting software models traceable to the system design. As changes are made to interfaces or configuration parameters, these are immediately propagated to other parts of the design. Assessment of the design can be facilitated through animation. Implementation of the software design can occur through automatic code generation
Quantitative reliability prediction	Labor intensive analyses that are generally done once based on a design representation at a fixed point in the design process. As the design changes, the design representation used for the analysis does not change	Automatically generated quantitative analyses based on parametric diagrams (in SysML), error annex models (in AADL) or other formalisms using data drawn from design models. As the design changes, analyses can be regenerated
Failure reporting and corrective action systems (FRACAS)	Failure experienced during test and development are reported and analyzed by a “failure review board”. Assessment of the implications and impact on requirements, design, and use cases depend on expertise on the board. FRACAS “lessons learned” that result in design changes require a laborious change process that may stymie necessary improvements	Failure experience traced to components or activities which are in turn linked to the design. Queries on any element enable relevant FRACAS reports to be linked to the FMEA and FTA as well as to impact requirements and design. Necessary information immediately extracted in a report for more expeditious adjudication by the program change process
Reliability test and verification	Reliability related requirements verification by test or analysis tracked manually or in requirements management tool	Requirements verification through test and analysis input through model elements associated with test (primarily behavior diagrams such as sequence, activity, or use case diagrams in SysML; alternative representation in other languages). Requirements verification status available immediately through predesign views.
Maintenance manuals and diagnostic documentation	Diagnostics, preventative maintenance, and other aspects of the design extracted manually be technical writers with varying degrees of completeness based on the skill of those writers and of the design team to convey the appropriate information	Maintenance documentation created largely automatically using views and reporting capabilities of modeling tools based on information input by the designers into the model

C.4. Transition and Implementation Recommendations

Reliability engineering need not drive the transition from conventional to model based system engineering but can benefit from any changes that are implemented. The following table shows the incremental changes that would potentially be made in the system engineering process and how they could be utilized by reliability engineering.

Table 6. Introduction of MBSE and Utilization by Reliability Engineering

Design Phase	MBSE Change	How Used by Reliability Engineering
Concept Development	Use of Use Cases to Describe Operational Concepts	Definition of operationally necessary success probability by using case rather than as a single overarching MTBF or availability quantity
		Use case extensions for exception handling and contingency operations to assist in the definition of subsequent requirements
Milestone A	Requirements models instead of requirements documents	Enables requirements allocation and definition of verification tests, detailed reliability and availability numerical requirements by state and operational perspective using “satisfied by” and “verified by” relations
	Top level architecture implemented as a model	Enables definition of degraded states, redundancy requirements, maintenance concepts, analysis of alternatives
	Detailed architecture implemented as a model	Automated failure modes and effects analyses, reliability models, change management, tradeoffs, lower-level requirements definition
Milestone B	Definition of physical architecture as a model	Automated failure modes and effects analyses, reliability models, change management, tradeoffs
	Hardware design and software design and coding using models	Model based analysis of software failure detection and recovery, automated failure modes and effects analysis
	Testing and verification using requirements models	Use of requirements and design verification methods identified in the requirements models (“satisfied by” and “verified by”) for creation of off-nominal test cases for integration testing
	Problem recording and tracking associated with model elements. (Also, operation and sustainment)	Link to FRACAS for model-based impact analysis as new failure modes or unanticipated effects are identified
	Transition preparation	Use of models for preparation of diagnostics and maintenance documentation, logistics support requirements, operator documentation for off-normal condition
Operation and sustainment	Maintenance and updates of model based documentation	Association of actual reliability and availability performance with model elements to enable data collection

C.5. Risks and Challenges

The following is a partial list of risks and challenges for the integration of reliability engineering and MBSE:

1. Workforce skills: To obtain the benefits of MBSE, reliability engineers must be sufficiently skilled in both MBSE and in reliability engineering. An inadequately skilled workforce in MBSE will not be able to interact with the rest of the system engineering program and as a result, critical information will be lost with a resultant failure of the reliability engineering effort.

2. MBSE methodology: The MBSE methodology must be sufficiently well defined that the reliability engineering workforce can locate data, libraries, and other modeling artifacts necessary to perform analysis and design activities. General processes such as naming conventions, model organization, and configuration management must also account for reliability engineering.

3. Accurate and timely data: Data on failures, unanticipated failure modes, recovery probabilities, and other items relevant to reliability engineering must be updated as properties in the appropriate modeling elements to enable reliability models to be updated and to ensure the accuracy of status reporting and verification processes.

Appendix D. MA Area: System Safety

D.1. Literature Search and Review

The safety process is well documented in MIL-STD 882E, the Department of Defense Standard Practice on System Safety. The following standard definitions were taken from MIL-STD 882E and are used to set the standard terminology for the rest of this section:

1. Hazard: A real or potential condition that could lead to an unplanned event or series of events (i.e., mishap) resulting in death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment.

2. Mishap: An event or series of events resulting in unintentional death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment.

3. Mitigation measure: Action required to eliminate the hazard or when a hazard cannot be eliminated, reduce the associated risk by lessening the severity of the resulting mishap or lowering the likelihood that a mishap will occur.

4. Risk: A combination of the severity of the mishap and the probability that the mishap will occur.

5. System safety: The application of engineering and management principles, criteria, and techniques to achieve acceptable risk within the constraints of operational effectiveness and suitability, time, and cost throughout all phases of the system lifecycle.

6. System safety engineering: An engineering discipline that employs specialized knowledge and skills in applying scientific and engineering principles, criteria, and techniques to identify hazards and then to eliminate the hazards or reduce the associated risks when the hazards cannot be eliminated.

7. System safety management: All plans and actions taken to identify hazards; assess and mitigate associated risks; and track, control, accept, and document risks encountered in the design, development, test, acquisition, use, and disposal of systems, subsystems, equipment, and infrastructure.

In addition, NASA STD 8719.13, Software Safety Standard, provides the following definitions related software functions.

8. Safety critical: An event, system, subsystem or process, which if lost or degraded, could result in a critical or catastrophic hazard. Critical hazard – A condition that may cause severe lost time injury or incapacitation, or major damage to flight assets, or loss of program critical assets, or loss of primary mission objectives. Catastrophic hazard – A condition that may cause loss of life or permanently disabling injury. Also, includes a condition that may cause loss of vehicle prior to completing its primary mission.

Motivation and Benefits

As MBSE is used to perform more aspects of the SE process during the design and sustainment of space systems, the safety engineers should adjust their analysis methods and tools to take advantage of the information provided within these models. MBSE allows the design to be shared between the many engineering disciplines and maintains the communication to keep all areas up to date on changes. This

allows for the analysis of changes during the iterations of the design; however, system safety analysis usually occurs late in the design process. MBSE and associated tools can allow the safety engineer to be involved in the design process early by incorporating safety requirements as they are found by the analysis. Also, the safety engineer can track hazards as the design changes and see how these changes affect system safety. The continuous monitoring of the system design for safety and risk could be a key benefit provided by MBSE.

MBSE is used to reduce manual effort during the system design and sustainment process. The safety processes would benefit from MBSE to make the process easily repeatable especially if the safety engineer's analysis was based on the same model generated by the system engineers. The safety engineer can repeat the safety analysis as changes are made and provide inputs to the design earlier. This would create more reliable and safer systems. An additional benefit that MBSE based analysis could provide is the ability to test all cases as well as looking at off-nominal states and behaviors. The analysis could also provide an analysis for graceful degradation, not just failure.

D.2. Expected Benefits and Outcomes

MBSE can be used in two course of actions (COAs) to support the system safety process. (1) Influence the design by creating constraints or requirements for the model based on the safety analysis. (2) Use the model to support traditional system safety analysis methods in evaluating the design. Both methods can be used along with the other SE processes by supporting a rapid, iterative design method that MBSE makes available to the systems engineer. This provides the benefit of quickly identifying the safety issues and exploring the trade-offs or validating alternatives. To make the analysis using MBSE useful to the safety engineer, the views or results created by the models should be like those already used to make safety and risk decisions. Creating these views requires the cooperation between the developers of the models and the safety engineers.

The first COA can be used with the preliminary hazard analysis (PHA). Based on the results from the PHA, requirements to mitigate the hazards can be generated to implement into the model. The resulting design can then be reviewed to determine if the requirements were met. The PHA can be repeated with the current design to create additional mitigation methods resulting in new requirements. This can be repeated until the risk level (probability and severity) is acceptable. To accomplish this, safety requirements, properties or attributes would need to be added to the model.

For the second COA, the model can be used to reduce the effort to perform the analysis required using tools such as the failure modes, effects, and criticality analysis (FMECA) and FTA. For the FMECA, an automated routine can induce a failure for each element in the model, one at a time, to determine the effect of the failure of the component, part or function. The automated routine would record the result and provide a table of the effects of each failure. Once created, an automated routine can be reused to evaluate designs after modification to provide an iterative design method for the system engineers. Some MBSE tools have routines to perform such analysis, for example. Magic Draw has a plug-in called 'Cameo' to generate FMECA output.

Generating a fault tree based on using the automated routine would be another implementation of COA 2. The attributes of the components would include types of failure which can be used to then build the fault tree as shown in Figure 9. The fault tree shows paths of component failures that could lead to a system failure. If any of events in the path can be prevented, then the risk of these component failures causing a system failure has been mitigated. The model is used to develop this view which is based on using 'And' and 'Or' gates to make the connections with failure events. In this example, one single point of failure is shown by as the 'Event A'. These single-point failures need to be mitigated and usually can only be mitigated with design changes. Fault trees are a top-down approach as opposed to FMECA which is a

bottom-up. The methods to create fault trees are varied and too difficult to explain in this section. Again, some MBSE tools have routines to perform this analysis or can be linked with the tools to perform fault tree analysis such as OpenFTA or MADe by PHM Technologies

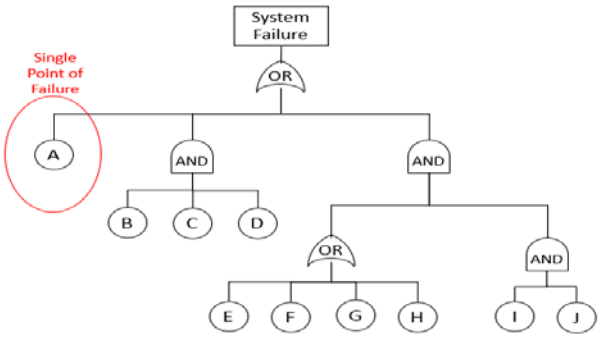

Figure 9. Example output from fault tree analysis.

D.3. Current State vs Future State

In MIL-STD 882, the safety process is divided into eight elements (or steps) that are conducted throughout the lifecycle of the system. These eight steps are shown in the table below and methods using MBSE are compared to how these steps are currently done.

Table 7. Current Safety Process vs MBSE Process

Process	Current State	Future State
Document the system safety approach	The safety process starts with building the plan on how the safety process will be performed	The same document but with added details on the MBSE model and the additional information needed to support the model and define the output views to support the safety analysis
Identify and document hazards	Hazards are identified based on past experiences and mishaps, analysis of components, and understanding of usage of the system and the environment in which it will operate.	Based on the system model, analysis tools will be used to identify hazards and document impacts of component failures. These results can be feedback into the model as attributes to components then used to evaluate the effect on the design.
Assess and document risk	This assessment determines the risk of system failure by determining the probability of occurrence and the impact of failure on the systems performance. These risks are then shown in the risk assessment matrix. To analyze the system, the system must be designed to a stage where all the functions and most of the components have be defined and built.	Using the model, the probability of occurrence and impact of the failure can be determined providing numerical answers for both. This information should be created using automated tools so changes to the model can be quickly evaluated. This allows for many iterations using the model allowing for a design that could eliminate risks.

Process	Current State	Future State
Identify and document risk mitigation measures using one of the following techniques (list from the most effective to least): - Design selection - Design alteration - Engineered safety features - Safety devices - Warning devices - Procedures/training	After the risks are identified, the mitigations of these risks are incorporated into the design, if possible, to eliminate the risk. Depending on when risk assessment, in the previous step, was performed, changes to the design might not be possible. Then safety devices, or warning systems are added.	Because the model represents the system before it is built, most of the risks identified can be eliminated with appropriate changes to the model. The best way to manage hazards and risks is to eliminate them. So, MBSE allows for early identification of risks followed by modifications to the model which are quickly analyzed to evaluate the effectiveness of the change, in order to eliminate risks.
Reduce risk	Mitigation measures are selected and implemented to achieve an acceptable risk level. For all critical or catastrophic hazards, the safety and system engineers must develop mitigation measures to reduce the probability of occurrence or the impact of the failure. For the remaining risks, cost, feasibility, and effectiveness of mitigation methods are used to determine if the risk can be reduced.	Once the risk mitigation measures are selected and incorporated into the model, the hazard analysis tools are used to reevaluate the current risks. This risk analysis and mitigation can be repeated to improve the design until an acceptable risk level is achieved.
Verify, validate and document risk reduction	The changes to the design are tested and evaluations are made. These changes could be analyzed using performance models.	Using the same tools during the hazard identification and assessment phases, the modifications to the model are evaluated to determine if the probability of occurrence and impact have been reduced.
Accept risk and document	Since, not all hazards can be eliminated, the decision is made to accept the system with the existing risks or repeat the processes to redesign the system.	The same will occur with MBSE, but the information to make the decision to accept the system should be based on results from the model.
Manage lifecycle risk	After the system is fielded, this process continues to evaluate the system as it is updated, modified, or used for a new mission or in a new environment. Also, results from operational use can validate or expose new risks.	The same will occur with MBSE, but the model will be updated and modified along with the system so the hazard identification and assessment tools can continue to be used. The modifications to the model are evaluated to determine if the probability of occurrence and impact of any hazards has changed. Also, the model will store and manage the safety and risk properties in a single source.

This table shows that the same process would be employed by the safety engineers but the MBSE models would allow them to perform a complete analysis earlier and with more repeatable results. These results could then be used to improve the safety of the system through an iterative process. This is shown in Figure 10 (below) and aided using automated tools that provide quick and repeatable analysis methods to identify and assess hazards. The best method to mitigate hazards is to modify the design to eliminate them. With MBSE, this analysis should occur early so safety can influence the design and any modifications made to the design.

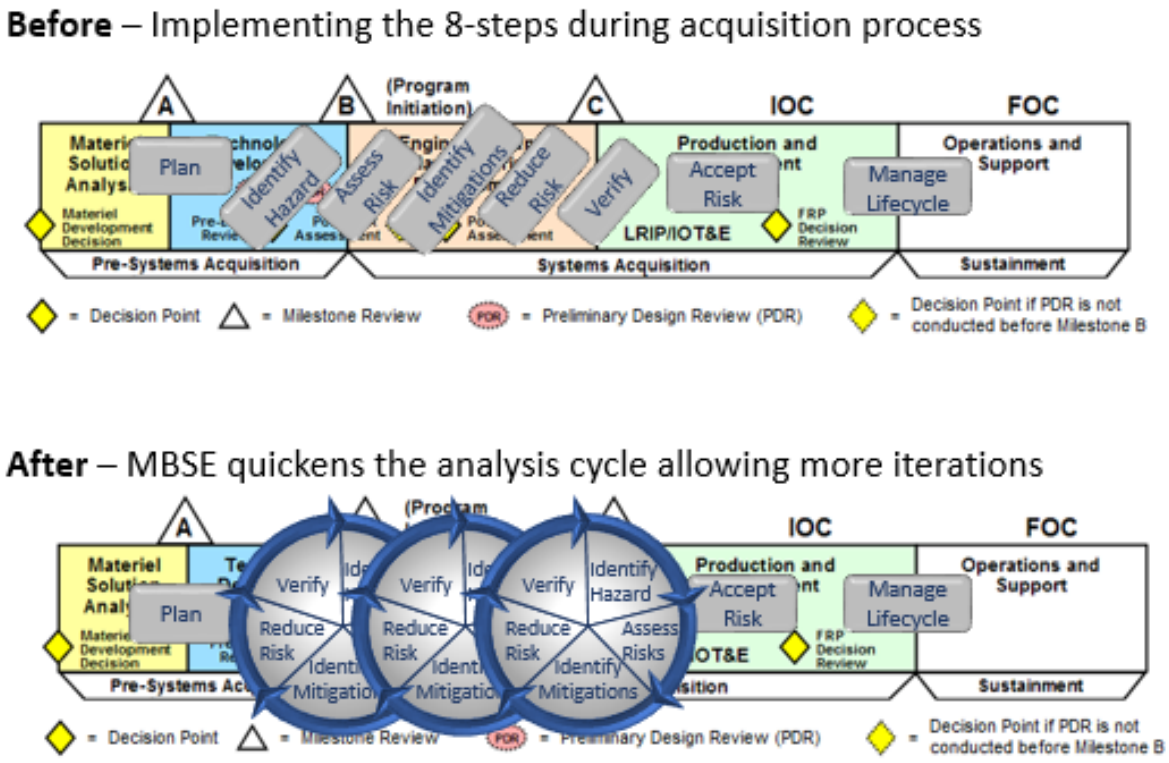

Figure 10. MBSE in the safety process.

D.4. Transition and Implementation Recommendations

The main steps in the safety process (steps 2 through 7) can benefit from the use of MBSE, but would not eliminate any of these steps. However, the safety engineer can be involved early in the design process as the model is being developed by providing requirements for the system, based on safety analysis. The safety engineer can use the following four stages to implement MBSE. First, add safety use cases and safety constraints to other use cases. Next, using these use cases, add requirements to the model or add attributes to the system components to evaluate risk. Third, build the design with the model including the safety requirements and the evaluation of risk. Finally, use the results including views, values, and queries to facilitate the safety evaluation to eliminate or reduce risk.

To incorporate safety analysis into the MBSE models, additional attributes will need to be added to the components of the model. These attributes will describe the possible failures of the component and the output produced if that failure occurs. Also, if known, include the probability of the failure occurring. All these attributes need to be developed between the safety engineer and system engineer developing the model. The precise nature of these attributes would depend on the types of analysis being performed and the tools developed to perform that analysis.

The results provided by the tools can vary based on the sophistication of the tool. These tools could provide a listing of the single points of failure while the more in-depth analysis could provide quantitative information. This can include probabilities of failures, reduced performance, and time between failures. These could support reliability or MA analysis with the ability to provide, not only to analyze failures, but to evaluate degrades states. Whatever the results provide by the tools, they should be formatted similar to the current analysis methods. This allows safety engineers, especially those not associated with the model build, to understand the results of the analysis. This aids in understanding the system and its hazards and

risks, gains trust in the analysis tools and model, and reduces the training required to gain understanding from those outside the program who are performing reviews.

Later in the lifecycle, the model and analysis tools can be used by operations to set normal operating conditions and limits. Also, they can be used to evaluate failures to determine causes and determine methods to eliminate or reduce the possibility and impact of future occurrences. Finally, as the system, environment or mission changes, the model and analysis tools can be used to reassess the safety aspects of the system. The model has kept the safety assumptions, identified hazards, mitigations methods, safety attributes and properties, and the analysis methods and views all in a single location.

D.5. Risks and Challenges

Several risks are associated with the application of MBSE for evaluating system safety. The first risk, an incomplete or incorrect analysis, can occur because the model does not include all the system faults or it is missing 'back-door', 'sneak circuit' or non-linear type effects. These types of errors can exist in any safety analysis and requires an experienced safety engineer to look for them. The second risk is relying too much on the model or tools to perform the analysis and not using an experienced safety engineer. The first risk highlights why this second risk can be an issue. The third risk is the misuse of the model, or the data from the model outside the 'valid' range of the model, or having incomplete attributes or constraints within the model. Therefore, the analysis produced for the model will be incorrect. This is risk not only for MBSE but for all engineering models.

Beyond these risks, many more challenges exist for using MBSE to perform safety evaluations. First is finding good tools to perform the analysis of the model to create FMECAs or fault trees. The next challenge is gaining the trust of the safety experts, range safety personnel, and user that the results generated from the model using the tools are useful and complete. The third challenge is the acceptance of new safety requirements by the system engineers causing a redesign of the system components based on the analysis. And finally, the time between iterations or modifications to the models might be too fast for the needed safety analysis to be performed. Even with the automated tools, the effort to perform the steps in the safety process would still take time for the experienced safety engineer.

Appendix E. MA Area: Configuration Management

E.1. Literature Search and Review

Definition of Configuration Management

MIL-HDBK-61A defines configuration management (CM) as "… a process for establishing and maintaining consistency of a product's performance, functional and physical attributes with its requirements, design and operational information throughout its life [26]." The Space and Missile Systems Center (SMC) Standard, SMC-S-002 [44], expands upon this definition by considering CM as a discipline across an item lifecycle that accomplishes the following:

1. Identify and document the functional and physical characteristics of configuration items.

2. Control changes to configuration items and their related documentation.

3. Record and report information needed to manage configuration items effectively, including the status or proposed changes and implementation status of approved changes.

4. Audit configuration items to verify conformance to specifications, drawings, interface control documents, and other contract requirements [44].

Inherent in maintaining control of and managing changes to requirements, documentation, and artifacts produced during the lifecycle is the establishing of baselines to consolidate evolving configuration states [21]. The three major types of baselines at the system level are the functional, allocated, and product baselines. These may be defined as:

1. Functional baseline: Describes the system's performance (functional, interoperability, and interface characteristics) and the verification required to demonstrate the achievement of those specified characteristics [8].

2. Allocated baseline: The approved requirements for a product, subsystem or component, describing the functional, performance, interoperability, and interface requirements, that are allocated from higher-level requirements, and the verifications required to demonstrate achievement of those requirements, as established at a specific point in time and documented in the allocated configuration documentation [8].

3. Product baseline: Describes the detailed design at a specific point in time, for production, fielding/deployment, and operations and support. The product baseline prescribes all necessary physical (form, fit, or function) characteristics and selected functional characteristics designated for production acceptance testing and production test requirements [8].

Configuration Management Processes

The objective of CM is to ensure that baselines for both hardware (HW) and software (SW) are "…consistent, accurate, and repeatable throughout the system's lifecycle and that any changes to those baselines maintain the same accuracy, consistency, and repeatability [16]." The system developer is charged with the responsibility to implement a change management program (CMP), established by a configuration management plan. This plan embodies the elements needed to ensure the consistency and accuracy of the baselines [25].

TOR-2011(8591)-21 describes key attributes of CM:

1. CM is a process for implementing CM principles and practices in the identified context and environment.

2. Change management is the practice for communicating and managing potential or actual baseline changes.

3. Communication of these changes combined with effective analytical tools and process facilitates evaluation of the impact of changes with respect to cost, schedule and performance, and allows the program team to make informed decisions [25].

Implementation of Configuration Management in a MBE (MBE) Environment

Both TOR-2011(8591)-21 and MIL-HDBK-61 define key CM processes needed to implement the major CM functions. This is accomplished via definitions in the TOR and by an activity model in MIL-HDBK-61A [26]. However, the "context and environment" used by both documents is the traditional document based approach. The goals of this document are:

1. Serve as a guideline and best-practices document for executing MBE in concert with CM.
2. Serve as the basis for an addendum to TOR-2011(8591)-21.

Key Configuration Management Processes

The following CM processes will be evaluated in terms of implementation in the current state consisting of a traditional document based environment and in a future state of an MBE environment. These processes are defined in TOR-2011(8591)-21 except for the addition of Configuration Data Management which is added from MIL-HDBK-61A.

1. Configuration management planning
2. Configuration identification
3. Change control
4. Interface control
5. Configuration status accounting
6. Configuration verification/audits
7. CM related data management. This is added from MIL-HDBK-61A.

E.2. Expected Benefits and Outcomes

The following are the expected benefits of migration to a future state for CM in an integrated MBE environment.

1. Improved synchronization of CM baseline artifacts: These consist of automated, n-way links between requirements, architecture, design, analysis, test, and verification. The future state ensures that the functional architecture of configuration items (CIs) are completely linked to functional requirements in the baseline. Performance allocations at the assembly and CI-level are verified via integrated engineering analysis using tools such as Nastran and OPNET Technologies, or to measured test results. Reallocation of performance parameters is accomplished, as needed. The future state also has the capability to track and manage the configuration of the design as the development process continues, is self-reporting, and can roll back to earlier configurations, if required.

2. Full traceability of changes and improved change control: There is a clear "chain of custody" from the allocated to the functional to the design baselines. As change orders are initiated, the impact from requirements to design can be quickly assessed. Full integration between SE, engineering analysis and the CM repository will result in a more rapid and accurate synchronization to new baselines as they are established in the change control process.

3. Greatly improved capability for identification and documentation of interfaces: Use of SysML-based tools in the modeling of interfaces versus use of a drawing tool forces the analysis and documentation of data interactions between systems and subsystems. The information contained in the model database can be output via matrix views such as the SV-3 and SV-6 that display the full description of data exchanges and on the data itself. Gaps are also identified.

4. Improvements to the functional configuration audit (FCA)/physical configuration audit (PCA) process: In the future state, the capability exists for continuously observing and checking the allocation of functional requirements to configuration items. Gaps and overlaps are readily apparent and displayed in the reporting tools. Performance requirements are also allocated to assemblies or CIs as needed and engineering analysis tools are used to verify the integrity of the allocation. Integration is also achieved with respect to test results, which become part of the MBE environment. This results in an automated FCA process.

5. Early identification of engineering change orders via integration of performance and functional models: Identified gaps between functional allocations and requirements or between performance allocation and analysis or test results can become change orders if required.

6. Good CM practices can also positively impact the design process: The future state has great capabilities to track and monitor changes to the design as initiated by the engineering team. The CM environment will provide capabilities similar to current software practices. The CM model environment will support tracking of changes down to individual model elements as well as branching for parallel development of variants, merging of changes into multiple branches and tagging of baseline configurations. Previous generations of CM tools tracked model changes at a higher level. Any changes result in the generation of new threads that must be reconciled via a configuration control board (CCB). The CM environment also has a roll back capability that can undo changes as needed. The entire model file also is configuration controlled using a tool such as Subversion or IBM ClearCase®. All of this results in documentation and traceability of changes to the design with a go-back capability.

E.3. Transition and Implementation Recommendations

CM Processes in the Document Based Environment (Current State): Problems and Lessons Learned

Configuration Management Planning

In a document-centric environment, the CMP is established as a stand-alone document in either Microsoft Word or PowerPoint. The elements of a CMP as given in the TOR [25] together with how each element may be implemented in the current environment is given below.

1. Definition of the CM system: This is a section in the stand-alone CMP that identifies how the documents that will eventually comprise the system baselines are to be managed. For example, documents may be stored in a file-based shared area repository using the capabilities of the operating system to manage access. Some tool-based environments can be established with document approach, such as use of Agile® or IBM ClearQuest®. However, it is the documents

themselves that are being configuration controlled. There is no link between the CM system and tools or engineering methodologies used to create the data contained in the document.

2. Identification and management of project data and configuration items (CIs): CI identification along with associated data is document based. An example is use of a specification tree to establish the CI-related schema of requirements.

3. Planning and control of project baselines:
   This involves the following two activities:
   a. Definition of the constitution of a project baseline by identification of artifacts (e.g., requirements, architecture presented at a design review)
   b. Control of the baseline by controlling the artifacts. Again, there is no direct link between the documents that comprise the baseline and the engineering processes used to create the information.

4. Planning of configuration audits and status accounting: These are milestone-based activities tied to the project schedule and are used to evaluate the baseline [21].

5. Build and release management: In a software-centric environment in the current state, key elements of a build and release management plan would include definition of a workspace, versioning, building (compiling), dependency management, release management or deploying software into a repository, repository management, and change management [11]. For hardware, build and release management is handled by version control of PDF drawings in a product lifecycle management (PLM) system. Software build and release is linked to source code and analysis. Hardware build and release may not have links to specialty engineering analysis such as finite element structural analysis. Therefore, there may be no indicator as to why a new hardware release was generated other than notations on the drawing or in the configuration control board (CCB) minutes. We will see that in the future state, both SW and HW build and release would be linked together in a robust PLM to the full suite of enterprise data.

6. Data backup and recovery planning: This ensures that the artifacts composing a CM controlled baseline are backed up and can be recovered in case of accidental deletion. However, the data used to generate the artifacts may not be covered by this plan. The engineering analyses used to create the data may reside in system folders that are not under active CM control. In this case, the only available backup and recovery is that associated with the computer system.

Configuration Identification

Configuration identification incrementally establishes and maintains the definitive current basis for control and status of CIs throughout the system lifecycle [26]. Per the TOR, configuration identification includes the selection of CIs, issuing of numbers to the CIs, release of CIs and associated configuration documentation, and establishment of configuration baselines for the Cis [25]. In many instances, CIs may be predetermined through the use of pre-existing designs and/or the use of commercial off-the-shelf (COTS) or Government off-the-shelf (GOTS) items. However, for new items, the use of the SE processes of requirements analysis, functional analysis and allocation, and synthesis are required to translate functional and performance requirements into a system description [26]. It is the output of the SE process that is used to identify CIs. Under the document-centric approach, there is no direct relationship between the “live” SE process and CI identification. Any corrections or adjustments to CI allocation occur during feedback from functional analysis to requirements analysis, referred to as the requirements loop, or from design synthesis to functional analysis, called the design loop. This is shown in Figure 11. SE loop and

CM, from MIL-HDBK-61A [26]. In an MBE environment, functional and design feedback can occur concurrently with requirements analysis, thereby shortening the configuration item identification cycle.

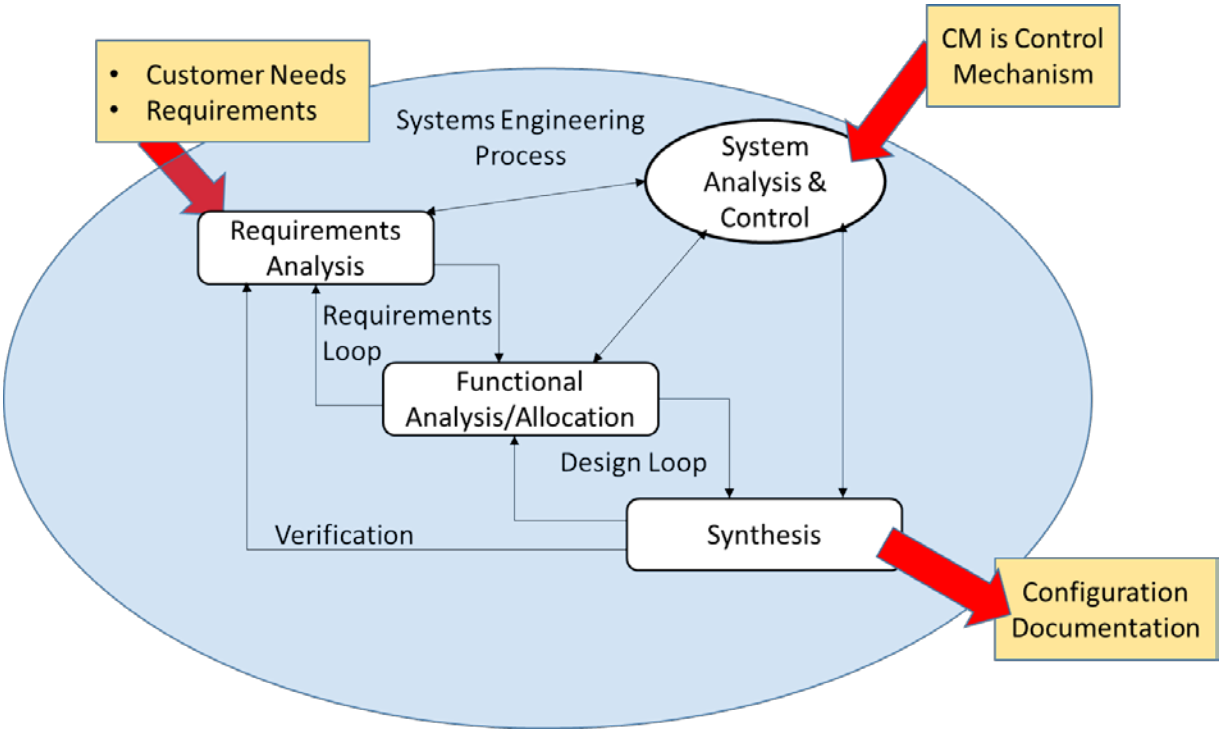

Figure 11. SE loop and CM, from MIL-HDBK-61A.

Change Control

Change control is defined in the TOR as "the systematic proposal, justification, evaluation, coordination, approval or disapproval of proposed changes, and the implementation of all approved changes, in the configuration of a CI after establishment of the configuration baseline(s) for the CI" [25]. In the current document-centric state, change control is established via a formal review board such as a CCB, which reviews and approves all controlled documentation including specifications, drawings, and manuals [16].

Interface Control

This is defined as "the process of identifying, documenting, and controlling all functional and physical characteristics relevant to the interfacing of two or more items provided by one or more organizations" [25]. In the current state, interfaces are typically controlled by the establishment of interface control documents (ICDs). An ICD may be a very detailed description of the logical, electrical, and physical connection between two systems, down to the byte exchange level. An n x n matrix of required interfaces can be generated in order to ensure that all needed exchanges are identified. In the current state, this is a painstaking process and can result in missed interface descriptions. This is a classic cause for failing a design review.

Configuration Status Accounting

The TOR includes, in this category all information needed to manage configuration items effectively and includes the following information [25].

1. A record of the approved configuration documentation identification numbers. In the current state, this can be a spreadsheet of controlled documents and allocated numbers.

2. The status of proposed changes, and deviations, to the configuration. In the current state, this would be a "status" column in the spreadsheet discussed above in item 1.

3. The implementation status of approved changes. Again, this would be an additional spreadsheet column.

4. The configuration of all units of the configuration item in the operational inventory. This would be a database of as-built items with a record of maintenance and upgrade activity, conducted at the unit and depot levels.

Configuration Verification Audits

In the current state, an FCA or PCA involves establishment of a mechanism to track discrepancies between design and document. The SMC SE Handbook defines an FCA as a formal examination of the functional characteristics of each CI to verify that it is in compliance with the requirements of the functional baseline [43]. MIL-HDBK-61A brings in compliance to performance specifications to the FCA [26]. Therefore, a key prerequisite to conducting an FCA is to first ensure that all functional and performance requirements have been allocated to architectural elements. This is synonymous with the approval of the allocated baseline, and normally occurs at the preliminary design review (PDR). In the current state, the correlation of requirements to architecture is document based and prone to error.

Configuration Management Related Data Management

This is an area that is not included in the TOR but is included in MIL-HDBK-64A. It pertains to the CM of working, released, submitted, approved and archived data [26]. In the current state, this process involves data and document identification, version control, and the relationship between data and product configuration. Missing are requirements for documenting the relationship between data items.

CM Processes in the MBE Environment (Future State)

Configuration Management Planning

In the future environment, CM will be an organic capability of the MBE suite of tools. Furthermore, rather than using a file-based system that disregards the underlying semantic structure of model-based objects, a model-based system would manage the objects within a model, versioning and saving them persistently [35]. It would also provide logical connections to web-based documents, thereby insuring that deliverable products such as CDRL documents represent the latest system configuration.

Figure 12 compares the CM environment from the current to the future state. It is adapted from the figures on charts 7 and 8 in Nguyen's conference presentation.

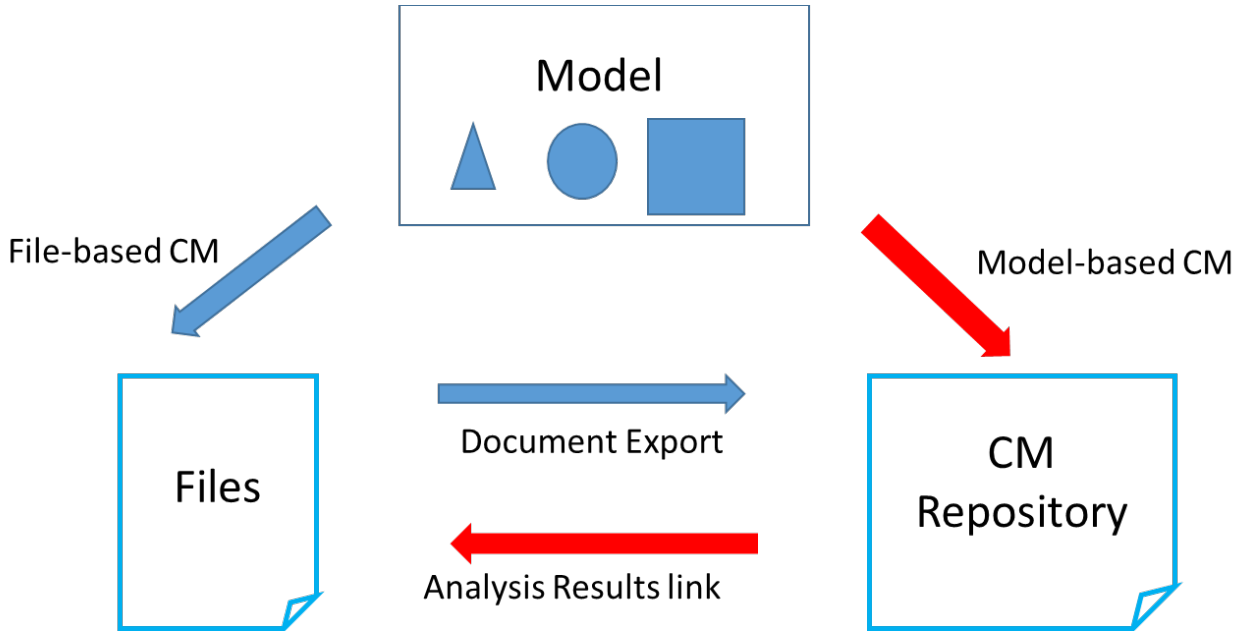

Figure 12. File vs model-based CM.

Definition of the CM system. - The CM system would be an organic structure of the tool and would manage down to the object level. For example, to edit a MagicDraw project under configuration control, the user can lock individual model elements thereby preventing others from editing them. New project branches can be created by users; these can be reconciled in a CCB and then baselined into the model [36]. Modeling tools also can compare two versions of a model and generate differences. As opposed to a simple diagram comparison, which would get cluttered by a display of simple element position changes, tracking changes to structural components provides a record of actual architectural modifications from version to version. The overall system model can be stored in a configuration controlled repository such as Subversion or ClearCase®, with the additional capability for role-based model access. Finally, an integrated collaboration environment can be created incorporating the architecture and design model with requirements, code, test, project management and product line management.

An example of an integrated CM environment for software collaboration is shown below in Figure 13. This system, developed by Boeing Defense, Space and Security uses Rhapsody Design Manager and the IBM Jazz™ solution for collaborative lifecycle management [41]. Note that CM has been substituted for project management in the original Boeing concept, the discussion on product line management is deferred, and a linkage is made between codec and test. The figure is adapted from chart 16 of Schulte's presentation.

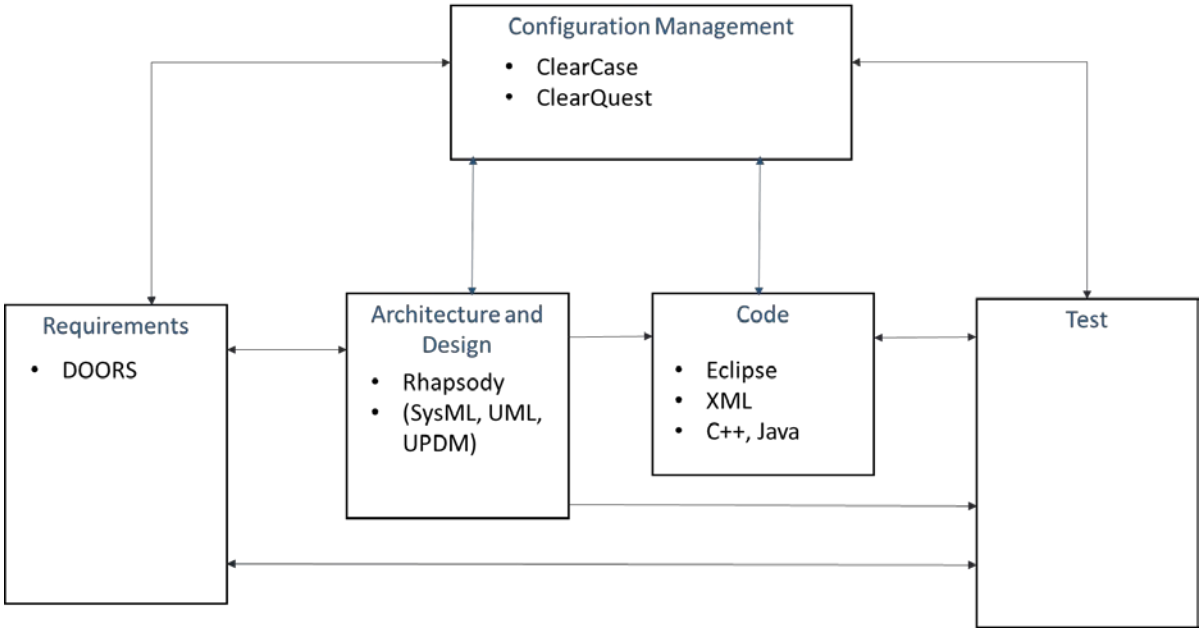

Figure 13. Integrated CM environment.

In a broader context to include both hardware and software, the integrated environment would be defined in reference to the selection of appropriate PLM tools that take advantage of repository-to-repository connectivity. The PLM would manage dependency between tools such as drawing changes linked to finite element structural analysis that shows the reason for hardware CM changes. The challenge is that a truly integrated approach may not be possible in a heterogeneous environment, necessitating the development of a federated approach between multiple PLM repositories. A product lifecycle manager of managers (MoM) (shown in Figure 14) would then have to connect and search enterprise repositories and tools to manage enterprise data [13]. It would also provide for version and configuration management of tools in the MBSE suite. To overcome the challenge of assuring connectivity among tools, models and repositories, links to engineering analysis results will be used. An example of this is the open services for lifecycle collaboration (OSLC) [37].

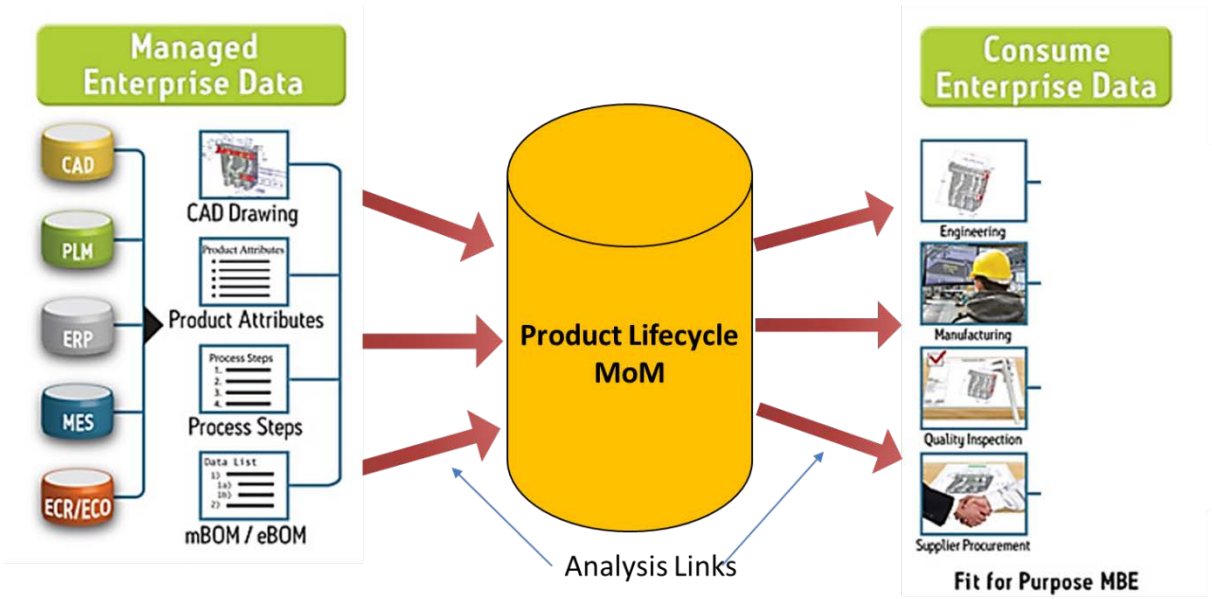

Figure 14. Product lifecycle manager of managers (MoM).

1. Identification and management of project data and Cis: As CIs are tightly integrated into the model, the requirements schema discussed above in "Configuration Management Planning", sub-paragraph 2, becomes the principle method for the organization of the architecture. Project data is also represented in the architectural model with links to architectural elements, allocated to CIs, showing the production, consumption and exchange of data.

2. Planning and control of project baselines: In the future state, all project information is contained in the model-based repository. As shown in Figure 11, the file based reports to be used in a design review or presented as a CDRL are merely a snapshot in time of what is in the CM repository. The project baseline is therefore not the reports, but rather the actual CM-controlled repository itself, containing the requirements, architecture, design information, test results, and associated data. Control of the baseline is in the hands of the CCB, which adjudicates architectural streams developed during the design process and changes to requirements as they arise. The final CM plan may be a document, but it would represent how the CM repository for the MBE project and associated tools actually function. The document-based plan would note exceptions to the environment for data that cannot be represented in the MBE framework and any needed "work-arounds" to make the MBE approach compliant with customer direction.

3. Planning of configuration audits and status accounting: As in the current state, these configuration audits and status accounts are milestone based activities. However, changes in the MBE environment resulting from analysis or test at or between project milestones are captured by the modeling tool as impacts to requirements or design. This results in parallel stream, consisting of simultaneous design, development, review and audit. The improved and integrated CM control in the future state facilitates near instantaneous check pointing, allowing for rollback to almost any point if a review finds an inconsistency or error.

4. Build and release management: In the future state, the CM tool is integrated with the MBE environment and would be coordinated using the product line MoM approach described above in "Configuration Management Planning". Software builds are managed along with the full project. Hardware releases are coordinated with drawing changes and analysis.

5. Data backup and recovery planning: In the future state, all MBE related data is contained in the environment. The environment is represented in a configuration controlled repository and can be managed by a tool such a Subversion or ClearCase®. This provides a go-back capability for both architectural/design models and engineering analysis. Finally, the computer system environment can have a backup and recovery strategy, serving as an extra layer of protection for the MBE data.

Configuration Identification

In the future state, the SE loop as seen in Figure 11 is modified by replacing synthesis with "analysis", consisting of output from the engineering performance or other non-functional analysis tools, such as Nastran or OPNET Technologies (see Figure 15). The impact of allocation of non-functional requirements to CIs can be seen directly in the MBE environment via output from the analysis tools. The feedback from analysis can be used to modify the allocation of performance parameters almost immediately, and also be used to change CIs should it become evident that the current CI allocation cannot support the required system performance. In the current state, performance gaps between the design and system performance may not be seen until PDR. In the future state, with analysis performed in the integrated MBE environment at any time, the gaps may be identified at an earlier stage with resulting savings in cost and schedule.

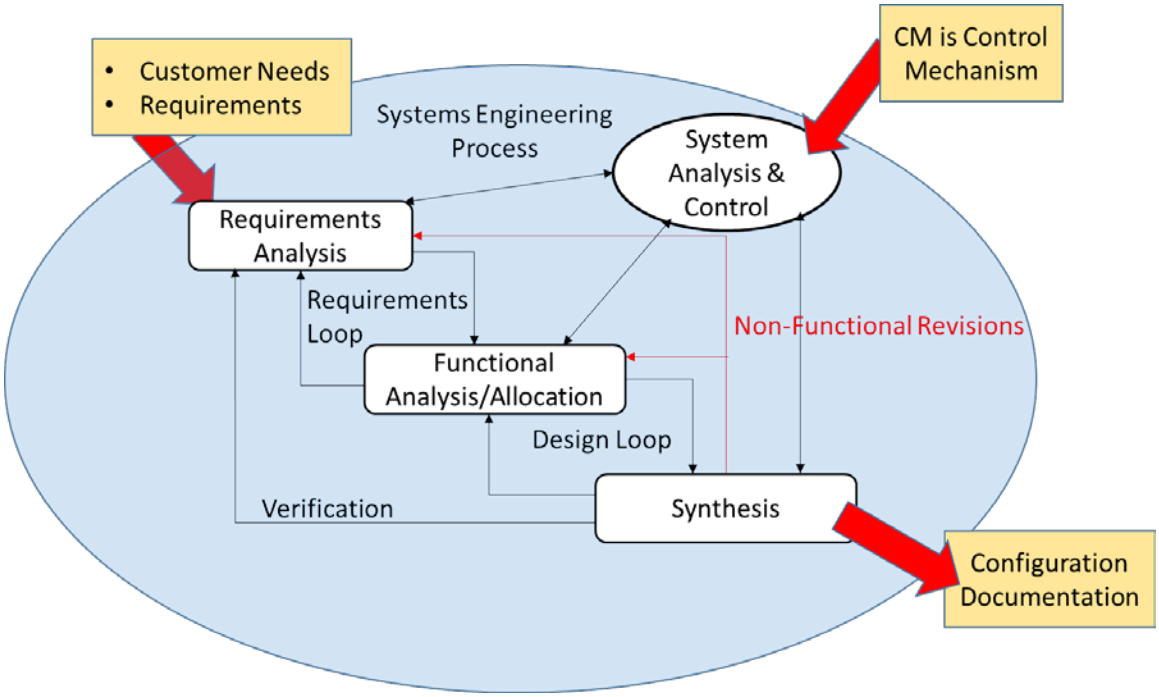

Figure 15. Impact of MBE and CM to SE loop.

In addition, MBE enabled CM can bring in additional engineering aspects. The analysis segment of Figure 15 can also include safety and security aspects, which are also considered non-functional in SE terminology. In the current state, safety and security may not be analyzed until relatively late in the design process. In the future state, MBE accomplishes this much sooner and can be used to provide an early definition of safety and security requirements, design, features, and baselines.

Change Control

In the future state, MBE provides enhanced linkage and traceability between requirements, architecture, design, and analysis. The need and effect of changes therefore is seen automatically using graphical tools and matrix views generated in the MBE environment. These tools and views can be used to generate a "CM dashboard", that indicates specific areas where potential changes are occurring. This enhances the accuracy and quality of the change control process over the current document-based state. Early identification of the need for change is also enhanced by the reviews needed to justify and reconcile parallel development streams in the model. If a review justifies the existence of new architectural artifacts that are required to achieve a desired effect but have no requirements basis, the need for an engineering change order (ECO) would be indicated. As these reviews occur more frequently in the MBE environment, change control actions can be addressed sooner than in the current state.

Interface Control

At the heart of the future state, is the functional model of the system using a modeling language such as SysML. Systems and subsystems are defined in a SysML block definition diagram. The detailed structure of the systems and interfaces are depicted in an internal block diagram (IBD). At first, all that is depicted is the item flow from one system to another in the IBD. As the model matures, the item flow is matched to a data element represented in the system logical data model. The data element in the logical model is represented by a SysML "block" or unified modeling language "class." The data format can be described using attributes of the block or class. The system-to-system exchanges can be output from the model into a matrix format, similar to the n x n matrix as previously discussed. This matrix may be referred to as an SV-3, "systems-systems" matrix in DODAF terms. The detailed description of each interaction is described in the SV-6 "system resource flow matrix". The SV-6 includes other information about the data exchange, such as the systems and functions that produce and consume the data being exchanged. The SV-3 and SV-6 can be automatically generated by information in the model, and these matrices form the basis for controlling interfaces.

In the current state, one must look through a variety of documents to find information on interfaces. These include ICDs, interface requirements documents, and interface specifications. These documents are subject to obsolescence and must be manually integrated. The future state provides the opportunity to have all required interface information in the model which can be updated at any time. It ensures that the interfacing elements are using the same interface exchange data, eliminating errors due to mismatch in the interface data type or size. This facilitates having a single state of interface truth with up-to-date and coherent information readily available. The future of MBE also holds the possibility of automated ICD generation from the model artifacts.

Configuration Status Accounting

In the future state, the product lifecycle MoM could have the capability to show the status of items under configuration control. Also included are capabilities for build and release management as well as the status of database items shown in Figure 14."

Configuration Verification Audits

The goal of MBE is developing an integrated environment with a clear "chain of custody" between requirements, architecture, design, analysis, test results, and verification. The integrated suite of tools that makes this possible has reporting capabilities to discover, analyze and correct gaps between requirements and architectural elements, design analysis and allocation of performance requirements, and test results with the verification cross reference matrix (VCRM). The VCRM can be generated automatically from

the model based on the requirements and their links to test elements in the model. This process can be done at any development phase and corrects deficiencies in the document-based approach. As configuration verification is done almost continuously, a formal functional configuration audit can be accomplished rapidly.

Configuration Management Related Data Management

The future state brings in data modeling to include a logical data model (LDM) and a physical exchange specification (PES). The LDM illustrates how data is transformed to or is composed of other data. The PES shows how web services and schemas can be constructed to exchange data and is built upon characteristics of the LDM. This closes the gaps identified in the current state.

Steps for Transition and Implementation Recommendations

Step 1: Migration to MBSE in a CM Controlled Environment:

The most immediate benefit to programs can be achieved by a rapid implementation of MBSE in the context of use of a CM tool. Use of MBSE brings in the linkage between requirements and architecture and achieves the future state for the functional baseline. Control of CIs and allocation of functional requirements results in the allocated baseline. Use of a CM tool achieves design traceability, thread reconciliation, and roll-back.

Step 2: Link to Engineering Analysis Results:

This achieves the future state of replacing synthesis in the SE loop with tangible analysis. Synthesis involves examination of engineering analysis and making judgements as to how to best modify architecture or requirements (if needed) to achieve performance goals. Analysis directly ties performance numbers to performance requirements, facilitating architecture adjustment, optimization, and reallocation.

Step 3: Link Test Results to Requirements via the VCRM:

This provides for real time analysis of test in the MBE environment. It achieves the future state of showing a requirement-by-requirement compliance to test results in the VCRM.

Step 4: Apply Advanced CM Capability to Achieve an Integrated Configuration Management Database:

This finalizes the future state by developing a CM dashboard that can display the CM status of assemblies and CIs together with a record of change.

E.4. Current versus Future State

Table 8. MBE and CM Process: Current and Future States

CM Process	Current	Future
Configuration Management Planning. Establish the CM approach for a program	• Develop CMP • Accomplish via Word//PowerPoint CM plan • Has non-active link between CM plan and implementation tools	• Uses MBE tools that organically follow and enforce CM procedures • Notes exceptions and "work-arounds" • Uses tools that help turn changes into ECOs
Configuration Identification. Establish and maintain a definitive basis for control and status of a CI throughout the lifecycle	• Identified based on: – Performance parameters – Physical characteristics – Risk, safety, logistics, maintenance. • Maintained by documentation schema and release repository	• Identified by functional characteristics – "Actors" or "Performers" • May apply traditional ID • Enforces tool-based CM baseline enforcement with roll back
Change Control. Apply processes for systemic proposal, justification, evaluation, coordination, and approval/disapproval. Establish new CI baseline	• Uses formal review boards (CCB) • Has a CM release plan • Establishes specification and drawing manuals	• Employs rigorous change management (e.g., Dynamic Object Oriented Requirement System (DOORS) Proposed Changes) • Traceability of requirements to architectural elements • Shows impact of RFCs/RFVs graphically with traceability
Interface Control. Identify, document, control physical and functional characteristics regarding interfaces	• Establish document-based ICDs • Uses matrix-based (nxn) ID of needed interfaces (missed interfaces a key reason for failed design reviews)	• Employs modeling process that identifies need for interfaces • Model language (e.g., SysML) facilitates specification of interfaces for information flow • Employs model-based ICD generation
Configuration Status Accounting. Records and reports of information needed to manage CIs	• Uses computer-c-based or manual list of baseline docs, CIs, baselines, configuration status, change status	• Linkage of data and documentation • MBE-based CM dashboard
Configuration Verification/ Audits. Includes FCA/PCA.	• Establishes audit mechanism to track discrepancies between design and documentation	• Track requirements to architecture and design with automated reports
CM Related Data Management (from MIL-HDBK-64). Includes CM of working, released, submitted, and approved data	• Uses file-based processes for: – Data and document ID – Version control – Relationship to product configuration	• Uses Model Based processes: – Logical Data Model – Versioning and baselining – PES auto-generation

E.5. Risks and Challenges

Risks

1. A robust configuration management approach is an essential element of MA. During the transition from the current to future states, there is the risk that current tool-centric CM approaches may be dropped prematurely, resulting in the potential loss of CM capabilities.

2. There may be cost and schedule risk in adopting the future state for configuration management. A program may decide that it cannot afford the cost or the potential schedule delays resulting in migration away from a tool-centric legacy CM approach.

3. CM is largely done through the implementation of various PLM tools/software suites. There is a lack of tool interchangeability in the MBE environment which can lead to errors and difficulties in tracking the CM of various parts, subassemblies, and even systems in the model. It is not desirable to impose tools but practitioners will need to work with tool providers to assure model interoperability.

4. It is essential for engineers across different domains on a program to have a clear pipeline for communication. Any given program has multiple actors with multiple approaches to solutions. In a model-based world, engineers will look to pull the latest revisions from the model and apply it to the work or test what they are doing. Total faith in a model can lead to a lack of communication among team members and the added risk that the wrong revision was used for analysis or basis of a test. It is important to insure that the modeler and decision-makers are involved in the change process. Working revisions are also key to differentiate and note in a model. CM in the model-based world will have to change helping mitigate these risks. The goal is to have informed team decision-making.

5. Each tool has versions that need to be tracked. This can be challenging when the MBE environment is comprised of many tool sets.

6. CM model access is often access that is only granted to a select few people to update or change the model. This leads to the challenge that not everyone on the engineering team/program has access to model. Only specialized personnel control and update the model. MBE is less likely to be adopted if people are excluded from the modeling processing. There must be a balance between keeping the model accurate, keeping CM up to date,

7. and allowing the team to have hands-on access with the model. Also, over-dependence on modeling should be avoided.

8. The modeling tools may not be sophisticated enough to compare the changes to a model or comparing two versions of a model and finding the meaningful differences. They currently generate a huge list of all changes including the ones to the diagrams, some of which may be negligible but detract from the review of the significant changes.

Challenges:

1. CM practices should be implemented in the MBE environment without becoming overly dependent on the tools. Effective interactivity makes models useful at the rate that humans can make decisions, enhanced by properly dividing tasks between humans and computers.

2. Unless a concerted effort is made to develop a fully integrated suite of MBE tools with the capability for configuration management of both tools and data, a federated approach will have to be developed. This is contingent upon the capability to integrate data but alleviates the problem of integrating tools. The challenge is to adopt versions of tools that support use of a data integration and transfer methodology including links to engineering analysis results.

3. Integrated CM in the future is also dependent upon the migration to a fully electronic development environment. The challenge is that it may not be possible to fully remove paper or other hardcopy data elements from the CM environment. A catalog of hardcopy data may have to be developed and integrated into the CM environment to show the contents and location of essential products.

4. An integrated CM environment will still have to contend with different information assurance regimes for product lifecycle data.

Lessons Learned in Transition to MBE

1. A clear CM strategy is essential to avoid baseline confusion: For example, requirements may exist in a text based system such as IBM Rational DOORS®, as artifacts in an MBSE model, or as a document in a PLM system. In the future state, a PLM MoM would automatically synchronize requirements in all three environments. During the transition, the requirements baseline must be defined as residing in a specific system; requirements in other systems would have to be manually synchronized.

2. Models need to be under CM control: This needs to be done (even for a few number of users) and would avoid model corruption, uncertainty, and data loss.

3. Plan PLM tool selection from an enterprise perspective: A piece-wise approach can result in duplication and excess costs.

Recommendations for Easing Transition to the MBE Future State

1. A transition plan from the current to future state should be developed at an enterprise level. This would include a schedule that individual programs could use for planning purposes, and would mitigate the risk of loss of CM capabilities.

2. Funding for key elements of the future state should also be provided at the enterprise level. This would include new tools and training. Combined with a migration plan, this would mitigate the cost risk to programs in adopting new approaches to CM in the MBE environment.

3. User-friendly tool interfaces, enhanced navigation capabilities, and model traceability are essential when selecting PLM software. Having the capability to link and accurately configuration manage various aspects of a design from part procurement through detailed analysis is key from a MA CM perspective.

4. Assigning “view only/read only” access to certain elements of the model helps ensure accurate configuration management throughout the development and creation of the model.

5. Ensure that CM is not over-applied in the MBE environment. MBE is a transformational change in the way we do engineering but we need to also balance this with the real time demand of product development. CM is critical in all aspects and elements of MBE from tool to model to

component CM but practitioners need not implement unnecessary gates in the process that would ultimately inhibit or slow the team's progress. This will ensure improved efficiency and effectiveness while still building a reliable product.

6. An information assurance API plan needs to be developed for CM and should also be done at the enterprise level. This would be part of an overall strategy on how to manage configuration across information boundaries.

Needed CM Skills in the Future State

CM personnel must be flexible in adopting new approaches and tools in the future state. Many long-standing document-based methodologies and management systems may be rendered obsolete as the focus of CM shifts to a data-centricity. It is not recommended that CM personnel become skilled on all the MBE tools and capabilities that will be available in the future state. However, a summary awareness of the issues involved in relating specialty engineering analysis to configuration items will be needed.

A detailed understanding of the interchangeability amongst tool sets would be required for a MA engineer to audit and find discrepancies in analysis, design, test, and procurement. This is done now by sifting through paper documents. The more familiar a MA engineer is with the layout of the programs model, tool suites, and CM controls, and how they all play a role with each other, the more efficient and accurate and detailed he/she can be in their job of independent MA oversite.

Appendix F. MA Area: Independent Reviews

F.1. Literature Search and Review

Motivation for Reviews

The Government Performance and Results Act (GPRA) of 1993 requires federal agencies focus on results and customer satisfaction, to address gaps in program oversight and "insufficient attention to program performance" (Government Performance Results Act of 1993). A practical and long-held method for ensuring quality outcomes and surfacing problems and errors early for research, product, and technical development programs is peer review [22, 40]. These reviews become even more critical in the space systems domain since the cost of failure can be catastrophic, including the loss of billion-dollar payloads [39] and even crew in the case of human spaceflight [7]. Even the best and smartest people are still human and can miss things; having others review our work gives us the necessary confidence to consistently execute the special feats required of demanding space missions in a flawless fashion [30]:

> Having a fresh set of eyes look at our work can help us see what our own blinders and mental filters may hide. This is the essence of the independent lifecycle reviews [30].

Useful peer reviews are implicitly independent; however, the level of independence varies. A reviewer may be independent of the project but part of the same organization, or come from outside the organization as a consultant. It follows that more independence is better than less, since independence implies freedom from project influence which should result in a balanced assessment. We expect reviewers to be experts in the review area (e.g., thermal, propulsion, power, etc.) and thus provide well-reasoned expert recommendations that are also unbiased [34].

Formal vs. informal reviews can either be informal where project members work directly with reviewers to walk through design and analysis material, or a formal setting where briefings are prepared and presented by the project team to reviewers and stakeholders together [40]. Contractual, gated, or government-driven reviews (see Table 9) tend to be formal in nature and independent assessments tend to be more informal. The difference between the two is largely held on the reviewed-project side where more time is required to prepare for formal reviews than informal.

Table 9. Review Categories [27]

Review Type	Definition
Contractual program (gated) reviews	Major reviews held over the lifecycle of a program. Examples: SRR, PDR, CDR, TRR, LRR, and selloff to customer.
Independent assessment	A review separate from contractual program reviews used to address specific technical issues.
Government driven review	Occur at the specific request of the government or designated third party organization, held in addition to the other types of reviews.

Whatever form reviews take (we will use the term "review" hereafter to refer to all of these types for simplicity), in sum they should completely cover the evolving technical baseline in a results-oriented [14] way in order to surface and resolve possible problems and risks early; however, due to the manual nature of reviews, limits on schedule, expert availability, and cost makes complete coverage an unlikely outcome (see Figure 16). Therefore reviewers must focus on top risk areas based on past lessons learned. Tom Freitag [14] describes a set of 10 main items that "pose the greatest risk to space launch missions" established by The Aerospace Corporation, including test-like-you-fly exceptions, critical qualification

margins, first-flight items, single-point failures, nonconformance, anomalies, escapes, unverified failures, out-of-sequence work, and out-of-family results. This risk-oriented approach also results in overlap (red highlighted areas in Figure 16) in reviews with one another, which can help ensure proper coverage of important risks but can also increase review cost when overlaps occur in relatively unimportant areas.

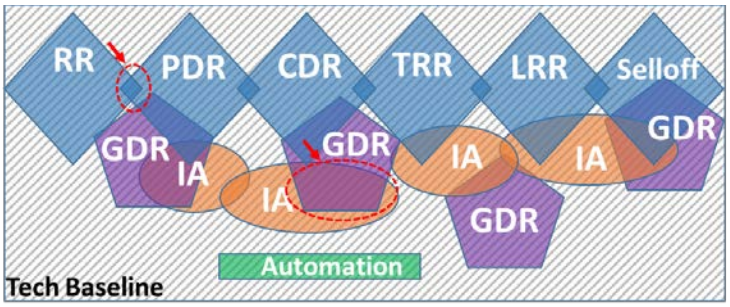

Figure 16. Review coverage of technical baseline.

Benefits of Reviews

Reviewing projects throughout the lifecycle is an important tool of program managers and customers alike, to keep both informed on key risks and mitigations but also to drive collaboration within the development team and between the team and external stakeholders.

Moreover, reviews are more valuable when independent SMEs, external and internal stakeholders from across the lifecycle are involved and can give feedback on tradeoffs, weaknesses in the technical baseline or unacceptable risks, early enough in the development process to positively impact the final delivered solution [31]. Reviews also serve to drive closure of trades, liens, and action items as well as the completion of deliverable work products, and so to help enforce a predictable development cadence for a project.

Successful reviews give stakeholders enough confidence in the project's direction and maturity to allow the project to continue to the next phase of development with acceptable risk. If reviewers and stakeholders judge the project to be unable to proceed with acceptable risk, in one or more key ways, then the work can be stopped and course corrections can be implemented [31].

Since preparing for a formal review requires the program team to pull together and summarize the current technical baseline for presentation at the review, the process of preparing for the review itself provides value since it forces each discipline to synthesize clear summaries of their progress along with supporting evidence and rationale for key design decisions. This content is useful as a historical record capturing the maturity of a design at the time of review [31].

Derivative projects may utilize the review material as a reference design and as an exemplar of successful review content at a given milestone.

F.2. Expected Benefits and Outcomes

Adopters of MBE methods can expect benefits including: reduced cost/schedule, reduced risk, and increased agility of the review cycle.

Reduce Review Cost and Schedule

1. Auto-generation of charts for reviews: MBE allows us to auto-generate review charts directly from the model onto known and approved templates. Further, by using review templates we can ensure the pertinent design details are reviewed consistently and adequately. In addition, auto-generated charts will reduce the number of chart inconsistencies across review materials.
2. Reduced review preparation time: By shortening the time spent preparing for reviews we allow high-value program activities, including critical design processes, to continue with minimal interruption. Moreover, preparation costs are reduced and team members spend more of their time maturing the design.

Reduce Risk

1. Concurrent engineering: By using an integrated model cross-discipline design teams collaborate increasing productivity and allowing errors to be detected early.
2. Higher variety of stakeholder involvement: Generate stakeholder-specific views from the model. Since stakeholder and SMEs across disciplines are continually involved as the design matures, design escapes, manufacturing and assembly shortcomings can be considered early which reduces late-cycle iterations and reduces the risk of schedule overruns.
3. Automated design check: Potential disconnects and design escapes are flagged in the model. Automated design checkers also verify that best practices and lessons learned are implemented in an efficient and consistent manner.

Increase Agility of Review Cycle

1. Explicit entrance criteria: Review entrance criteria determined by completion of explicit entrance criteria in model. Metrics in model will track results from design activities and determine when a review should be held.
2. Focused review groups: Tailored reviews are held where SMEs are able to walk through the baseline and assess maturity and risk through views tailored to their background.
3. Continual delivery of incremental information: Model is accessed by the customer at agreed-to entry-points (i.e., prior to a major milestone).

F.3. Current versus Future Processes

Current State

In the current state (see Figure 17), a review is needed to assess the current state of the design, whether formal or informal, as discussed in Motivation for Reviews and Benefits of Reviews sections above. The entrance criteria for an independent review is subject to interpretation by the project. When a formal review approaches, design development is frozen and the design team focuses their attention on preparing for the review. Since the design team is devoting most of their time to review preparation and the design is frozen for the review, little progress is made on the design during this time. During review preparation designers prepare slides, gather work products, and often find problems during the collaborative preparation process. Though Figure 17 focuses on formal reviews, the same pattern occurs for informal reviews but with less emphasis on preparation versus engaging with peer reviewers, gathering requested information, and answering questions. Once the review is held, either formally or informally, action items are generated which the team must now consider, disposition, and incorporate in the design when appropriate. Throughout this process the designer is expected to be completely transparent and open about the state of the design and reviewers can spot-check true maturity by review of detailed design work products (e.g., schematics, analysis results, trade studies).

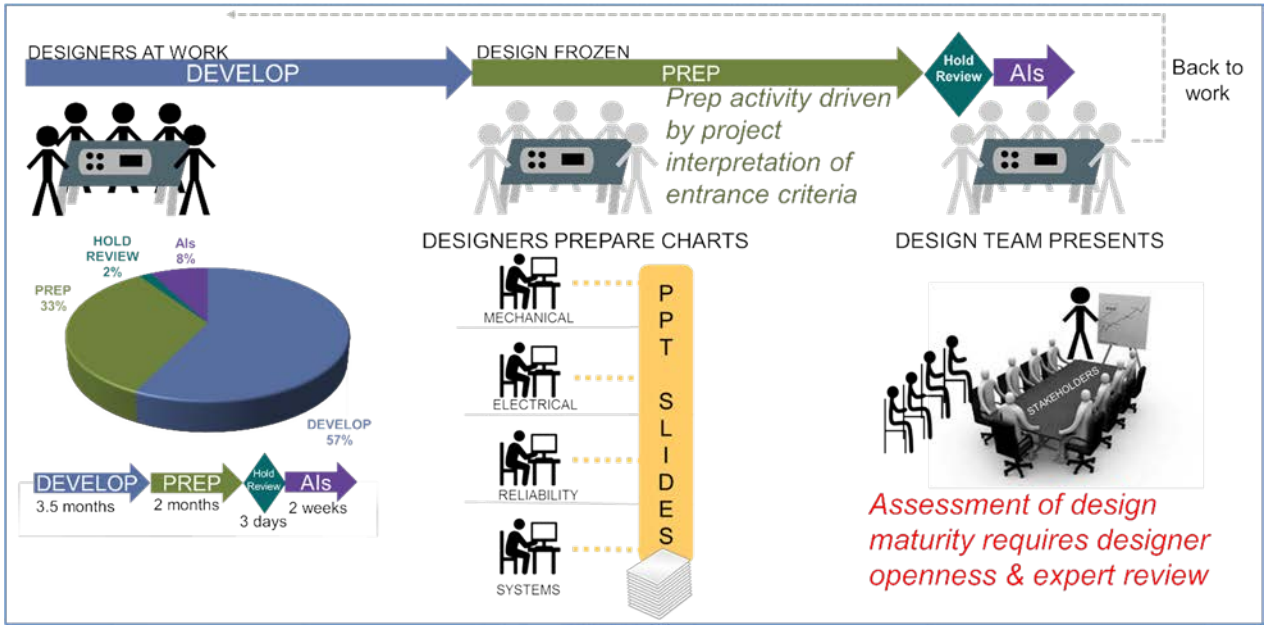

Figure 17. Current independent review process.

Though independent reviews can be a valuable tool, their demanding preparation requirements, often ambiguous entrance criteria, and uncertain stakeholder expectations from the review itself can distract the project team from maturing the technical baseline. Work products may be inconsistent since they were generated by independent disciplines without a clear collaboration mechanism. The reviewers expect the design team to be completely transparent which may not always be the case, especially for troubled programs. The design is often frozen during review preparation, and though reviews tend to drive collaboration, this process occurs out of phase with the design itself, leading to rework and swirl. Above all, reviews depend on the expertise of reviewers to apply the appropriate standards, lessons-learned, and also challenge the design team in areas of perceived risk.

Inconsistency of Work Products and Review Material

Further, design work products and review presentations are often inconsistent due to the manual preparation of each independently. Stakeholders expect consistency across work products and errors in this area can unfairly reduce their confidence in the design. Additionally, a litany of small consistency errors robs from the clarity of the design and reduces the effectiveness of SME reviewers, and at the same time even small errors can lead to mission failure so reviewers cannot dismiss even seemingly trifling inconsistencies [39].

Dependency on Transparency of Design Team

A subtle problem with the way reviews are commonly held is that the design team is expected to be forthright and transparent in presenting the current state of the design, and this characterization may be biased. Subject matter experts are expected to ask probing questions and intuit design maturity from answers, and may probe into specific design work products to understand more, but completely reviewing a design as part of a review milestone can be impractical.

Design Frozen during Preparation

In preparation for a formal review, the design is often frozen thirty to sixty days prior to the review to provide designers time to manually capture the state of the design in appropriate work products and charts for the review.

Collaboration Occurs out of Phase with Design

Reviews also serve to force collaboration where disciplines work independently until the start of review preparation. For example, requirements interpretation differences may not be evident until a review preparation cycle; however, errors that surface during these preparation (or review presentations) can result in design rework that could have been avoided if the collaboration occurred earlier or continually.

Process Tailoring Complexity

Tailoring review criteria for different types of programs is challenging. Teams with experience in large space programs have difficulty scaling down the review entrance criteria for smaller programs for fear of excluding critical review material. Reviewers may also impose unnecessary standards on a project without due consideration of the tailoring that may be required, they may not have the required expertise and their feedback may not be relevant or timely enough to influence the project direction. Further reviews do not usually cover organizational problems such as culture, competency and authority problems which can be the root cause of systemic problems [40] which may lead to MA escapes later. Checklists are often utilized as part of the review process but they do not substitute for reviewers with expertise in their review area.

Lessons Learned Rely on SME Reviewers

Reviews may not efficiently or adequately incorporate lessons learned from previous programs into the new design. This knowledge usually comes from SMEs experience. Data base containing lessons learned could evaluate model, identify risk, and suggest improvements to the design. Can even be applied to following best design practices. Reasons for deviations can be captured in the model.

We expect MBE to address these problems going forward, which we describe in the following section; however, the transition between the current and future state will itself create problems that are important for MA practitioners to consider and so we also describe the expected transitions and the relevant MA implications of each in Section 5.2.

Future State

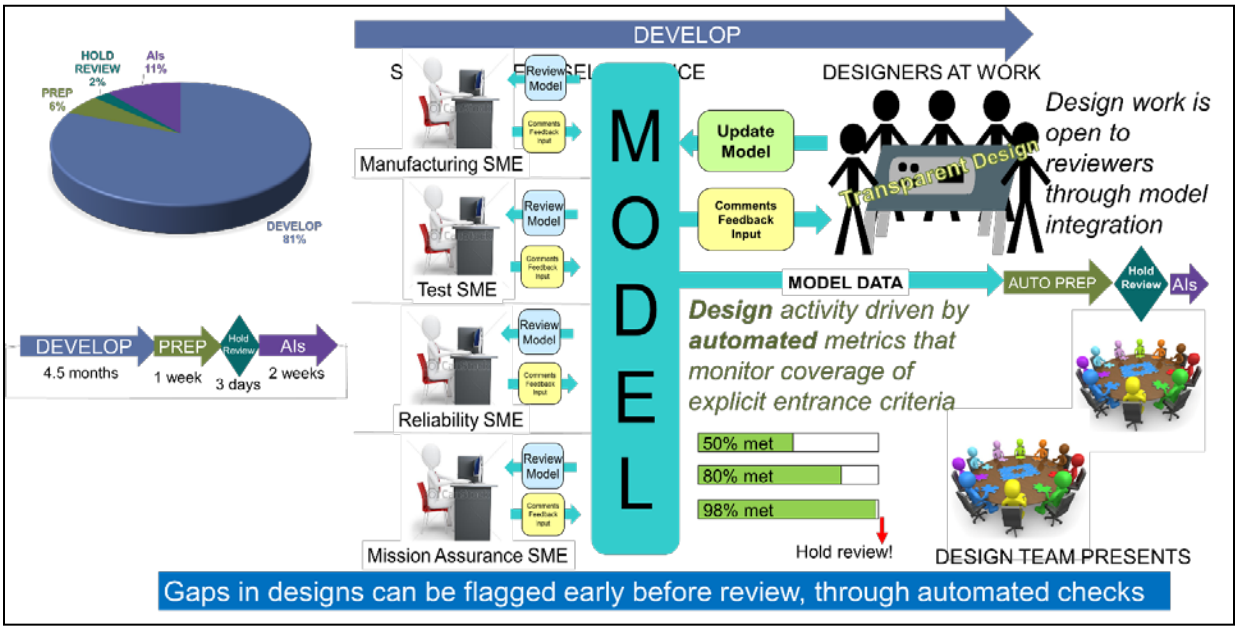

Figure 18. Future independent review process.

In the future state, the design team collaborates using a fully integrated model. SMEs have access to the current design and at any time can provide designers with comments and feedback. Designers can then incorporate SME input into the model. The model is evaluated via automated metrics that monitor coverage of review entrance criteria. When review entrance criteria is met, review material is automatically generated from the model, and a *focused* review is held (note smaller tables in Figure 18 vs Figure 17).

Table 10. Purpose of Interdependent Reviews Current State vs. Future State

Purpose of Independent Review	How (current)	How (end state)
Assess the status of and progress toward accomplishing the planned activities [32].	Present schedule, cost status	Automated assessment of status with standardized metrics and reports
Validate the technical tradeoffs explored and design solutions proposed [32].	Describe rationale of key tradeoffs, decision criteria, baseline design and alternates	Detailed backup to decision rationale integrated in model during collaborative development and viewable by stakeholders, design team summarizes for review
Identifying technical weaknesses or marginal design and potential problems (risks) and recommending improvements and corrective [31, 25, 32].	List predicts, compliance to requirements, risk and plans for risk burndown for high probability risks	Auto generate predicts, compliance, risk plans from model data

Purpose of Independent Review	How (current)	How (end state)
Making judgments on the activities' readiness for the follow-on events, including additional future evaluation milestones to improve the likelihood of a successful outcome [31, 32].	Go/no-go decisions are made by leaders with SME input	Go/no-go based on extended detailed content review driven from model, rather than subset of review content
Making assessments and recommendations to the project team, center, and agency management [32].	Extended review team manually reviews work products, charts, provides feedback	Annotate model directly during review
Providing a historical record, that can be referenced, of decisions that were made during these formal reviews [31, 25, 32].	Review products are captured by CM (charts, etc.), Requirements, allocations, margins, schedules, risks, budgets	Total model, with annotations during review, CM control
Assessing the technical risk status and current risk profile [32].	Assess presented technical baseline, assess risk & status based on presented material. 100% manual assessment.	Assessments done through summarizing results of automated independent model checks (50%+ automated assessment)

Table 11. Independent Review Improvement Goals

Review Property	Current State	Mid State	End State
Quantity	~15 reviews	Same	Same
Frequency	~6-month review spacing	Same	Review spacing determined by completion of explicit entrance criteria in model
Prep time	30-60 day review prep time	Current-state review work products auto generated, manually reviewed (separate models)	Prep automated, new prep focused on internal review of 'dashboard' content (integrated models)
Breadth of involvement	Broad stakeholder involvement (science, program, SMEs)	Same	Stakeholder self-service, pull coherent tech baseline on-demand anytime in lifecycle, give real time feedback
Review CONOPS	Formal PPT-type presentation by program team, stakeholder in-person	Generate key content of review PPT deck from model with automation, stakeholder in-person.	Two phase review, A) broad collaborative on integrated model, content coverage measured by analytics B) focused Q/A on key areas surfaced in (A).
Schedule impact	Development stopped during prep/review cycle, waits for 'go' decision	Reduced prep time due to auto-gen of portions of PPT deck	Eliminate prep time for review and associated schedule impact
Team impact	Loss of design momentum during prep/design context switch	Team must prep for areas of review content not automated	Reduce impact in prep phase from auto-prep, team must still review content, update baseline
Design inconsistency checks	80% found manually in prep, 10% found in review, 10% latent	85% found manually in prep, 10% found in review, 5% latent (reduced through key content generation)	90% found in auto review prep through analytics, 10% found in review (A/B), <0.5% latent defects left

Table 12. Top Ten Risks in Space Launch Missions

Top Risks in Space Launch Missions [14]	Current Identification Method	Future State Identification Method
Test-Like-You-Fly Exceptions: hardware and/or software testing is not representative and end-use case.	Identified manually	Exceptions are still identified manually process, but fully integrated model facilitates identification of these exceptions
Critical Qualification Margins: hardware with minimal safety margins	Identified manually for critical items	MBE provides checks for qualification margins of critical items
First-Flight Items: items that have not been demonstrated to work under actual flight conditions	Identified via review of the bill of materials	Bill of materials part of MBE model
Single-Point Failures	Identified manually and documented in the reliability analysis	Analysis of MBE model will identify single point failures
Nonconformance: hardware or software that does not meet specification	Manually review customer justification for use "as is"	Fully integrated model aids in analysis/ justification of whether hardware can be used "as-is"
Anomalies: hardware or software did not perform as expected	Careful review and analysis required to determine root cause	MBE will facilitate determination of root cause
Escapes: events in which the contractor missed something	Identified manually	Computer analytics will aid in locating escapes
Unverified Failures: root cause of failure not identified	Manually detail cause and effect relationships and potential root causes and remedies	Computer analytics will aid in identifying potential root causes
Out-of-Position/Sequence Work	Review team manually reviews impact on the mission, and offers recommendations	Out of sequence work is tracked and managed. Process adherence is enforced through tool constraints.
Out-of-Family Results	Manual statistical process control is utilized	MBE will facilitate trending analysis to identify out-of-family results and identify any changes in the production process

F.4. Transition and Implementation Recommendations

During the transition from traditional reviews to model-based reviews, we see three main transition stages: review of essentially the same content as today but linked in an integrated model, review of domain and subject-matter specific views of the model tailored for the audience, and automated review checkers that handle most of the review tasks.

The first stage relies on the traceability of model elements between various tools and engineering design and analysis work. Today, most model-based efforts are focused on traceability and linking, and so the transition from current practice to this first stage should be relatively simple.

Following the first stage we see tailored reviews that will be driven by the need of SMEs to review complex integrated models without being held back by the inherent complexity of these models. This may involve viewpoint design based on discipline (e.g., thermal, power, communications, optical, RF) and allow SMEs in these areas to completely walk through the technical baseline from their perspective to assess maturity and risk. We also expect these tailored viewpoints to extend to non-technical areas such as the cost and schedule baseline.

This second stage prepares us for the final automated stage where most review tasks are done automatically using a combination of machine learning and human computation. Here, machines are

trained to know what to look for based on legacy program data gathered during the first two stages, along with historical recording of reviewer actions in the second stage. The potential is great in this last stage for a dramatic positive impact on MA, especially as this approach is applied broadly across all mission classes. Initially humans will provide review feedback on ambiguous areas, but over time as machine learning algorithms build up adequate knowledge then manual review steps will become less and less frequent. Finally, these automated review systems can be deployed earlier within the design cycle, and thus give the earliest and quickest response to design decisions—even responding in real time as design modifications occur.

To gain the most MA impact of MBE on independent reviews the transition from early stages to the final automated stage should be prioritized. This first involves open capture of technical baseline data in a way that can be commonly understood. One way to help this along is standardizing on model ontologies. Work within industry on ontology tends to be product-based, and in organizations like JPL they tend to be mission-focused. It is likely that ontologies will play a key role in allowing automated model checkers that speed independent reviews, and so we recommend studies that consider how ontologies can help in the MA area for model-based projects.

F.5. Risks and Challenges

1. Automation risks
   a. Automated model views need to be tailored to each reviewer's background increasing deployment cost for reviews and reducing return on investment (ROI). The risk here is in anticipating the reviewer's needs properly and working out the interaction between visualizations and reviewers in a way that is efficient and effective.
   b. Increased reliance on automation also increases the likelihood of a systemic fault passing through unnoticed by the automation.
   c. MA practitioners may not accept or trust automated results or automation escapes can lead to abandoning automation entirely in favor of legacy manual processes.
   d. Resistance to change from manual to automated within engineering disciplines due to perceived loss of work, retraining, entrenched workflows/tools/skills.
2. Infrastructure investment challenges:
   a. Systems and databases outside the normal MBE realm must be integrated to provide MA information together with the technical, cost, schedule baselines increasing integration costs, and data including:

      - Nonconformance information
      - Bill of material historical data (first flight)
      - Action item tracking systems
      - Test results
      - Manufacturing databases

   b. Out of sequence work problems and process adherence can be tracked and managed with current technology and connected to integrated system models but entail significant infrastructure investment that may not be warranted for the benefit of a single contractor, example technologies include:

      - Paperless test procedure planning, control, and execution
      - Tool instrumentation and electronic integration
      - Machine vision

3. Ramp-up time for reviewers on model may be prohibitive:
   a. If we keep the same review team throughout the review cycle then ramp-up time is reduced; however, diversity of perspective suffers and this may reduce model coverage.
   b. If a variety of reviewers is used between events then reviewed material needs to be broadly intuitive, and browse-able by technical competent but non-expert reviewers as well as experts.

Lessons Learned

1. Currently available MBSE tool suites and graphics standards are complex enough to be unusable outside modeler and advanced systems engineer communities, and this often limits collaboration to systems team for MBSE models.

2. PowerPoint-type diagrams, that simplify and summarize various aspects of a system, continue to be needed even though they often duplicate model content. Automated layout technology exists but application to reviews is in its infancy, currently.

3. There are no comprehensive open data exchange format standards that also retain views and layouts of diagrams between tools, even within the MBSE tool space, when considering the wider set of MBSE tools information sharing is difficult and often requires custom scripts and automation. Tool vendors are addressing this individually but no independent standard has been adopted as of this writing. When extended into the PLM space, all vendors have the same basic strategy, namely to lock you into their tool echo systems and then upsell more features. This strategy is challenged by open data exchange and so vendors really have a disincentive to implement this.

What are the Questions MA Practitioners Should Ask for an MBE-Based Project?

1. How do I know this is the latest model view being reviewed?
2. How are changes controlled and governed for the model(s)?
3. How can we be assured that the model checkers catch errors?
4. How do reviewers know the changes and their significance from the prior review?
5. Was new functionality added or removed from the prior review, and how do the review materials reflect this?
6. What actions were taken from the last review and how can we see from the model that the proposed changes and dispositions were incorporated in the baseline?

Recommendations for Transitioning to Future State

The infrastructure challenges for achieving the MBE vision require a significant investment as part of each company's internal infrastructure and, as with any large Information Technology (IT) projects, success is not assured, in fact by some estimates less than 20 percent [5] of large IT infrastructure projects are successful. Organizations like NASA JPL have invested heavily in transforming their IT infrastructure to support more broad use of MBE/MBSE and often publish the result of their labor in open-source repositories for use by others. Though these resources are available understanding and adopting this infrastructure is difficult for the industry because it is not integrated into a framework that is built for wide adoption and deployment. There would be value in starting an open-source community surrounding MBE for space systems in particular. Within this community, the required frameworks, tools, and integrations can be developed to help push adoption of MBE along quicker by reducing each company's up-front investment and reducing risk of taking on a large IT infrastructure development project.

How CDRLs May Change:

For formal reviews that are called out contractually, review materials are often requested as a deliverable at some time before the review (e.g., two weeks prior to review). In the future with model based, there may be value in a continual delivery of incremental information as the design progresses rather than a large delivery prior to a review. This would allow for early feedback, improved stakeholder alignment and would reduce the likelihood of surprises during the review process. How this delivery occurs may be far different and instead of a formal delivery there may be access to model entry points relevant to the customer. This method of delivery reduces the overhead of preparing work products for delivery; however, there are perils on both the customer and the contractor side. For the customer, they must have the ability to de-couple their model view from the contractor's infrastructure such that they can refer to the model in the future even if the contractor ceases operations or decommissions infrastructure. For the contractor, they must consider when system information is mature enough to share, and therefore receive feedback and comments against. With this in mind, review CDRLs may transition to call outs for a certain level of model fidelity. There may be certain expected information in the contractor's model prior to a major milestone but with "access" to the model starting from some earlier date and with little expectation for maturity. The CDRL for the MBE approach will specify the model elements with applicable details relevant to the program phase to be included in the model.

General MA Recommendations:

1. Review completeness of the model: how mature is the information contained in the model? Look for indications of model completeness and model quality (are there orphan requirements, floating parts, etc.)

2. Run validated model checkers
   a. Ensure a valid relationship exists between functions, requirements, and hardware
   b. Ensure interfaces are defined and designed properly
   c. Proper qualification margins
   d. Identify single-point failure

3. Review configuration control process of the model
   a. What is the latest version of the model
   b. What has changed in the model and why
   c. Who approved/authorized the change

4. Consider volatility of the model as an indicator of risk
   a. See what areas of the model are changing the most, and which are stable. Does this information coincide with the program plan, directed program changes, etc.

5. Assess gap between model and as-built configuration.

Appendix G. Summary of MA Process Areas

The selected MA processes are detailed in the MA MBE environment within this document to supplement TOR-2011(8591)-21 [27].

External Distribution

REPORT TITLE

Mission Assurance Considerations for Model-Based Engineering for Space Systems

REPORT NO.	PUBLICATION DATE	SECURITY CLASSIFICATION
TOR-2017-01695	August 31, 2017	UNCLASSIFIED

Chuck Abernety
Aerojet Rocketdyne
charles.abernethy@aerojet.com

Bob Andrews
Ball
randrews@ball.com

Matt Beckner
Blue Canyon
mbeckner@bluecanyontech.com

Robert Adkisson
Boeing
robert.w.adkisson@boeing.com

Keith Atagi
NGC
keith.atagi@ngc.com

Greg Berg
Boeing
greg.g.berg@boeing.com

Suzanne Aleman
NASA
suzanne.m.aleman@nasa.gov

Mark Balwin
Raytheon
mark.l.baldwin@raytheon.com

Wayne Blackwood
NOAA
Wayne.blackwood@noaa.gov

Yaana Allen
OrbitalATK
yaana.allen@orbitalatk.com

David Barnhart
U of Southern California
barnhart@isi.edu

Rosemary Brester
Hobart Machined
rosemary@hobartmachined.com

Scott Anderson
SEAKR Engineering
Scott@seaker.com

Theresa Beach
MetiSpace
tbeech@metispace.com

Bill Burk
Raytheon
wburk@raytheon.com

Jennifer Bryne
Lockheed Martin
jennifer.c.byrne@lmco.com

Jerry Cogen
Frequency Electronics
jerald.cogen@freqElec.com

Larry DeFillipo
OrbitalATK
Lawrence.defillipo@orbitalATK.com

Kerri Cahoy
MIT
kcahoy@mit.edu

Bill Cook
OrbitalATK
william.cook@orbitalatk.com

Renelito Delos Santos
SSL
renelito.delos-santos@sslmda.com

Robert Choo
Boeing
robert.choo@boeing.com

Stephen Cross
ULA
stephen.d.cross@ulalaunch.com

Andrew Demo
NASA
andrew.g.demo@nasa.gov

Mike Chory
MIT LL
michael.chory@ll.mit.edu

Jamie Cutler
U of Michigan
jwcutler@umich.edu

Tracy Dillinger
NASA
tracy.dillinger@nasa.gov

Ray Chowdhury
NOAA
ramin.chowdhury@mda.mil

Steve Danley
Frontier Electronics
steved@fescorp.com

Tony Dotson
Nye Lubricants
tdotson@nyelubricants.com

Jeffrey Christensen
Boeing
jeffrey.a.christensen@boeing.com

Dave Davis
SMC
david.davis.3@us.af.mil

Brian Douglas
Planetary Resources
brian@planetaryresources.com

Brad Clevenger
SolAero
brad_clevenger@solaerotech.com

Mike Dean
Ball
mdean@ball.com

Stan Dubyn
Millennium Space Systems
Stan.dubyn@millennium-space.com

Harry Durettle
Self
harry.c.durette.civ@mail.mil

Tony Fernandez
Harris
aferna16@harris.com

Conor Galligan
MIT LL
conor.galligan@ll.mit.edu

Nadia El-Sherief
Raytheon
nadia.el-sherief@raytheon.com

Lance Fife
Utah State/SDL
lance.fife@sdl.usu.edu

Howard Gans
Harris
hgans@harris.com

Barbara Erbacher
Ball
berbache@ball.com

Rich Fink
NRO
richard.fink@nro.mil

Jace Gardner
Ball
jgardner@ball.com

John Evans
NASA
john.w.evans@nasa.gov

Chad Fish
Astra Space
cfish@astraspace.net

Rick Gebbie
MIT LL
rgebbie@ll.mit.edu

Martin Feather
JPL
martin.s.feather@jpl.nasa.gov

Mike Floye
General Dynamics
mike.floyd@gd-ms.com

Dave Gianetto
Raytheon
gianetto@raytheon.com

Terry Feehan
OrbitalATK
Terry.feehan@orbitalatk.com

Teressa Franks
NOAA
teressa.franks@noaa.gov

Scott Gibbons
Harris
sgibbons@harris.com

Todd Fenimore
Lockheed Martin
todd.w.fenimore@lmco.com

Bill Galary
Nye Lubricants
bgalary@nyelubricants.com

Ricardo Gonzalez
BAE Systems
ricardo.gonzalez@baesystems.com

Chuck Gray
Frontier Electronics
chuckg@fescorp.com

Bob Hoffman
Nye Lubricants
rhoffman@nyelubricants.com

Thomas Johnson
NASA
Thomas.e.Johnson@nasa.gov

Dan Gresham
OrbitalATK
daniel.gresham@orbitalatk.com

Lars Hoffman
SpaceX
lars.hofman@spacex.com

Edward Jopson
NGC
edward.jopson@ngc.com

Jed Hancock
Utah State/SDL
jed.hancock@sdl.usu.edu

Jerry Holsomback
Raytheon
jerry.b.holsomback@raytheon.com

Alisa Joseph
NGC
alisa.joseph@ngc.com

Mark Hanson
SSL/MDA
Mark.hanson2@sslmda.com

David Hook
Harris
dhook01@harris.com

Geoffrey Kaczynski
NEA Electronics
gpkaczynski@eba-d.com

Vivek Hazari
SpaceX
vivek.hazari@spacex.com

Pablo Hopman
MIT LL
hopman@ll.mit.edu

Jin Kang
US Naval Academy
kang@usna.edu

Kevin Hefner
Harris Exelis
kevin.hefner@exelisinc.com

Charlene Jacka
AFRL/RV
Charlene.jacka.1@us.af.mil

Fred Kelso
MDA
frederick.kelso@mda.mil

Mike Herzog
Pacific Scientific
mherzog@psemc.com

David Johnson
Honeywell
david.c.johnson@honeywell.com

Kyle Kemble
AFRL/RV
Kyle.Kemble.2@us.af.mil

Mark King
Micropac
markking@micropac.com

Gary Kushner
Lockheed Martin
gary.d.kushner@lmco.com

Aliki Loper-Leddy
SSL
aliki.loper-leddy@sslmda.com

David Klumpar
Montana State U
klumpar@physics.montana.edu

Ken Label
NASA
Kenneth.a.label@nasa.gov

Frank Lucca
L-3
frank.l.lucca@l-3com.com

Byron Knight
NRO
knightby@nro.mil

CJ Land
Harris
cland@harris.com

Mike Lutomski
SpaceX
michae.lutomski@spacex.com

Hans Koenigsmann
SpaceX
hans.koenigsmann@spacex.com

Jesse Leitner
NASA
Jesse.leitner@nasa.gov

Richard Lutz
SolAero
Richard_Lutz@solaerotech.com

Brian Kosinski
SSL
brian.kosinski@sslmda.com

Scot Lichty
Lockheed Martin
scot.r.lichty@lmco.com

Brian Maguire
Ball
bmaguire@ball.com

John Kowalchik
Lockheed Martin
john.j.kowalchik@lmco.com

Glenn Lightsey
Georgia Tech
glenn.lightsey@gatech.edu

Dan Mamula
NOAA
Dan.mamula@noaa.gov

Steve Kuritz
NGC
steve.kuritz@ngc.com

Frank Lombardo
Harris
frank.lombardo@harris.com

Ronald Mandel
Lockheed Martin
ronald.h.mandel@lmco.com

Bob Manthy
Ball
rmanthy@ball.com

Kenneth McGill
MDA
kenneth.mcgill@mda.mil

Ed Moshinsky
Lockheed Martin
edward.a.moshinsky@lmco.com

Jamal Mardini
Boeing
jamaleddine.mardini@boeing.com

Geoff McHargue
US Air Force Academy
matthew.mcharg@usafa.edu

Cynthia Nafus
ULA
cynthia.l.nafus@ulalaunch.com

Patrick Martin
NASA
patrick.martin@nasa.gov

KurtMeister
Honeywell
kurt.meister@honeywell.com

John Nelson
Lockheed Martin
john.d.nelson@lmco.com

Steven Martin
SMC
steven.martin.36@us.af.mil

Melissa Meyers
JPL
melissa.a.meyers@jpl.nasa.gov

Andreas Nonnenmacher
UTCS
andreas.nonenmacher@uts.utc.com

Steven Mayers
MDA
Stephen.mayers@mda.mil

Eli Minson
Ball
eminson@ball.com

David Novotney
NEA Electronics
dbnovotney@eba-d.com

Michael McCarrick
Boeing
michael.f.mccarrick@boeing.com

Miquel Moe
NASA
miquel.a.moe@nasa.gov

Ronald Nowlin
Eagle Picher
ron.nowlin@eaglepicher.com

Bill McGeary
L-3
william.l.mcGeary@l-3com.com

Philip Montag
Honeywell
philip.Montag@honeywell.com

Ryan Nugent
Cal Poly SLO
rnugent@calpoy.edu

Alfredo Nunez
Boeing
alfredo.nunez2@boeing.com

Frank Pastizzo
SSL/MDA
frank.pastizzo@sslmda.com

Thomas Pham
Boeing
thomas.v.pham@boeing.com

Larry Ostendorf
Pacific Scientific
lostendort@psemc.com

Rich Patrican
Raytheon
richard.a.patrican@raytheon.com

Dave Pinkley
Ball
dpinkley@ball.com

Jeff Osterkamp
Ball
josterka@ball.com

Pat Patterson
Utah State/SDL
Pat.Patterson@sdl.usu.edu

Paul Pinner
Boeing
paul.r.pinner@boeing.com

Joseph Packard
Harris
joseph.packard@harris.com

Steven Pereira
John Hopkins/APL
Steven.Pereira@jhuapl.edu

James Poirier
Boeing
james.v.poirier@boeing.com

Regina Palmer
Lockheed Martin
regina.palmer@lmco.com

Mike Perez
Lockheed Martin
mike.a.perez@lmco.com

Robert Pollard
Ball
rpollard@ball.com

Scott Pano
NGC
scott.pano@ngc.com

Ronald Persin
MDA
ronald.persin.ctr@mda.mil

Luis Ponce
OrbitalATK
luis.ponce@orbitalatk.com

Mark Pasquale
Lockheed Martin
mark.pasquale@lmco.com

Amy Peters
OrbitalATK
Amy.peters@OrbitalATK.com

Mark Porter
General Dynamics
mark.porter@gd-ms.com

Curtis Potterveld
Boeing
curtis.w.potterveld@boeing.com

Mike Rice
Kratos Defense
mrice@relogic.com

Mike Sampson
NASA
Michael.j.sampson@nasa.gov

Tim Priser
Lockheed Martin
timothy.a.priser@lmco.com

Luis Rodriguez
Boeing
luis.rodriguez@boeing.com

Bill Sargent
Boeing
william.s.sargent@boeing.com

Jordi Puig-Suari
Cal Poly SLO
jpuigsua@calpoly.edu

Reuben Rohrschneider
Ball
rrohrsch@ball.com

Chris Schreiber
Lockheed Martin
chris.schreiber@lmco.com

Anne Ramsey
Harris
aramsey@harris.com

Joyce Ross
NGC
joyce.m.ross@ngc.com

Philip Scott
Chemring Energtic
pscott@ced.us.com

Ben Randolph
SSL/MDA
Ben.randolph@sslmda.com

Nigel Rowe
Boeing
nigel.c.rowe@boeing.com

Mark Seay
SSL
mark.seay@sslmda.com

David Rea
BAE Systems
david.a.rea@baesystems.com

Bill Rozea
Aerojet Rocketdyne
william.rozea@rocket.com

Bill Sharp
Boeing
william.c.sharp@boeing.com

Brian Reilly
DCMA
brian.reilly@dcma.mil

Cynthia Rueckert
Ball
cmruecke@ball.com

Dave Shelton
Lockheed Martin
dave.shelton@lmco.com

Gwynne Shotwell
SpaceX
Gwynne.shotwell@spacex.com

Laurie Stupak
Ball
lstupak@ball.com

Adrian Tudor
OrbitalATK
adrian.tudor@orbitalATK.com

Jeff Shykula
Ball
jshykula@ball.com

Dave Swanson
OrbitalATK
david.swanson@orbitalatk.com

Deb Valley
MIT LL
deborah.valley@ll.mit.edu

Robert Sinclair
Ball
rsinclai@ball.com

Michael Swartwout
Saint Louis U
mswartwo@slu.edu

Monifa Vaughn-Cooke
U of Maryland
mvc@umd.com

Kathleen Smidt
COMDEV USA
kathi.smidt@comdev-usa.com

Anthony Taconi
Lockheed Martin
anthony.r.taconi@lmco.com

Mike Violet
OrbitalATK
michael.violet@orbitalatk.com

Dave Staley
OrbitalATK
David.staley@orbitalATK.org

Larry Tew
Center of Error
LtewCEM@aol.com

James Wade
Raytheon
james.w.wade@raytheon.com

Rob Stefan
Boeing
robert.j.stefan-jr@boeing.com

Mike Tolmasoff
Boeing
mike.w.tolmasoff@boeing.com

David Wayne
US Navy/SPAWAR
David.t.wayne@navy.mil

Spencer Studley
SSL/MDA
spencer.studley@sslmda.com

Nyla Tuck
NRO
myla.tuck@nro.mil

Howie Webber
SSL
howie.webber@sslmda.com

Craig Wesser
NGC
craig.wesser@ngc.com

Bruce Yost
NASA
Bruce.d.yost@nasa.gov

Andrew Whiting
Boeing
andrew.whiting@boeing.com

Connie Zarate
SSL
connie.zarate@sslmda.com

Tom Wiedenbauger
Harris
twiedenb@harris.com

Alan Zoyhofski
Harris
azoyhofs@harris.com

Catherine Wilson
Boeing
catherine.a.wilson@boeing.com

Lenny Rosenheck
Boeing
leonard.roenheck@boeing.com

Jerry Winton
SolAero
jerry_Winton@solaerotech.com

Mike Worcester
Boeing
Michael.s.Worcester@boeing.com

Carter Wright
Ball
cwright@ball.com

APPROVED BY_____ DATE_____
(AF OFFICE)

AEROSPACE REPORT NO.
TOR-2017-01695

Mission Assurance Considerations for Model-Based Engineering for Space Systems

Approved Electronically by:

Jacqueline M. Wyrwitzke, PRINC DIRECTOR
MISSION ASSURANCE SUBDIVISION
SYSTEMS ENGINEERING DIVISION
OFFICE OF EVP

Cognizant Program Manager Approval:

Todd M. Nygren, GENERAL MANAGER
SYSTEMS ENGINEERING DIVISION
ENGINEERING & TECHNOLOGY GROUP

Aerospace Corporate Officer Approval:

Malina M. Hills, SR VP SPACE SYS
SPACE SYSTEMS GROUP

Content Concurrence Provided Electronically by:

Albert C. Hoheb, PROJECT LEADER SR
SYSTEMS ENGINEERING DIVISION
ENGINEERING & TECHNOLOGY GROUP
OFFICE OF EVP

SQ0232

AEROSPACE REPORT NO.
TOR-2017-01695

Mission Assurance Considerations for Model-Based Engineering for Space Systems

Technical Peer Review Performed by:

Jacqueline M. Wyrwitzke, PRINC DIRECTOR
MISSION ASSURANCE SUBDIVISION
SYSTEMS ENGINEERING DIVISION
OFFICE OF EVP

SQ0232