Model Based Mission Assurance (MBMA): NASA's Assurance Future

John Evans, PhD, NASA OSMA

Steven Cornford, PhD, JPL

Martin S. Feather, PhD, JPL

Key Words: Assurance, Model Based Systems Engineering

SUMMARY & CONCLUSIONS

Model Based Systems Engineering (MBSE) is seeing increased application in planning and design of NASA's missions. This suggests the question: what will be the corresponding practice of Model Based Mission Assurance (MBMA)?

Contemporaneously, NASA's Office of Safety and Mission Assurance (OSMA) is evaluating a new objectives-based approach to standards to ensure that the Safety and Mission Assurance disciplines and programs are addressing the challenges of NASA's changing missions, acquisition and engineering practices, and technology. MBSE is a prominent example of a changing engineering practice.

We use NASA's objectives-based strategy for Reliability and Maintainability as a means to examine how MBSE will affect assurance. We surveyed MBSE literature to look specifically for these affects, and find a variety of them discussed (some are anticipated, some are reported from applications to date). Predominantly these apply to the early stages of design, although there are also extrapolations of how MBSE practices will have benefits for testing phases.

As the effort to develop MBMA continues, it will need to clearly and unambiguously establish the roles of uncertainty and risk in the system model. This will enable a variety of uncertainty-based analyses to be performed much more rapidly than ever before and has the promise to increase the integration of CRM (Continuous Risk Management) and PRA (Probabilistic Risk Analyses) even more fully into the project development life cycle.

Various views and viewpoints will be required for assurance disciplines, and an over-arching viewpoint will then be able to more completely characterize the state of the project/program as well is (possibly) enabling the safety case approach for overall risk awareness and communication.

1 INTRODUCTION

Model Based System Engineering (MBSE) – for a survey, see [1] – is beginning to see substantial use within NASA for the planning and design of space missions. The promise of MBSE includes a single authoritative source of truth, correct by construction models, scalability, discipline-specific viewpoints, customizable interface, and fully integrated design environments. For these reasons and more, it is anticipated MBSE will enable more capable missions without sacrificing cost-effectiveness despite increase in complexity. Because of its growing adoption in the aerospace industry and because it is imperative that there is also no sacrifice of safety and mission success, NASA's OSMA has initiated a roadmapping effort to pave the way for full integration of mission assurance into this model-based world – "Model Based Mission Assurance."

Contemporaneously, NASA has begun to develop Objective Structure Hierarchies [2]. As stated therein, *"Creating a hierarchy as a basis for assurance implementation is a proven approach that has emerged from the safety case ideals. This approach holds the opportunity to enable new directions in an evolving design framework, in which models will govern optimization to achieve the best designs and prescribed documents will take a back seat."* NASA's hierarchies consist of strategies and objectives that build upon each other to support their top objective. For example, the Reliability and Maintainability hierarchy has as its top objective "*System performs as required over the lifecycle to satisfy mission objectives*." This subdivides into:

> *1. System conforms to design intent and performs as planned.*
> *2. System remains functional for intended lifetime, environment, operating conditions and usage.*
> *3. System is tolerant to faults, failures and other anomalous internal and external events.*
> *4. System has an acceptable level of maintainability and operational availability.*

To date, the *Reliability and Maintainability*, *Software Assurance*, *ELV Payload* and *Range Safety* hierarchies have been developed, and Quality Assurance and EEE Parts hierarchies' development is underway. As reflected in Figure 1, they are to serve as the starting points from which to develop system-specific safety or assurance cases, central to the direction in which system safety and reliability is evolving within NASA [3],[4]. This will occur in an environment where the assurance of the system is shared within the modeling frame work.

We have surveyed MBSE literature to find how MBSE capabilities have the potential to support assurance. In section 2 we compare those capabilities to the strategies and objectives of the Reliability and Maintainability hierarchy to estimate how and where they will contribute to assurance. From this perspective we see that sub-objectives 1, 2 & 3 (from the list above) are well covered by MBSE literature through the stages of system concept, requirements and design, including validation and verification during those stages. For the subsequent stages: development (build, integrate, test), operation, and maintenance, there is some discussion of MBSE's support for testing, and some mention of (software) code generation from models, but overall relatively little is said about these later lifecycle stages. However, given the evolving practice of MBSE, this lack of application in later project phases is expected since notions familiar to OSMA have yet to be fully integrated into MBSE. Thus, the time is right for inclusion into MBSE.

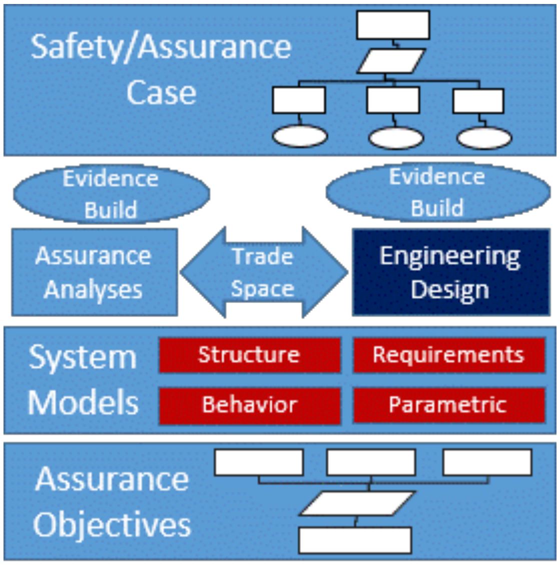

Figure 1 - Assurance Framework in Model Based Environment

2 MBSE'S SUPPORT FOR ASSURANCE OBJECTIVES

We surveyed relevant MBSE literature to find how MBSE approaches are being applied to support assurance needs. This section presents s summary, organized into 7 areas. We indicate each area's relevance to NASA's R&M Objectives Hierarchy [5]. More details of each area are given in the following section.

1. *Representation and management of systems engineering information to ensure consistency and (some aspects of) completeness.* Rigorous model-based representation (i.e., with a semantic underpinning that ensures a shared, unambiguous understanding) of systems engineering information is the hallmark of MBSE. This has cross-cutting relevance and benefit to the entire R&M Objectives Hierarchy to the extent that MBSE is carried through the mission lifecycle. May provide specific benefit in 1.B.1.A (Test, inspect, and demonstrate to an acceptable level to ensure that issues are found) through heading off subtle and hard to detect problems that stem from misinterpretations prevalent when less rigorous documentation is the norm.

2. *Support of the contractual interface between acquirer and potential suppliers*. Relevant to the overall purpose of using NASA's Objectives Hierarchies as the means for a developer or service provider to communicate assurance information to NASA.

3. *Generation of review documentation from the shared MBSE system model*. Preliminary results from ongoing NASA applications show evidence of benefits to R&M Hierarchy 1.A.1.A (Demonstrate to an acceptable level that the functionality of the system meets the design intent). E.g., for generation of review material at key decision points.

4. *Automated assistance for generating reliability artifacts (FMECAs, Fault Trees, etc.).* Relevant to the sub-objective 2 (System remains functional for intended lifetime, environment, operating conditions and usage) and 3 (System is tolerant to faults, failures and other anomalous internal and external events). However, results to date have not extended into later phases of development, most especially operations.

5. *Representation of and reasoning about off-nominal states and behaviors*. A fundamental capability supporting the previous area (reliability artifacts), therefore likewise relevant to sub-objectives 2 and 3.

6. *Support for activities post-design.* While most of the MBSE literature focuses on the design stage, some addresses later activities, notably testing, Planning for and managing the testing activities could potentially benefit from the same MBSE principles of capturing the pertinent information in a formal representation, relevant to: 1.B.1.A, 2.A.1.D (Perform qualification testing and life demonstration to verify design for intended use), 2.B.1.D (Plan and perform life testing), 4.A.1.G (Provide demonstration testing to verify 'detect, diagnose, isolate' capability of systems and confirm corrective and preventive maintenance task actions and analysis) and 1.C.1.D (Screening, proof testing and acceptance testing). In the software arena there are instances of software being automatically generated from models, thus contributing to 1.C (Achieve high level of process reliability), and of software being extensively tested against computer simulations of the system and its operating conditions.

7. *Correctness of the MBSE models themselves*. Since the system design information is captured in models, it is crucial that they be correct, with obvious cross-cutting relevance. This is an area where assurance practices can contribute to the MBSE methodology.

3 DETAILED LITERATURE SURVEY

3.1 Representation and Management of systems engineering information

In general, many papers and presentations argue for the benefit of model-centric rather than document-centric design; the following examples report application to space systems.

[6] briefly reports on the integration of the flight software development James Webb Space Telescope's Integrated Science Instrument Module Instrument JWST ISIM C&DH Core Flight Software (FSW) development at GSFC and the Science Instrument FSW applications developed by different

teams at several disparate locations. A Rational Rose Structure Diagram was used to unambiguously indicate the core system's communication ports to other subsystem components and to its own internal "capsules".

[7] describes use of MBSE to manage development of the ground system for control of the OSIRIS-Rex spacecraft during its encounter with asteroid Bennu. Three areas are listed as benefiting from MBSE: representation and flowdown of the requirements on the ground system, representation of its architecture (leading to development of the formal documentation of that architecture in the form of Interface Control Documents and Operational Interface Agreements), and capture of testing and V&V plans. In addition to diagramming static views of the architecture, FFBDs (Functional Flow Block Diagrams) are used to support discrete-event simulation of system behaviors to validate the system's run-time behavior.

Similar benefits are reported in [8] in the related area of systems engineering of astronomical telescopes.

3.2 Support of the contractual interface

[9] "reports on research practices pertaining to methods, tools, and techniques proposed to facilitate the use of MBSE across the contractual interface in a competitive tender environment." The authors assert that MBSE has long been successfully applied across contractual boundaries in settings where "mutual trust is well developed and mutual goals are well understood." Their paper addresses the situation of a *competitive* environment, where a supplier would wish to excise proprietary information from the model they submit as part of the bid, and the acquirer would wish to excise certain sensitive information (e.g., costing, management) from the model they put forth to elicit bids. The paper lists details on these topics resulting from "workshops with key stakeholders." They go on to mention that the acquirer's model served as a good starting point for the supplier to elaborate further into a more detailed model.

3.3 Generation of review documentation

[10] contrasts "Document/Expert – Centric Acquisition" with "Data-Driven/MBSE Acquisition", suggesting many of the analysis results needed for acquisition decisions can be machine-generated from models, replacing labor intensive human assessments by teams of experts.

[9] describes textual forms of the acquirer's Operational Concept Document, Function and Performance Specification and Test Concept Document as being generated from reference model. It also goes on to say that some members of the supplier team worked directly from the model, and those that initially preferred to work from the generated textual forms as increasingly switching to the model.

[11] reports a pilot study on a "moderate size flight project" – MISSE-X, a payload for hosting experiments, to be installed on the exterior of the ISS. The report discussed the pros and cons of MBSE in preparation for the project's System Requirements Review. They used:

- SysML to "document the system concept of operations as well as some assembly, integration and test activities" (use-case diagrams for stakeholder interactions, activity diagrams to document system functions, activity diagrams, state-machine diagrams and sequence diagrams to document intended behavior, package diagram of system architecture and external boundaries, IBD for flows and interfaces between systems) and
- Vitech's CORE™ to manage the requirements (traceability through levels of requirements, allocations, owners and verification methods).

They report benefits of consistency, ease of access to complete and current information, and clarity across the team. The review need was to demonstrate existence of a feasible and satisfactory system design (feasible = could be implemented consistent with cost, schedule & risk; satisfactory = design meets project goals).

Exports from CORE used to provide information to team members without requiring them to be CORE users, and to generate complete documents directly from CORE. Review materials were developed from the SysML and CORE models.

However, their paper identifies several challenges:

- "no guarantee that the model is correct" – note that they did not pursue extensively execution of models, stating "developing executable models within the model itself was found to be challenging"
- Restarts in creating the meta-model (ontology)
- Default presentation options from SysML tool often needed extensive re-working.

3.4 Generation of reliability artifacts

There are numerous examples of MBSE in use to provide automated assistance to generate reliability artifacts (FMECAs, Fault Trees, etc.).

[12] & [13] (see also the next section for further papers by the same set of authors) describe generation of FMEAs from SysML information, specifically from Sequence Diagrams (SDs) and Internal Block Diagrams (IBDs). They assume a database (referred to in the paper as a "Dysfunctional Behavior Database") of components and their failure properties – their failure modes, and (optionally) additional information such as failure rate. They further allow for the following:

- A Parametric Diagram (PD) expressing the computation of functional attributes degradation during the failure mode occurrence.
- A PD indicating the computation of the failure rate of each failure mode. (This constraint depends on the environmental and structural parameters)
- A Statechart Diagram describing the dynamic behavior of the component in the failure mode state.

They stress that in IBDs they utilize two kinds of ports – standard ports and flow ports. The former are suited for representing control and command requests (including exchange of information); the latter are suited to representation of data, material or energy flowing through the connectors to/from such ports.

[14] is a concise report of scalable, automated generation

of a FMEA, illustrated on a model of a satellite, its ground control system, and ground users. They assume a SysML model with BDDs, IBDs (in which failure propagation paths are represented), state transition machines (including both normal and failed states) and activity diagrams (that generate the triggers driving state transitions). From these they also automatically generate a model to input into AltaRica ("a tool and language implementing mode automata") to model fault propagation.

[15] & [16] report collaborations between Johnson Space Center and Tietronix Software Inc. [15] illustrates generation of a FMECA and a Fault Tree from a SysML model. They too assume that SysML IBD has the details of the system architecture and that the state transition diagrams include nominal and off-nominal (failed) states, using state machine events and guards to encode propagation of failure effects from one component to another. The example they use as illustration is a "Common Cabin Air Assembly" to provide life-critical air circulation in the ISS, and they show how the generated FMECA takes into account the fault-tolerance provided by redundancy in the modeled system. It does not appear that they deal with continuous physical flows in the same manner [12]. [16] reports application to design of a water recycling system (the "Cascade Distillation System") intended for use in the context of a human mission to Mars (for which high reliability and low mass are both driving concerns that make the design a challenge). The paper shows the results of their tooling to (a) "extract the FMECA from the … FSMs [Finite State Machines] defining the possible failed states" and (b) "traverse behavior diagrams to extract the fault event paths for analysis", combining these into a fault tree.

[17] and [18] outline an approach to automatic generation of FMEA and Fault Tree Analysis (FTA) artifacts from system models. Their FMEA generation process starts from a top-level functional breakdown for the system, and yields a list of generic failure modes for those functions and potential causes and effects. At this stage additional failure modes can be added manually. Having allocated components to functions, generation of a component FMEA proceeds in a similar manner. Their Fault Tree generation process utilizes the FMEAs, and hinges on graph-traversals of SysML Internal Block Diagrams (IBDs) that express component interactions and the internal structure (connectivity) of the system. They show how they convert patterns inside an IBD representing redundancy or feedback loops into the corresponding fault tree structures using AND and OR gates as appropriate. They use as illustration an electromechanical actuator used to actuate ailerons – this is described as a case study, but not an actual industrial application.

3.5 Off-nominal states and behaviors

Off-nominal behavior, of particular interest to the OSMA practitioners, has begun to be explored in the context of MBSE. Open questions remain regarding the integration of risk into the models and a methodology for managing/reducing these risks.

[6] briefly reports verification of fault management behavior by execution of a stateflow model of NASA's Ares & Orion communication during abort. Provides an illustration of a missing transition guard, presumably discovered in a simulation that exhibited an unwanted launch delay. Other examples of errors "found through modeling" are also listed.

[19] proposes, and reports on a small manually conducted feasibility study, of representing failures so as to check for safety and security properties such as "robust to any single failure", "robust to erroneous data", "resilient to fake GPS signal". They indicate the potential for translation into Alloy, a language and toolset for formal analysis of consistency.

[20] addresses the need to integrate the representation of, and reasoning about, off-nominal behavior with the standard system engineering process. They describe typical reliability activities (e.g., FMEA) and their data sources & sinks. They stress the need to model the dynamics of off-nominal behaviors, presenting a metamodel (ontology extensions) appropriate to this, including treatment of off-nominal behaviors at multiple levels of abstraction (boolean to represent working or not; qualitative to include representation of "degraded" conditions; formulae for quantitative calculations e.g., using failure rate values). [21] provides detail on semi-automated mapping from SysML into AADL for purposes of analyzing real-time aspects of the computational aspects of the system (the AADL language and associated tools provide another model-based approach to representation and reasoning for real-time software systems development; see also [22] for AADL and MBSE).

[17] presents a "safety profile" – stereotypes to extend SysML in order to represent information relevant to failures etc. ("safety profile" could equivalently be called meta-model or ontology). They offer a case study using it to represent failure information for an electromechanical actuator – specifically, an actuator of the ailerons in an aircraft. They also describe semi-automatic generation of FMEAs from this information – see the "Automated assistance for generating reliability artifacts" subsection for further discussion of this.

3.6 Support for activities post-design

Relatively few instances of the MBSE literature report on post-design application of MBSE. This may be because MBSE cannot readily be adopted midway through a system's lifecycle, so the majority of the applications (and hence the majority of the attention) has so far focused on systems in their early phases. Nevertheless, a few papers take the approach of looking ahead to how MBSE might assist those later lifecycle phases.

In [23], two of us argue for a model of system information that spans the entire lifecycle, all the way from design choices, through development and V&V, to operation. The main elements of this model are requirements, functions, components, risks, work breakdown structure, and implementation – that last encompassing operational scenarios in which the modeled system is analyzed for its operational performance. This end-to-end model allows study of the cost and schedule implications of alternative designs and alternative approaches to their V&V, and the resulting likelihoods of mission success measured in terms of probabilistic attainments of mission objectives. The key addition this paper offered to model based engineering frameworks was its representation of risk (though inclusion of off-nominal behaviors of the system's components).

[24] is a set of presentation slides that address the topic of "Managing the development of system testing using the principles of Model Based System Engineering." It makes the observation that requirements, functions and components (which we see repeated as the core concepts of many of the systems engineering ontologies) all factor into system testing.

[25] addresses the question "could system integration and verification planning benefit from the capabilities of MBSE?" and "and proposes an information model to use model-based systems engineering to actually plan integration and test activities of a system." The usual benefits attributed to model-centric representations of design information (single source of information, model-based rather than document based, representation of relationships between artifacts) are described as applicable to systems integration and verification. The paper reports having developed an information model with which to represent integration and test artifacts and activities.

[7] asserts that "V&V may be the biggest benefactor from the MBSE approach" going on to say "… by mapping requirements to architecture and defining operational scenarios as they are being developed, …, the basis for defining detailed verification descriptions, success criteria, and other verification artifacts are distribute throughout the lifecycle."

[26] considers verification of safety requirements in software-intensive systems. They point out that formal (mathematical) methods are required to rigorously prove safety properties, and discuss translation from typical SysML representations to the models and properties needed by the formal methods, for example the model checker UPPAAL (from www.uppaal.org "UPPAAL is an integrated tool environment for modeling, validation and verification of real-time systems modeled as networks of timed automata, extended with data types (bounded integers, arrays, etc."). They present an example of a mechanical press controlled by software in a programmable logic controller, considering the verification of requirements allocations from system to its components, a mix of software, mechanical and electro-mechanical.

In the software arena, [6] reported the James Webb Space Telescope's use of automated code generation of "Task distribution" and "Message distribution" code. [27] and [28] report the use of Model Based Software development to develop much of the on-board software of the LADEE spacecraft, and also to develop ground code to simulate the spacecraft. The latter was used in extensive testing of the on-board software, in operator training during mission simulations and readiness testing, and for verification of command sequences prior to their uploading to the spacecraft during the operational portion of the mission.

3.7 *Correctness of the MBSE models themselves*

[10] is a presentation advocating ways to address the question of correctness of MBSE models. It describes inspections of several of the kinds of models found in typical MBSE developments, using as illustration reference systems from grad school projects (in the naval domain). The inspections listed cover: requirements; mission and operations and interoperability; functionality, interfaces and continuity; system content flow; system behavior; system realization; allocation, and integrity; integrate-ability; qualify-ability.

4 *ACKNOWLEDGEMENTS*

This research was carried out at NASA HQ and at the Jet Propulsion Laboratory, California Institute of Technology under a contract with the National Aeronautics and Space Administration, sponsored by the NASA Office of Safety and Mission Assurance. This activity was managed locally at JPL through the Assurance and Technology Program Office.

REFERENCES

1. J.A. Estefan, "Survey of Model-Based Systems Engineering (MBSE) Methodologies, INCOSE MBSE Initiative, 2008: Retrieved 26 July 2015 from: http://www.omgsysml.org/MBSE_Methodology_Survey_RevB.pdf
2. F.J. Groen, J.W. Evans, A.J. Hall, "A Vision for Spaceflight Reliability: NASA's Objectives Based Strategy," *Proc. Ann. Reliability Maintainability Symp.*, 2015.
3. NASA/SP-2010-580, "NASA System Safety Handbook Volume 1 – System Safety Framework and Concepts for Implementation," and NASA/SP-2014-612 "NASA System Safety Handbook Volume 2: System Safety Concepts, Guidelines, and Implementation Examples" Washington, DC, 2011 and 2014 respectively.
4. H. Dezfuli, C. Everett, F. Groen, "The Evolution of System Safety at NASA," 2014, http://ntrs.nasa.gov/search.jsp?R=20140010745
5. NASA OSMA, "Reliability and Maintainability Objectives Hierarchy," https://sma.nasa.gov/docs/default-source/News-Documents/r-amp-m-hierarchy.pdf?sfvrsn=4
6. M. Aguilar, "Fault Management Using Model Based System Engineering (MBSE) Tools and Techniques," NASA Spacecraft Fault Management Workshop 2012, http://www.nasa.gov/sites/default/files/637605main_day_1-michael_aguilar.pdf
7. D.R. Wibben, F. Furfao, "Model-Based Systems Engineering approach for the development of the science processing and operations center of the NASA OSIRIS-REx asteroid sample return mission," Acta Astronautica 115 (2015) 147-159.
8. R. Karban, L. Andolfato, P. Bristow, G. Chiozzi, M. Esselborn, M. Schilling, C. Schmid, H. Sommer, M. Samparelli, "Model Based Systems Engineering for

Astronomical Projects," SPIE Astronomical Telescopes+Instrumentation. International Society for Optics and Photonics, 2014.

9. Q. Do, S. Cook, M. Lay, "An investigation of MBSE practices across the contractual boundary," CSER 2014, Redondo Beach, CA, March 2014, pp. 692-701.
10. P. Montgomery, "'Top-10' MBSE Tool Inspections to Analyze System Design Quality," Systems Engineering Conference, 2014 (SEDC2014), Washington D.C.
11. K. Vipavetz, D. Murphy, S. Infeld "Model-Based Systems Engineering Pilot Program at NASA," AIAA, 2012.
12. P. David, V. Idasiak, F. Kratz, "Improving Reliability Studies With SysML," Reliability and Maintainability Symposium, 2009.
13. R. Cressent, P. David, V. Idasiak, F. Kratz, "Dependability analysis activities merged with system engineering, a real case study feedback," Advances in Safety, Reliability and Risk Management: ESREL 2011.
14. M. Hecht, E. Dimpfl, J. Pinchak, "Automated Generation of Failure Modes and Effects Analysis from SysML Models," 2014 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). IEEE, 2014.
15. L. Wang, M. Izygon, S. Okon M. Bareh, J-F. Castet, J. Nunes, L. Fesq "MBSE Methodology for FM System Design" presentation at JPL, January 29, 2015
16. M.J. Sargusingh, M.R. Callahan, S. Okon, "Cascade Distillation System Design for Safety and Mission Assurance," 45th Int. Conf. on Environmental Systems, Bellevue, Washington, 2015.
17. F. Mhenni, J-Y. Choley, N. Nguyen, "SysML Safety Profile for Mechatronics," Mechatronics, November 2014, Tokyo, 29-34.
18. F. Mhenni, N. Nguyen, J-Y. Choley, "Automatic Fault Tree Generation From SysML System Models," IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics, 2014.
19. J. Brunel, D. Chemouil, L. Rioux, M. Bakkali, F. Vallee, "A Viewpoint-Based Approach for Formal Safety & Security Assessment of System Architectures," 11th Workshop on Model-Driven Engineering, Verification and Validation, Sep 2014, Spain. 1235, pp. 39-48.
20. R. Cressent, P. David, V. Idasiak, F. Kratz, "Designing the database for a reliability aware Model-Based System Engineering process," Reliability Engineering and System Safety 111 (2013) 171-182.
21. R. Cressent, P. David, V. Idasiak, F. Kratz, "Increasing Reliability of Embedded Systems in a SysML Centered MBSE Process: Application to LEA Project," 1st M-BED workshop, during DATE 2010, Dresden, Germany, 2010.
22. M.M. Fernández, "Using AADL to Enable MBSE for NASA Space Mission Operations," SpaceOps Conference, Pasadena, CA, 2014.
23. S.L. Cornford, M.S. Feather, J.S. Jenkins, "Intertwining Risk Insights and Design Decisions," 8th Int. Conf. on Probabilistic Safety Assessment and Management, New Orleans, 2006.
24. R. Kratzke, "MBSE for System Testing," Systems Engineering Conference, 2014 (SEDC2014), Washington D.C.
25. A. Salado, "Efficient and Effective Systems Integration and Verification Planning Using a Model-Centric Environment," INCOSE International Symposium, 2013.
26. J-F. Petin, D. Evrot, G. Morel, P. Lamy, "Combining SysML and formal models for safety requirements verification," 22nd International Conference on Software & Systems Engineering and their Applications, Paris, France, 2010.
27. K. Gundy-Burlet, "Validation and Verification of LADEE Models and Software," 51st AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, 2013.
28. N. Benz, D. Viazzo, K. Gundy-Burlet, "Multi-Purpose Spacecraft Simulator for LADEE," IEEE Aerospace Conference, 2014.

BIOGRAPHIES

John Evans, PhD
Office of Safety and Mission Assurance
NASA
300 E St SW
Washington, DC, 20546, USA

e-mail: john.w.evans@nasa.gov

John Evans earned his PhD in Materials Science and Engineering from Johns Hopkins University, and is the Program Manager for Reliability and Maintainability and Program Executive for the NASA Electronic Parts and Packaging Program at NASA HQ Office of Safety and Mission Assurance. He has authored or coauthored 3 textbooks, a book chapter on electronic materials and over 50 publications.

Steven L. Cornford, PhD
Strategic Systems Office
Jet Propulsion Laboratory, California Institute of Technology
MS 202-202
4800 Oak Grove Drive
Pasadena, CA, 91109, USA

e-mail: steven.l.cornford@jpl.nasa.gov

Steven Cornford earned his PhD in Physics from Texas A&M University, and has worked at JPL since 1992. He has performed a variety of line management and project management functions. He has been a system engineer, reliability engineer, test engineer and a DARPA Principal Investigator. He is currently particularly interested in cross-cutting problems and efforts to increase the breadth of early-phase modeling and trade space exploration. He has authored over 100 papers.

Martin S. Feather, PhD
Quality Assurance Office
Jet Propulsion Laboratory, California Institute of Technology
MS 125-233

4800 Oak Grove Drive
Pasadena, CA, 91109, USA

e-mail: martin.s.feather@jpl.nasa.gov

Martin S. Feather earned his PhD in Artificial Intelligence at the University of Edinburgh, and is a Principal in the Software Assurance and Assurance Research group at JPL, where he has been since 1995. He has authored over 140 papers in areas of risk assessment, technology infusion, automatic programming, formal specification, program evolution, runtime monitoring, verification and validation, test automation, software assurance, and information visualization