

Early Formulation Model-Centric Engineering on NASA's Europa Mission Concept Study

Todd Bayer, Seung Chung, Bjorn Cole, Brian Cooke, Frank Dekens, Chris Delp, I. Gontijo, Kari Lewis, Mehrdad Moshir, Robert Rasmussen, David Wagner

Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
818-354- 5810

Todd.J.Bayer@jpl.nasa.gov

Copyright © 2012 by the California Institute of Technology. Published and used by INCOSE with permission.

Abstract. By leveraging the existing Model-Based Systems Engineering (MBSE) infrastructure at JPL and adding a modest investment, the Europa Mission Concept Study made striking advances in mission concept capture and analysis. This effort has reaffirmed the importance of architecting and successfully harnessed the synergistic relationship of system modeling to mission architecting. It clearly demonstrated that MBSE can provide greater agility than traditional systems engineering methods. This paper will describe the successful application of MBSE in the dynamic environment of early mission formulation, the significant results produced and lessons learned in the process.

1. Introduction

The Dynamic Nature of Formulation. When the proposed Jupiter Europa Orbiter (JEO) mission became an official “pre-project” in Sep 2010, the team had already developed a strong partnership with JPL’s Integrated Model Centric Engineering (IMCE) initiative, had resolved to apply Architecting and SysML-based MBSE from the start, and had begun laying these foundations to support work in Phase A.

In March 2011 the President’s budget and NASA’s science priorities were released and together fundamentally changed the course of this formulation effort, turning it back into a study task whose objective was to propose more affordable ways to accomplish the science mission. This is a common and even typical characteristic of early mission formulation. Stakeholders, priorities and funding all can and often do change and it is essential that the formulation team be agile in responding to a changing environment.

In this case the formulation team came up with an innovative proposal to split the original JEO mission into two independent elements. Each mission element was to have approximately half of the original science and would be a standalone mission, launched independently. The splitting was done such that each element could obtain sufficient science to justify itself, so that if only one of them was built it could still be successful scientifically. Keeping the science objectives as the guiding principle, it was decided to have one of the mission elements in low orbit around Europa, carrying the science instruments appropriate to that platform. The other element naturally evolved into a Jovian orbiter with a trajectory that includes multiple low altitude flybys of Europa, having a payload complement amenable to such an observing scenario. Each of these elements by itself is expected to

be significantly less expensive than the JEO orbiter. Later on in the study, at the direction of the sponsor, a third mission element was added. The new element had the objective of delivering a science package to the surface of Europa.

A conscious choice was made to continue application of MBSE on the Europa Study, refocused for early formulation, and agility was likewise important for enabling the tailoring of the MBSE infusion effort to adapt to the evolving needs and resources. In what follows we discuss the approach taken to employing MBSE in this endeavor.

2. MBSE Approach

Architecting. Architecting is a broad term for the meandering path we take from system objectives to design. It is an exploration of concepts, addressing a typically intricate and evolving variety of concerns and aspiring eventually to converge upon an architecture that is harmonious from all points of view. Systems engineers are inclined to view the product of such endeavors narrowly as a set of requirements, each dutifully accompanied by its own isolated rationale. While this imperative, staccato style is necessary for a clear and verifiable direction to designers, it belies the complex reasoning behind a balanced architecture. Requirements alone are not ordinarily expected to address the full range of obligations and constraints under which a project works — hence the role of policies, principles, procedures, budgets, reviews, and so on. Failure to consider all the angles and instill this architectural thinking in order to preserve the relationships among concerns, concepts and design is a recurring theme in criticisms of systems engineering. Omissions, misinterpretations, inconsistencies, and poor foresight regularly lead to problems later in development or during operations. Without strong guidance from a sound architecture, attention can be drawn unwittingly to a blinkered point design, where the big picture gets lost.

Wanting to avoid such pitfalls, this mission concept study decided to cast the architecting effort in a more structured form, so that a clear line of reasoning could be followed from each stakeholder's concern to system requirements and other project directives. Specifically, the decision was made to write the rationale first, in coherent narratives tailored to each concern. Requirements, policies, and so on would then flow from this unified rationale in a manner somewhat reminiscent of the "literate programming" approach to software development. [1] The imposed architectural structure would then enable these narratives to be carefully interwoven for consistency and balance.

Such an imposition of structure immediately suggests modeling formalism. Indeed, there was strong motivation to align the architecting effort and supporting tools as closely as possible to larger interests in system level modeling in this study. After all, a structured architecture description is effectively a model, fitting neatly with the resulting requirements into the "Four Pillars" view of system modeling advocated by the Object Modeling Group in its approach to SysML. [2]

The need to launch architecting early, while the modeling effort gathered momentum, argued for a simplified approach: one that could be learned easily and adapted quickly into available tools. Thus, it was important to stick with mostly familiar systems engineering concepts, readily adaptable to comfortable forms of expression, but with just enough structure to accomplish the primary ends of the approach. Care was also necessary to ensure both that near term links to the growing system modeling activity would be straightforward, and that architecting information could migrate easily into the more involved and comprehensive system model, once it had matured. Nothing off the shelf

quite fit the bill, so a modest architecting framework (described below) was developed and subsequently adapted using an open source web development tool for collaborative databases. The tool resulting from this development is referred to as the Architecture Framework Tool (AFT).

A further innovation of interest was a more disciplined approach to the architecting effort, which sometimes can seem a bit improvised. Commitment to structure in architecting artifacts has the additional advantage of instigating similar diligence in other aspects of systems engineering, permitting for instance the productive use of workflow tools. It also encourages an iterative, incremental approach to development, establishing a regular and reviewable cadence to the effort (3-4 month cycles).

Up to the present stage in this concept study, the notions of architecture framework, collaboration tools, and staged development have worked reasonably well. An associated cultural shift at the institutional level is the more daunting hurdle, but buy-in from both project and line management can have the role of the essential catalyst.

System Modeling. As the pre-project work got under way, the team began laying the foundations of a SysML (Systems Modeling Language)-based MBSE environment to support the project team. It was clear that this would take time, so it was important to choose an achievable target availability date. The resulting infusion plan called for the MBSE environment to be in place and ready to support the project team as it entered Phase A, which was planned to be a year in the future.

Prior to this time significant groundwork had been done by JPL's institutionally-funded Integrated Model-Centric Engineering (IMCE) initiative. The IMCE Concept of Operations [3], [4] explored key scenarios and use cases for how flight projects would actually use MBSE in their execution. This work allowed the modeling team to start off in the right direction and avoid many pitfalls and blind alleys. Other IMCE funded work which benefitted this team included: deployment of a collaborative SysML tool environment (MagicDraw from 'NoMagic Inc.' and TeamWork); development of ontologies specific to our business of robotic space exploration; and incorporation of these ontologies into the SysML tools via the MagicDraw plug-in support. Finally, IMCE-funded training and support had already produced a group of several dozen competent practitioners of MBSE on which the project could draw, as well as a curriculum covering MBSE, SysML and MagicDraw which was adapted to the specific needs of the modeling team.

In order to inform the decision on how to adapt the MBSE infusion to the changing circumstances, a new Modeling Plan was developed. This emphasized a more modest, quicker, and more agile approach to application of MBSE. This revised plan won acceptance by the project and has guided the efforts and results described in the remainder of this paper.

3. Summary of Modeling Results

As mentioned before, the mission conceptual architecture description is hosted in the AFT. SysML models of all three mission concept elements have been created consisting of physical decomposition, system and subsystem block diagrams and mass reports which are automatically generated and served via web pages. The engineering team routinely interacts around the model itself, both in one-on-one conversations as well as in full team meetings. Production of the information needed by our institutional cost models now takes several days rather than the usual

several weeks. The results described below fall into three categories: Architecture Description; Concept Description; Analysis and Reporting.

Architecture Description. The architecting framework used in this study was inspired largely by IEEE 1471-2000 (ISO/IEC 42010) [5] — a reasonable model, considering the software-intensive nature of modern space developments. Balancing hardware, software and operational aspects of description in customary terms for systems engineers suggested only a few basic extensions to ensure that the full architecting story could be told. The result appears in the diagram in Fig. 1, which is described in

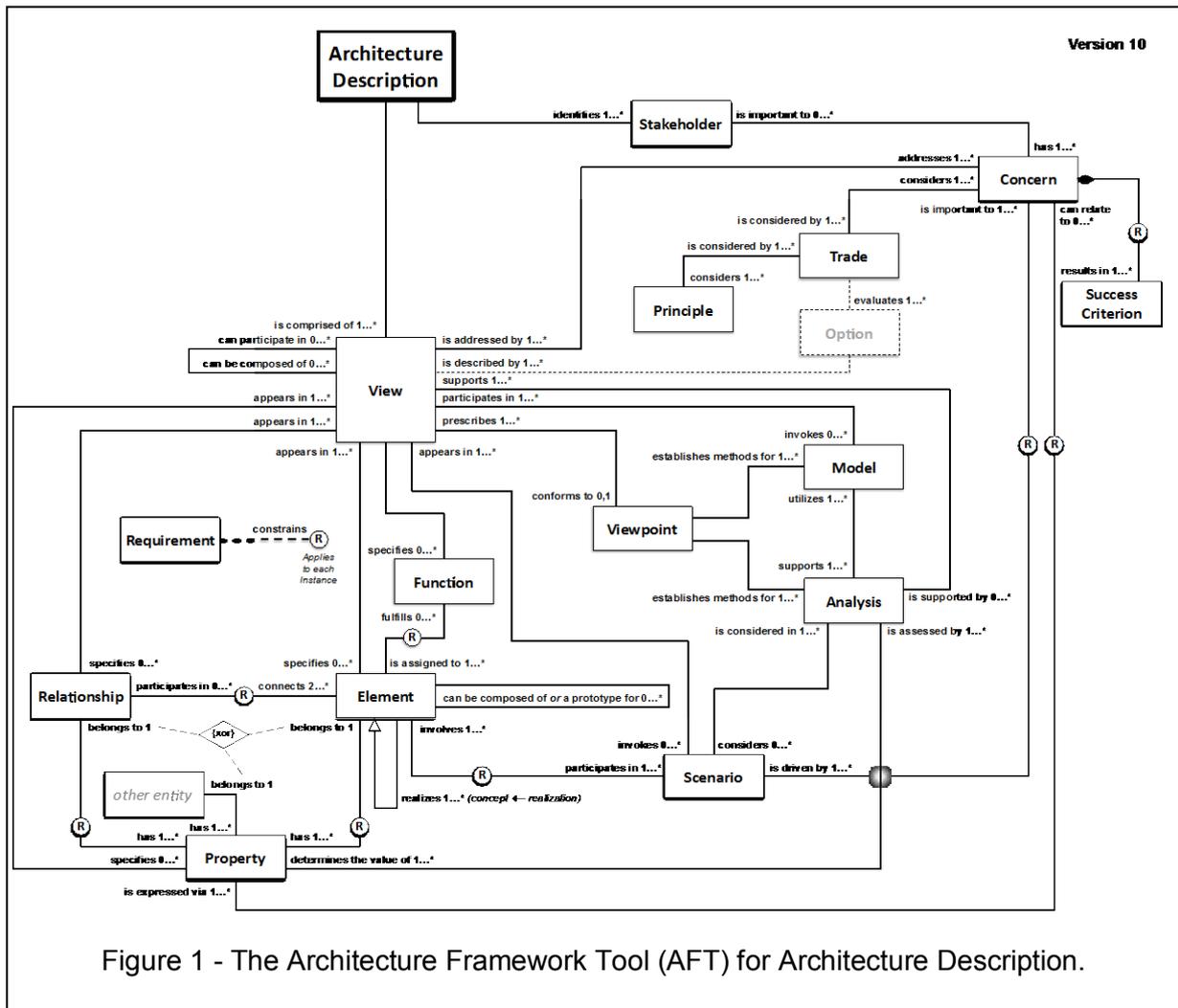


Figure 1 - The Architecture Framework Tool (AFT) for Architecture Description.

some detail below.

Blocks in the diagram define categories of items requiring exposition in the architecture description. Accompanying each category is a template (not shown) specifying the sorts of information required for each member of that category. Members are then linked in a variety of ways, according to the relationships described by connecting lines in the diagram.

Stakeholders and their Concerns are the drivers for everything else in the architecture, i.e., they can be considered the ‘entrance points’ to explore the framework in Fig. 1. Among the information required are engagement plans for Stakeholders and success criteria for Concerns. Views describe

the architecture responding to these Concerns, their form, approach, and so on. In addition, views draw together a variety of other information for consideration from a particular Viewpoint, often tailored to meet a particular Stakeholder's needs. Especially important are the Trades explored in arriving at the results reported in Views, including any Principles of note that may have been invoked in their outcome.

The system itself, plus aspects of the environment and supporting services with which it interacts, are described in assorted compositional or prototypical hierarchies of Elements. An Element is essentially any item in the decomposition of the system to which Functions or Properties can be assigned. These may be the Elements of architectural concepts, described in comparatively abstract form, or they may be the actual Elements that realize these concepts, given allocations. Interfaces or other interactions between Elements are described in Relationships. (A block diagram would thus be one way to describe a set of Elements and their Relationships.) Both technical and programmatic structures are captured in these hierarchies. By supporting a variety of conceptual and realizational hierarchies, the tyranny of dominant decomposition is avoided in favor of a far more insightful separation of concerns with exposition of the resulting relationships across domains.

Models, the Analyses performed on them, and the Scenarios that drive them round out the description. As with each of the other categories, templates of content plus associated guidelines ensure systematic coverage when they are applied methodically.

The framework above is defined in UML (Unified Modeling Language) [6] and has been integrated into the OWL ontology (Web Ontology Language) [7] guiding the larger modeling effort. The AFT has now been in use for nearly a year, validating for the most part the basic ideas behind the approach. Subsequent development will take the lessons learned from this activity to inform the eventual absorption of architecting into the overall modeling enterprise.

Flight System Deployment Description. A major part of the project architect's vision for architectural capture was the idea of product deployment. Rather than talk about subsystems, assemblies, parts, or modules, the system is described by a simple composition hierarchy. Products (e.g., propellant tanks, electronic cards, primary structure elements) are defined by their specification, with the understanding that often they are also made of other products. The connecting of atomic components to make a composite is called a deployment in this hierarchy. Many deployments are possible; for example a star tracker may be deployed to the flight system, or it may be deployed to a test bed or to an inventory for spare parts. Even the flight system is also deployed in various configurations during integration and test, which are different from the flight deployment. Some items show up earlier, while others require substitutes for certain tests. Additionally, there are different testbeds with different needs and support equipment will be a required part of many test deployments. Extender cables, breakout boxes or other non-flight variants may be needed as well. Also, there may be different deployments in flight (as in a staged system) that require separate descriptions. In all cases, variation from one deployment to the next is in the complement of components involved, the interfaces among the ones represented, and the functions assigned to that configuration. These are defined by specification, not by instance. The usual subsystem hierarchies, on the other hand, are related more to assignment of development and delivery responsibility for product instances.

This concept study developed deployment hierarchies for the flyby, orbiter, and lander elements of the flight system, but test beds and sparring have yet to be considered in the model. These deployments are presented to the engineering team primarily as traditional block diagrams, with interconnections captured in SysML Internal Block Diagrams (IBDs). These deployments allow the team to discuss the collection of electronics into a single box or separate boxes, functional vs. block redundancy and the flow of integration for example. An excerpt of a system-level deployment diagram is shown in Fig. 2.

Work Breakdown Description. Complex systems usually are described by recursive decompositions into subsystems that are then further decomposed in a top-down approach. A single hierarchy normally is used to describe all aspects of a conceptual architecture and the organizations responsible for each subsystem. This often results in oversimplifications and the failure to capture important relationships. Realistically, in many cases several inter-related hierarchies that differ significantly in structure are needed to model properly a complex system and to address different viewpoints.

Three primary types of hierarchies have been defined in this concept study, with formal models created to describe two of them at present. First, deployment hierarchies were modeled, in which composition is defined mainly in terms of modularity and integration order. The second hierarchy conforms closely to the work breakdown structure, which was decomposed into "work packages". Each product or the deployment of products is the responsibility of a particular work package. Work packages in turn are typically assigned to institutional (or business) entities, which are also organized hierarchically. The business hierarchy has not been modeled yet and is not part of this study. The important point to notice is that neither the work breakdown hierarchy nor the institutional/business hierarchy matches perfectly any of the deployment hierarchies. Defining the deployment hierarchy and the work breakdown hierarchy enables us to produce the distinct views of the system which different stakeholders require.

Analysis and Reporting Overview. Analyses such as the computation of technical margins are crucial for verifying and validating a design against the mission objectives. Technical margins measure the difference of what is predicted and what is required throughout the lifetime of the mission. Generally they include mass margin, power, energy, DeltaV and data margins. In addition, due to the harsh radiation environment of the Jovian System, radiated equipment lifetime margin becomes crucial as well. Other analyses that are fundamental to science mission design are science margin, cost estimation and sizing of the flight system. Given that the system architecture was captured as a SysML model, we moved beyond Excel wherever practical, using two main approaches. One approach used scripts that operate directly on the SysML design specifications, thus reducing error prone and time-consuming data entry and inconsistent information. When the analysis became too complex for scripts that operate within the SysML MagicDraw tool, design specifications were exported to a mathematical engine such as Wolfram Mathematica where the analysis methodology was explicitly documented and made visible. This required the use of appropriate transformations,

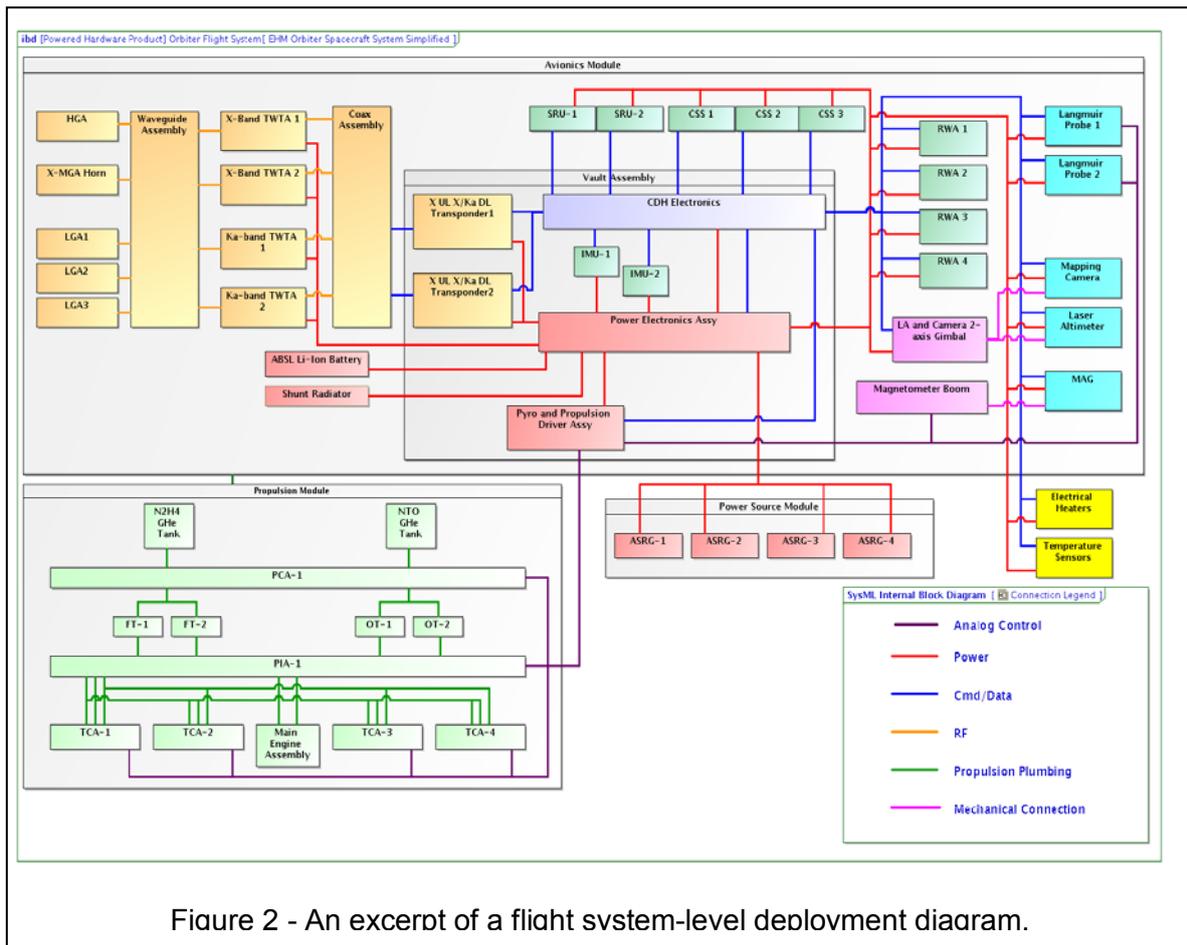


Figure 2 - An excerpt of a flight system-level deployment diagram.

as described in [8]. Documenting and reporting the analyses is as important as the analysis itself. See the discussion later on automated report generation and web publishing.

Equipment List & Mass Margin. Capturing the “Mass Equipment List” (MEL) is typically done using tools like Excel by making a list of equipment and their masses. A System Model on the other hand captures the structure composition of products (type of equipment) and their deployments. This basic object oriented structure of deployed ‘parts’ is connected to the mission domain above, providing a framework for relating a variety of information to the composite structure of the flight

system. There are several advantages to this approach; for instance, the properties that characterize the mass of the flight system are modeled complete with units and a temporal context. These mass characterizations then appear on every product in the flight system that has mass.

Mass has been treated as a pathfinder to our understanding of how to model other technical resources on the spacecraft. Traditionally, handcrafted MELs are drawn up to emphasize a hodge-podge of concerns brought to light by the systems engineer: responsibility for estimation and control of resource allocations, resource levels at critical events, “special” types of mass (e.g., radiation shielding, atmospheric entry mass, sprung mass), or items provided by external organizations. The challenge in developing MELs for Europa was to apply consistent rules to develop the mass table, meaning that ad hoc rearrangement of the MEL for “presentation” is declared out of bounds.

The use of consistent rules drove the elucidation of two major hierarchies of interest. The first is the deployment hierarchy described above. The second is the Work Breakdown Structure hierarchy, which determines what work is authorized by higher authorities and who is responsible for delivering which hardware products. The combination of these two trees ultimately resulted in three views of the MEL. The first is a simply a compositional tree of the spacecraft. The second is the Bill of Materials MEL, which organizes leaf-level components according to delivering WBS and indicates how much mass a given work package (e.g., 06.04 Power) is estimated to consume in performing its duties. The final version of the MEL is the Workpackage Assembly MEL, which is a combination of the previous two showing what the constituents of a given work package’s assemblies are, including deliveries due from other work packages.

To get totals of masses by composition or by work package requires slightly different approaches that have been implemented in the Europa team’s body of custom code. The compositional MEL rollup is a simple recursive sum of the children of a given starting node. Work-based rollups are a little more difficult. They are based on a rule that each product must be supplied by one and only one work package. This means that all copies of this product will be delivered by the same work package. In order to provide a sum of the masses supplied by a given work package, the query must work its way up the compositional tree to count how many times a given product is deployed within a given system or assembly. This counting technique is preferred because it lets the computer assure consistency in count with the deployment, rather than the alternative which is to have a separate instance of each product separately counted by the work package. The rollup traversal path is shown below in Figure 3.

This technique has provided a robust and repeatable accounting of all equipment mass in each of the spacecraft concepts. Currently the analysis of this dry mass against constraints such as total launch capability, including propellant, is still performed using Excel. This was a practicality and not a technical limitation, and is consistent with the modest and pragmatic approach necessary on the small study team.

Power Margin & Energy Balance. A flight system generally is either power or energy limited. The present mission concept is most likely power limited and as such, much of the focus has been on analyzing the power margin. The power margin compares the predicted power consumption of the flight system over the mission lifetime to the power availability. This adds a level of complexity over and above the mass analysis because power demand is variable and depends on which equipment is

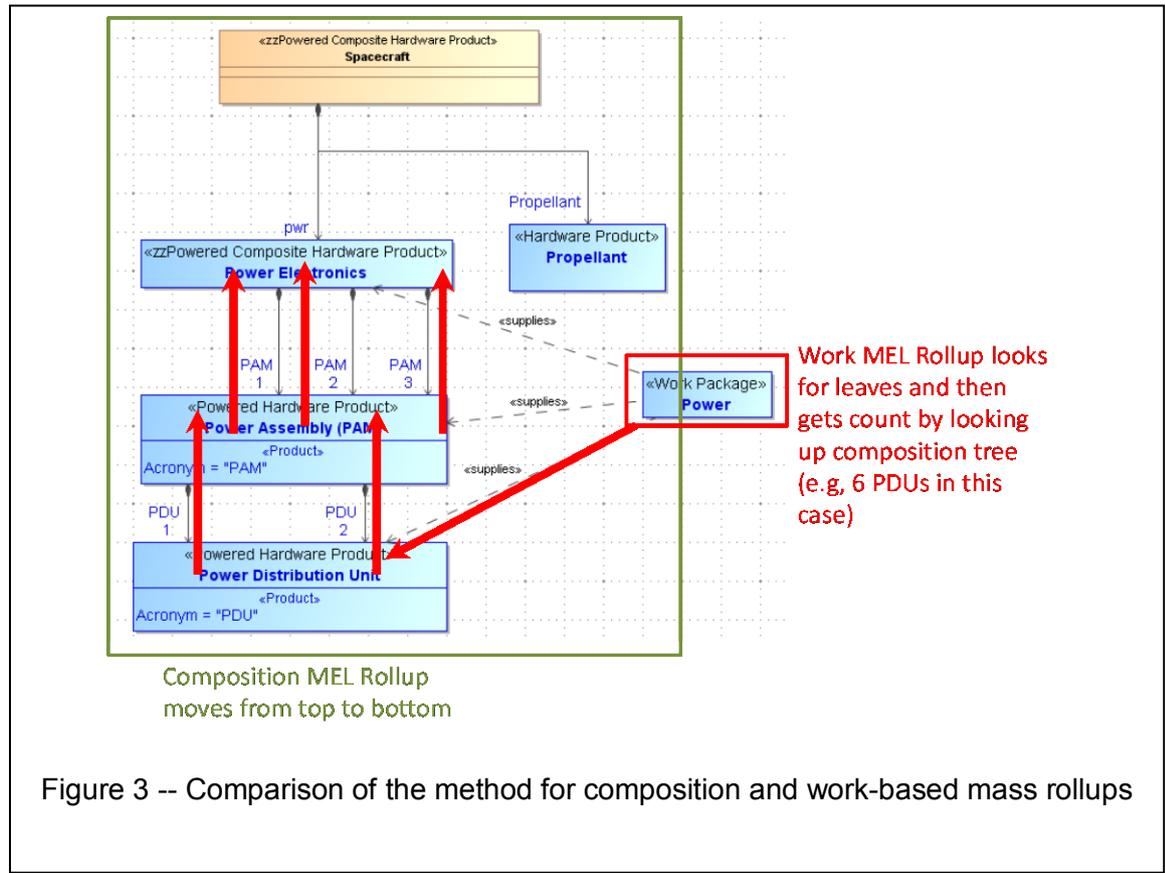


Figure 3 -- Comparison of the method for composition and work-based mass rollups

powered on, in what mode, at any particular time. Our team has not yet incorporated treatment of time in the SysML model. We therefore currently capture power consumption of each component in the model, much like mass, but use external tools to perform power usage scenarios and margin analyses.

Data Balance Margin. A key analysis in all space missions is the data balance. Remote sensing missions are fundamentally designed to send a payload of instruments to a destination where they can take measurements and return data to scientists on Earth. The data balance model considers the rate at which the system can produce data compared to its ability to move and store this data onboard and to transmit it over available radio links. It is often the case that even simple instruments can produce more data than can be returned due to the limitations of deep space communications. Therefore a model was needed to help explore the trades between observing strategies and instrument properties, transmitter power, antenna size and mass to develop an optimized mission design. Here again, because our SysML model did not yet have the notion of time, the data balance model was implemented in a self-documenting Mathematica workbook that could be exported as a report. This expanded model included relationships between orbit geometry, Earth visibility and range, mission duration, and various telecom properties, and was able to present numerous options in the form of parametric plots.

Radiated Equipment Lifetime and Margin. A critical aspect of any mission to orbit the Jovian moon Europa is radiation shielding. Previous space missions targeting high radiation environments succeeded by using very large design margins on radiation. The target of the present study is arguably one of the highest radiation dose environments in the solar system and such conservatism cannot be implemented, as it would render the mission concept unfeasible due to the large penalty incurred in radiation shielding. Therefore, more sophisticated radiation analysis to remove some of the conservatism is a topic of great interest to this study. For example, a trade option for increasing the mission lifetime may be to reduce shielding for assemblies which have long lives, while adding shielding around items whose shorter lifetimes are mission limiting. Spot shielding, vaulting and probabilistic lifetime determination need to be studied in depth, in order to reduce the shielding mass while still preserving the spacecraft life after European orbit insertion. The Radiated Equipment Lifetime and Margins (RELM) model was developed in an attempt to answer “What if...” questions and scenarios. It takes three main types of input: the Radiation environment as described by the amount of energy accumulated by typical hardware items versus shielding thickness; the radiation hardness rating of each item; and the system architecture, which describes how subsystems are assembled.

The RELM uses Excel as an interface for input and output and Wolfram Mathematica for processing. The results from the calculations are the number of months each subsystem will survive in European orbit, with a probability of 95%. These results are plotted in excel to show graphically which subsystem is lifetime-limiting. This not only helps identify subsystems that need further shielding trades, but also help in gauging the chances of survival for extended missions..

Science Margin. To assess how well an architecture or point design addresses the science concerns the classical approach has been to calculate the system capability to meet a specific quantitative requirement, such as spatial resolution for example (measured in arc-seconds or meters). This classical approach does not take into account the synergistic interactions that lead to addressing a science concern. Using the classical approach it is not a straightforward matter to make quantifiably defensible trade arguments, but often hidden margins come to the rescue. On a recent astrophysics mission, a concept has been developed that gives quantification of changes in science return vs changes in technical design.[9],[10] These lessons learned have led to the development of a “Science Margin Model” (SMM) for this mission concept. This enables the stakeholders to assess the impact of technical trades on the science return, with the objective of keeping the Science Margin at a healthy level.

For this study two of the broad science objectives, determining Europa’s magnetic induction response and the amplitude/phase of tides, were cast into two SMMs; this was accomplished by working in concert with the science domain experts.

Cost Estimation and Integration with Cost Models. In early formulation, most cost models are parametric, and by far the most important parameter is mass. By capturing the concept description in the SysML model and automating the reporting of mass, the time to produce inputs for a cost estimation run has shrunk from a couple weeks to several days. This is viewed as a highly effective provisional “integration” of system model with cost models.

Automated Report Generation and Web Publishing. The ability to separate the concern of how information about a mission and flight system is captured and the concern of how it is viewed is a

key benefit of Model Based Systems Engineering (MBSE). The capability to produce and manage reports from the model is part of realizing this benefit. On the other hand, using the model-authoring tool to access information in the model is not always desirable, particularly when producing and reviewing deliverable products such as architecture descriptions and requirements documents. For this study concept the focus was to produce a report similar to the system block diagrams and mass roll-ups produced by conventional office productivity tools. The product deployment model contained the Mass Equipment List (MEL) and mass estimate information. The semantic rigor of the model facilitates analysis and generation of tables describing this information. These tables are explicitly consistent with the views in the report that depict the flight system block diagrams. The table view provides a concise way to read the MEL that conventional office tools are not capable of. Additionally, generated documents have the benefit of making much more powerful use of Internet technology. The project now provides a website where the latest reports are accessible and configuration controlled.

4. Lessons Learned

In our successful application of MBSE to the Europa Studies, we identified certain things that worked and others that didn't. The following 15 "Lessons Learned" are a distillation of this experience that we hope will be helpful to others attempting first-time MBSE applications.

Investment is Crucial. When EHM embarked on its scaled-back modeling efforts in May 2011, it had the benefit of several years' worth of investments in foundational infrastructure by IMCE. It also had the benefit of almost a years' worth of application investment by the JEO Pre-Project. A SysML tool was selected and deployed at JPL (MagicDraw). A project collaborative modeling environment was established. MagicDraw customization was done enough to be useful. The Architecture Framework Tool was mature. And finally, SysML/MagicDraw training had been given to the Europa team.

Unity of Leadership is Essential. In the first infusions, management support for the effort must be clear and consistent. Management must be willing to pay the startup costs and to give time for the effort to pay dividends. In addition, the engineering leadership must be reasonably unified in their willingness to work together to figure out how to do this. Both of these things were true on JEO and were essential to finding success even after the transition to the smaller, leaner EHM phase.

Early Efforts Draw on a Limited Pool of Talent. Similarly to the "Unity of Leadership" above, the first infusions will not have the benefit of an engineering pool with ubiquitous modeling skills. On JEO we found that the best way to get started on the right path was simply to hire as many of the existing cadre of skilled MBSE practitioners as we could afford. At that time there were no more than ten or so truly expert system level modelers versed in SysML, and we hired nearly all of them.

Leverage Learning with Synergistic Work. With the limited pool of modeling talent available, we were tempted to ask for a full-time commitment. But we knew there were two other efforts where MBSE application was being tried and that these efforts would have a strong desire for the same personnel. We also believed that having the experts engaged in two or three modeling efforts would provide benefits that outweighed the lack of full time commitment. We have found this belief to be fully validated: the learning that has been shared between the three efforts has been enormously beneficial for all, and has clearly accelerated the institutional infusion.

Innovation is Bottoms-Up. We didn't know what scripts or plugins or modeling patterns to develop before we started. We let the discovery of the need drive the solution. There was 'top down' innovation but not in the traditional sense of pre-ordained specifications: it consisted mainly of constant guidance during the modeling process to keep the effort focused on satisfying the end objectives.

Team Organization Matters. It is important to consider carefully the use cases for how modeling is done. Before JEO, most of the MBSE pilots that had been tried at JPL were small scale and often grass-roots. Systems Engineers on these small teams tended to become the primary modelers and therefore tended to be the primary custodian of the single source of authoritative information as well as being the most expert SysML users. The IMCE Concept of Operations looked at what a large scale flight project would need to do differently. JEO, as a fledgling project and as the first full, top-down infusion of MBSE at JPL, had the benefit of that work as we worked out what the model usage patterns should be. The pattern we have found that works well is a three-tiered one involving a small set of core modelers within a larger set of modeling-savvy systems engineers, within a larger set of all project personnel. While we have found that descriptive modeling can be done by almost anyone with the basic training, the additional rigor and consistency needed for quantitative analysis requires us to designate a smaller team of people who are modeling experts and who can apply best practice to the official configuration managed project models. Presently we have a core modeling team of a half dozen or so, within a larger team of 20 or so engineers.

The experienced systems engineers provided guidance to keep the modeling focused on providing useful information, as well as mentoring of the core modelers who tended to be more junior. Frequent (daily) interactions were crucial to getting useful products: we were pathfinding so we had to stay very closely in touch.

As important as it is to have a core modeling team, it is just as important to avoid fencing them off from the rest of the project. If the models are to be useful to the project, the project must understand and interact with the modeling team regularly, and likewise the modelers must be engaged in the larger engineering effort. Modelers who are also capable systems engineers will naturally employ their modeling skills to deliver engineering products in a model based way.

Everyone Needs Training, but to Different Levels. In the usage model above it is clear that all three groups need to receive training commensurate with their level of interaction with the models. Different levels of modeling familiarity are required, thus resulting in different levels of training. Working with IMCE, we have constructed a set of classes that addresses all three user-type groups. The classes are sequential, each one building on the one before it. We start with the basics: Architecting, AFT, MBSE and SysML familiarization for everyone. The second class is the advanced SysML and Tool-Specific training, for the engineers who will be working with the models (MagicDraw and AFT). Then finally the third class, really a continuing series of special topic sessions, is for those engineers who actually construct, maintain, and analyze the official project models.

Just Do It. We've found that the best way to figure out how to apply MBSE is to do it for real: make the commitment to adopt MBSE as the way to produce (at least some subset of) the project products, and then figure out how to accomplish this. This is in contrast to the suggestion sometimes made by skeptics, that a "safer" or "more gradual" approach would be to conduct a "shadow" or "parallel" pilot that allows side-by-side comparison of benefits and drawbacks, including cost.

So how does a project control infusion cost and risk without this comparative knowledge? Do it by carefully scoping the infusion: start small, but always start on a real product. These ideas are further discussed below.

CM Can Start Modestly. In thinking about the needs of a Configuration Management (CM) system for our models, we found that the Initial exploration in the IMCE Concept of Operations was helpful. Initially setting up the model to support collaboration, we focused on: structuring modules and packages with collaboration in mind; and we emphasized single owner packages in topically-defined modules. Model access permissions were set loosely for the time being. Lightweight versioning was found to be sufficient: Teamwork was used to track changes to model elements; DocWeb reports captured snapshot of full model and resource reports; reviewed and baselined versions are tagged as such in DocWeb. Quality Control is developing as needed: scripts are now doing some rudimentary model validation; a hand calculation is used before report release as final correctness check.

Models are Meant to be Abstractions. A common misconception of MBSE is that in order for the model to be useful it must describe everything, and describe it to a fine level of detail. This misconception needs to be corrected for an infusion to succeed, because otherwise resources will run out long before the job is complete. A key principle we have followed is to model only as far as we need to answer the question at hand. Assuming this is done on an infrastructure of common languages and tools, then the model can grow over time, as necessary, and each new model element will add synergistically to the body of work.

In the early formulation phase in which we find ourselves there is a curious duality. On the one hand, the key work in early formulation centers around conceptual thinking. The spacecraft we propose are mere sketches, and a critical function of models is to describe the design space generously, in which the concept can evolve and take shape. On the other hand, we must always show that our concepts are feasible, and one of the ways we do this is build and analyze a 'point design' which we analyze for technical resources, performance, and cost. The models that we build to address these two disparate viewpoints must of necessity be partly conceptual and partly realizational. This should not mislead one into thinking that the space between them must be entirely filled in: it has proven very workable to have some parts of our model be treated as strictly conceptual, and other parts be treated as realizational (e.g., for the mass margin analysis).

Models Evolve. The model needed in concept formulation is very different than the model needed in detailed design, or in operations. Models need to evolve and grow, and sometimes shrink. This should be the focus of model reuse along the project lifecycle. It also helps to answer the people who will suggest that building a detailed model of the last flown mission will help you formulate the next. It all goes back the principle of modeling for a purpose, and not more. While the models may change, these changes can be evolutionary and cumulative as long as they are connected by a common set of ontologies and methodologies.

First Description, Then Analysis. Another common misconception is that models are not really useful until they can be subjected to quantitative analysis. This is simply not the case. Capture and description are powerful and far-reaching first steps. Just describing something in a formal modeling language like SysML immediately improves communications and understanding. The benefits of this would be difficult to overstate.

Additionally, model capture is relatively easy, especially compared to analysis. For the mass margin analysis, even our modest ambitions were a bit of a stretch the first time because of all the capabilities that needed to be created along the way. But if the first mission element took longer than expected to analyze for mass margin, the second and third ones showed the power of developing reusable methods: they each took a fraction of the time of the first. For the first mission element, the EHM Orbiter, both model capture and analysis were performed in the SysML model and the mass report took approximately two work-months to complete. The subsequent two concepts we modeled (Flyby and Lander) each took about one half work-month each. And, a subsequent change of Flyby design was accomplished in a fraction of *that* time.

So, our advice is to first focus on description, and then implement analyses. A large part of the benefit accrues as soon as people start using the descriptive models, and this gives time and support to allow the more difficult work of analysis to be done.

Separate the Model from the Analysis. Looking at Excel, the traditional tool for systems mass analysis, is instructional for understanding why we need better systems modeling tools. Two troublesome characteristics worth mentioning in this context are: as it is commonly used, the model and the analysis are inextricably intertwined; and by the nature of the tool, the model is forced into a form which facilitates the analysis.

It is clear that the more a model can be a self-contained, internally self-consistent, and an intuitive description of the concept, the more informative it will be. Moreover, the more the analysis can be separated from the model, the more reusable it will be. For our mass analysis we have achieved a high degree of separation of the model from the analysis, and as a result we are able to run exactly the same mass analysis script on all three of our mission option models.

The corollary to this is “keep the model aligned with the concept rather than with the analysis”. We initially found ourselves adopting modeling patterns which made the analysis scripts easier (drifting back into the Excel mindset). But we soon discovered that in order to further expand and refine these analyses, we would be forced to model in more and more non-intuitive ways. Therefore we discovered, and adopted, the principle that the model should be kept intuitive and aligned with the concept. We are convinced that the extra work required to make the analysis tools work is well worth it in the long run.

Keep the Focus on Engineering Products. Keeping focused on real engineering deliverables is important to avoid the pitfall of delivering a modeling solution everyone thinks is finished but which doesn't provide the required engineering answers. After all, the engineering deliverables are the whole point of the exercise. Our early attempts at “rolling up the mass” for the mass margin report showed this in stark relief: getting the numbers to add up, make sense, and be reliable turned out to require significantly more modeling and scripting than expected.

Real Examples are Powerful. Trying to describe to stakeholders and potential collaborators what MBSE looks and feels like has proven to be rather difficult and not very effective. We have found that many people ‘get it’ for the first time only when they see an actual example.

The EHM Orbiter Mass model and Margin Report were immediately recognized as higher fidelity work than reports generated by traditional methods. Since parametric cost estimates are based

heavily on mass, this is a crucial parameter to estimate accurately. This was also the feature that helped the 'light go on' for several skeptical but open-minded stakeholders.

Finally, projects are where the 'just do it' happens, working on actual products - that's where the applications are really worked out, and that is what feeds back into IMCE for others to use. These first examples discover useful patterns which can be fed back into IMCE for capture, stewardship, and provision to the next users.

5. Future Work

The Europa Study will deliver a Final Report to our sponsor in May, 2012. Further direction may include developing one of the concepts into an actual mission. If so, we will solidify the MBSE gains already made and take the next steps in development of our model-based approach. These next steps would include: completing implementation of the full mass margin analysis in the SysML environment; and incorporating time into the SysML model to enable power, energy, data balance and other time-based analyses in the model. In addition we would develop scenarios, functional decompositions, interfaces, and requirements. Finally, we would complete the enhancement of the Architecture Framework Tool to allow creation of flexibly structured and formatted reports, so that the AFT can be the single source of truth for the full narrative of the architecture description.

6. Conclusion

Our use of system modeling has contributed directly to the recognized high quality of our mission concept, including: increased stability of the concept description, increased accuracy of the key technical resource estimates such as mass and power, and increased agility in the face of changing sponsor needs and priorities. If these studies result in an approved mission, then that team will have much more useful information to draw on than the traditional study would have provided: our use of MBSE will pass to the project team a much more durable, rich and extensible body of information from which to start. Finally, through institutional stewardship, our progress will serve as a positive example and provide a powerful springboard to the next projects adopting MBSE.

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

7. References

- [1] D.E.Knuth, "Literate Programming." The Computer Journal, NO27(2) May:97-111. (1984)
- [2] <http://www.omg.sysml.org/>
- [3] "An Operations Concept for Integrated Model-Centric Engineering at JPL", Bayer, Cooney, Delp, Dutenhoffer, Gostelow, Ingham, Jenkins, Smith, 2010, IEEEAC Paper #1120.
- [4] "Update: Concept of Operations for Integrated Model-Centric Engineering at JPL", Bayer, Bennett, Delp, Dvorak, Jenkins, Mandutianu, 2011, IEEEAC paper#1122.
- [5] <http://www.iso-architecture.org/ieee-1471/>
- [6] <http://www.uml.org/>

[7] <http://www.w3.org/TR/owl-ref/>

[8] B. Cole and S. H. Cheung, "Getting a Cohesive Answer from a Common Start: Scalable Multidisciplinary Analysis through Transformation of a System Model", 2012, IEEEAC paper #1327,

[9] R. Duren, Aerospace Conference, 2006 IEEE

[10] M. Moshir, et al., Proc. SPIE 7734, 2010

8. Biographies

Todd J. Bayer is a Principal Engineer in the Systems and Software Division at the Jet Propulsion Laboratory. He is currently the Flight System Engineer for the Europa Habitability Mission Studies. He received his B.S. in Physics in 1984 from the Massachusetts Institute of Technology. He started his career as a project officer in the US Air Force at Space Division in El Segundo, California. After joining JPL in 1989, he has participated in the development and operations of several missions including Mars Observer, Cassini, Deep Space 1, and Mars Reconnaissance Orbiter, for which he was the Flight System Engineer for development and Chief Engineer during flight operations. During a leave of absence from JPL, he worked as a systems engineer on the European next generation weather satellite at EUMETSAT in Darmstadt, Germany.

Seung Chung is a systems engineer in the System Architectures and Behaviors Group at the Jet Propulsion Laboratory. His research is focused on model-based autonomy architecture, model-based estimation and diagnosis, and fault-tolerant planning and execution. His expertise is in the formal methods of Artificial Intelligence, including satisfiability, constraint satisfaction, planning, symbolic model-checking and probabilistic analyses and methods. He is currently working on projects in which he participates in architecting and designing both the ground and flight systems using model-based approaches. He received his Ph.D. in Autonomy and S.M. in Aeronautics and Astronautics from the Massachusetts Institute of Technology and his B.S. in Aeronautics and Astronautics from the University of Washington.

Brian Cooke received a B.S. in Engineering Science and Mechanics from Virginia Tech in 1995. He has been with JPL for more than 15 years and is the recipient of three NASA Exceptional Achievement Medals. He is currently the Project System Engineer for the Europa Habitability Mission helping to develop and plan NASA's exploration of this intriguing Jovian moon. He has previously served as the Kepler Project System Engineer, Dawn Project V&V Engineer and the GALEX Instrument I&T Manager.

Bjorn Cole is a systems engineer in the Mission Systems Concepts section of the Jet Propulsion Laboratory. His research interests are in the fields of design space exploration, visualization, multidisciplinary analysis and optimization, concept formulation, architectural design methods, technology planning, and more recently, model-based systems engineering. His most recent body of work concerns the infusion of systems modeling as a data structure into multidisciplinary analysis and architectural characterization. He earned his Ph.D. and M.S. degrees in Aerospace Engineering at the Georgia Institute of Technology and his B.S. in Aeronautics and Astronautics at the University of Washington.

Frank Dekens is a Systems Engineer in the Instruments and Science Data Systems division at Jet Propulsion Laboratory. He is a member of the project systems engineering team on the

Europa Habitability Mission. Prior to this, he was the Instrument Systems Engineer for the Space Interferometry Mission. Frank received his BS and Ph.D. in physics, both from the University of California, Irvine.

Christopher Delp is the Systems Architect for Ops Revitalization task in MGSS. He is also a member of the Flight Software Systems Engineering and Architectures Group at the Jet Propulsion Laboratory. His interests include software and systems architecture, applications of model-based systems engineering, and real-time embedded software engineering. He earned his M.S. and B.S. degrees from the U of A in Systems Engineering.

Ivair Gontijo is a member of the project systems engineering teams on the Europa Habitability Mission and the GRACE-FO projects. Prior to this he lead the design and manufacturing of the RF packaging for the radar that will control the final descent of the NASA Mars Science Laboratory on Mars. His area of technical expertise is optoelectronics and RF devices. He has a BSc in Physics and an MSc in Optics from UFMG in Brazil and a PhD in Electrical Engineering from Glasgow University, UK.

Kari Lewis has been a flight system engineer at JPL since 1996. She joined JPL after graduating with her BS in Aerospace Engineering from the University of Texas at Austin. She also received her MBA from the Anderson School of Business at the University of California, Los Angeles in 2004. Ms. Lewis has worked several missions in her career, including Deep Space 2, Mars Reconnaissance Orbiter, Mars Science Laboratory, and recently JEO. Her current position is project system engineer for the JPL contributions to the Jason-3 mission.

Mehrdad Moshir is a Systems Engineer in the Systems Engineering Section at the Jet Propulsion Laboratory. He is currently the Flight Systems Engineer for the Surface Water and Ocean Topography (SWOT) Mission. He was previously the performance modeling manager for the Kepler and SIM/SIM-Lite missions and incorporated many Model-Based concepts into this work and evolved the approaches for developing Science Margin Model. He holds an A.B. in physics from UC Berkeley as well as M.A. and Ph.D. in physics from Princeton University. He began constructing particle detectors and electronics for high energy physics experiments at Brookhaven National Laboratory while in graduate school and over time his interests migrated towards systems engineering of space missions.

Robert Rasmussen is an Engineering Fellow in the Systems and Software Division at JPL, which he joined in 1975 with a Ph.D. in Electrical Engineering from Iowa State University. He has since supported several flight projects in both technical and management roles. These have included G&C Technical Group Supervisor during Galileo development, and Subsystem Cognizant Engineer for the Cassini Attitude and Articulation Control Subsystem. In addition, he has led various research efforts in fault-tolerant parallel computing, spacecraft automation and autonomy, and model-based software architecture. Bob has been both Chief Technologist and Chief Engineer for JPL Technical Divisions. He is presently the Architect for Europa mission studies, where he is helping to introduce and refine improved architecting and model-centric engineering methods.

David Wagner is a software system engineer and architect in the Flight Software Applications and Data Management group at JPL and was a principal developer of the Mission Data System in 2000-2006. Since then he has continued to apply MDS technology and State Analysis in several applications. He is currently a member of the project system engineering team on the Europa Habitability Mission formulation project. He has a BS in Aerospace Engineering from the University of Cincinnati, and MS in Aerospace Engineering from the University of Southern California.