

FINAN
03-2751

Model-Based Engineering Design for Space Missions^{1,2}

Stephen D. Wall
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109
818-354-7424
steve.wall@jpl.nasa.gov

Abstract—The basic elements of model-based design for space missions have existed for almost a decade, awaiting an opportunity to implement them in the same place at the same time. In early design phases, combinations of models, concurrent engineering methods, and scenario-driven design have been used for several years with results that have exceeded even optimistic expectations; but the goal of extending these methods to later phases of design has been more elusive.

JPL's Model-Based Engineering Design (MBED) initiative will provide opportunity to reach that goal. It enables advanced systems engineering practice through a series of integrated, increasingly detailed models that provide continuity from architectural concept through detailed design. It extends current capability for rapid conceptual design, allowing thorough exploration of design tradespaces and selection of an optimal design point with associated cost and rationale; and it provides seamless connection to subsystem models and detailed design tool suites.

In this paper we will review the goals and status of MBED and show the expected interconnectivity between conceptual and detailed design.

TABLE OF CONTENTS

- 1. INTRODUCTION 1
- 2. PHASES OF DESIGN 2
- 3. PRINCIPLES OF MODEL-BASED DESIGN ... 2
- 4. MODELS AND DESIGN PHASES 3
- 5. SUMMARY AND CONCLUSIONS 7
- 6. ACKNOWLEDGEMENTS 8
- REFERENCES 8
- BIOGRAPHY 9

1. INTRODUCTION

For very close to a decade, engineering design practices have experienced increasing pressure to change—pressure that originates both from the availability of new design capabilities such as tools, infrastructure, teaming practices, and the like, and from sponsor and management demands for increased productivity (e.g., [1]). The world of space mission design has sometimes led, but more often followed, that trend. Advances in information technology and related fields have created tools that free designers from many of the more pedestrian elements of the job, such as data exchange challenges (data entry, file exchange, etc.) [2], standards development [3], and communications challenges (phone tag, meetings).

Similar process-related changes have also been developed. Among the notable areas are concurrent engineering [4], Integrated Product and Process Development (IPPD), Integrated Product Teams (IPTs), Smart Product Models [5], and others (e.g., [6]). Human factors issues have been addressed as well [7,8]. Many of these new capabilities have been well received by engineers and lower-level managers, and some results have approached order-of-magnitude increases in productivity. Sponsors and upper management have also applied pressure to improve efficiency through process changes. Both designers and their managers generally respond positively when the net effect is to increase percentage of time spent in the creative aspects of the task, but less so when there is uncertainty in the outcome. Managers seem especially wary of changes that might add risk, or the perception of risk, to their program or that might not produce results as quickly or as well as existing practice.

JPL's Model-Based Engineering Design (MBED) initiative grew from beliefs (1) that there existed a need for change in the engineering design process, (2) that early modelling of design seemed to offer a solution to the need, and (3) that pieces of that changed process were available but not well-enough connected together to be useful. MBED represents a commitment on the part of JPL management and designers

¹ 0-7803-8155-6/04/\$17.00 2004 IEEE
² IEEEAC paper #1005, Version 1, Updated 2 December, 2003

to assemble those pieces into a coherent whole. In this paper we will first briefly review the space mission design process at a high level; we will next introduce the types of models that MBED brings to the process and show how the models will work together to reform the design process and integrate it into the implementation phase.

2. PHASES OF DESIGN

Space missions are traditionally divided into four phases: conceptual design, formulation, implementation, and operations. The conceptual phase is dedicated to transforming an idea into a feasible design and a total cost accurate to perhaps $\pm 30\%$ (generally expressed as margin), usually for the purpose of assessing its marketability to some potential sponsor. The traditional goals of this phase are (1) to determine if a feasible design exists; (2) to state the requirements on the development of the design and to develop at least a preliminary balance among them; (3) to estimate the total cost; and (4) to establish a realistic schedule by which the project could be executed.

Designs that are accepted pass into a second phase called formulation. The objective of this phase is to develop a buildable design based on requirements generated in the conceptual phase. In principle, high-level requirements should be well developed and fairly stable in the formulation phase; in practice this is often not the case, for three reasons. First, deeper consideration of design almost invariably uncovers issues, some of which involve reconsideration at the system or even requirements level. Second, sponsor requirements, even at the highest level, frequently continue to change due to funds availability, changes in overall objective, or both. Third, science (or other user) requirements may change, due to maturation of the investigations, results from other missions, or changes in the target itself. Thus, keeping the evolving design and the evolving user needs in balance with each other becomes a significant challenge in this phase.

The implementation phase involves fabrication, purchase, integration, and test of the hardware and software necessary to accomplish the mission—both the flight and the ground segments. Logically, this phase would involve no design, but all three classes of exceptions noted above continue to exist in most cases, and design and rework are not uncommon.

Finally, the flight operations phase begins with launch of the spacecraft and continues through data collection, downlink, processing, and early science data analysis. Even here, however, design can continue, as almost all spacecraft can be modified by command and/or onboard software updates. Mission alterations can still be required as a result of either discovered design flaws, sponsor initiative, or changes in science requirements. As in formulation, keeping user requirements and design capabilities consistent with each other is important in both implementation and operations phases.

The traditional space mission design practice has produced many highly successful missions. However, the lack of consistency between design and user scenario has been a concern in several major missions and has caused significant cost overruns in others. The traditional process does not allow for validation of design until the design has been largely completed. It does not provide any forward indicator of success or failure, leaving designers and implementors to discover many design errors during subsystem or system test. Not only is this practice extremely costly because of the rework it implies, it is also complicated by the confusion of design errors with fabrication errors. The ability to validate designs against user scenarios at all phases of design, especially in early phases, would give space missions welcome insurance against both schedule and cost slips.

3. PRINCIPLES OF MODEL-BASED DESIGN

A concept dubbed model-based, or model-driven, design was popularized by L. Baker and others in 1997 [9], although suggestions for use of performance and other models to capture design were described previous to that date [e.g., 10,11]. The principles of model-based design are (1) that designs are captured in a model environment, in contrast to the more traditional text and drawing capture methods; and (2) that these models are used in simulations demonstrating behavior similar to that expected of the final product, and approaching exactly that of the final product as the design matures. Model-driven design promised to allow early detection of design errors through the use of early and broad models, first as “executable specifications,” which would replace text-based requirements, and later as ever-increasingly detailed models. Simulation of design behavior promised early detection of design errors and potential savings of time and money.

Realization of model-driven design has, however, been slow. In previous works we have proposed how various kinds of models might be applied at various phases of design [12] and described pilots testing use of those models in mock design situations to gauge the acceptability of the new practice within the aerospace culture. Until now, however, we know of no large-scale, end-to-end implementation of model-based design in a real mission environment, and we know of no metrics that demonstrate that the promises of model-based design are or are not achievable. The reasons for this slow acceptance are not completely clear. The most often quoted reasons are that the engineering culture is slow to change, and management culture is slow to accept the risk of doing something new. In addition, models proposed for mission use have sometimes been poorly matched in level of fidelity and have thus been unable to keep up with designers in early phases as the design evolves. Until recently, complications of data connectivity among commercially-built modelling and design tools has inhibited progress, as has the difficulty in parameter-level access to commercial product data management systems. Finally, in the era of “faster, better, cheaper” missions, validation has sometimes

been neglected in the name of cost savings, and to compound matters, mission failures arguably resulting from this neglect have led to a retreat to more traditional design methods.

The most practical objective of model-based design is to achieve a connection between design (represented by the designers) and scenario (represented by the users) throughout the mission. System tests uncover design errors and requirements incompatibilities early in the process. In a previous study [13], we tested the idea that requirements themselves can be captured in a model, the basic architecture of which is shown in Figure 1. Specifically, we tested whether designs could be meaningfully captured in models at varying levels of fidelity appropriate to different design phases, and whether science or other user scenarios could be used to drive these models as performance simulations at those levels, thus providing two early tests of the system not previously available. For this study (and in subsequent work described here), scenarios were captured in the JPL program APGEN [14], which is used to develop and manipulate activity plans at several different levels for many NASA spacecraft. Parameters describing the system were managed in a database that also serves as a link to later models [2]. These lower-level, subsystem-fidelity, models were in turn used as links to design tools (e.g., CAD or ECAD tools) that produce detailed design leading to actual hardware and software. The nature of these models will be described more fully in the following section.

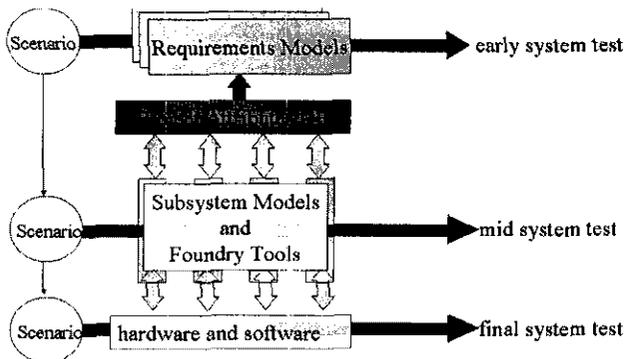


Figure 1 - Basic Architecture for Model-Based Design
(after [13])

To extend the previous work and take a major step towards establishing model-based design as accepted process, JPL's Research and Technology Development Program has now begun MBED, a multiyear strategic initiative starting in fiscal year 2004. MBED will build an environment, infrastructure, model library, and laboratory for integrated, model-based space mission system design. A set of integrated, increasingly detailed models will provide continuity from architectural concept through detailed design, enable rapid design tradeoffs, support quick generation and evaluation of new mission concepts, and encourage thorough exploration of design tradespaces. The ultimate goals of MBED are (1) to facilitate intelligent selection of an optimal design point from a fully-explored

tradespace with associated cost and rationale; and (2) to enable implementation of that design within cost and schedule, maintaining the original (or as-modified) performance and achieving the required science goals. Should system or science requirements change, the MBED system is anticipated to have the agility to respond to those changes in a timely and consistent manner by creating a new balance among cost, schedule, and performance.

4. MODELS AND DESIGN PHASES

Words like “model” and “simulation” have been used in so many contexts with differing intent that it is helpful to define them precisely. A definitive taxonomy of modeling is beyond our scope, but in this paper “model” will be used as a noun referring to a device or software entity that is or represents an abstraction of some thing (in the present case the thing being designed), to some level of fidelity that offers an advantage over the use of the thing itself. A behavioral or functional model may be easier or cheaper to construct than the real thing, for example, and it may offer the advantage that it allows evaluation of the thing's behavior without spending the time or effort to build it. The “fidelity” of a model may be measured by the number of parameters representing the required functions and properties and their required uncertainties (e. g., mass, power, length, attitude control uncertainty). An implementation-phase model of a spacecraft that approaches the fidelity of the spacecraft in every detail would include every dimension, every physical or nonphysical measure, and might involve 10^5 or more parameters. A conceptual-phase model of the same spacecraft might begin with as few as four parameters—its mass, power, data rate, and cost. “To simulate” is a verb referring to the driving of a model with some intended path or scenario, for example to elicit an approximation of the behavior of a thing in the scenario or when traveling the path. A “scenario” is a time-ordered collection of events or actions, and is comparable to some uses of the word “conops,” a contraction of the phrase “concept of operations.”

At the conceptual phase of design, MBED will expand to considerable advantage what is known as the Team X process [4]. In Team X, a four-parameter design can be grown to a design of approximately 10^3 parameters in a few weeks using concurrent engineering techniques and some simple data exchange mechanisms. MBED will develop augmentations to Team X to enable more complete exploration of the available tradespace. A hyperspace filled with these points will then be shown to a design team by adapting a novel visualization method [15] shown in Figure 2. A collection of techniques will allow the team to explore this space, identifying points of interest and comparing their attributes. Any parameter of interest can be assigned to any of three axes, or to color, intensity, or symbol size. Preliminary trials with design teams indicate that, with experience, designers can use the tool to analyze up to seven dimensions of a tradespace simultaneously.

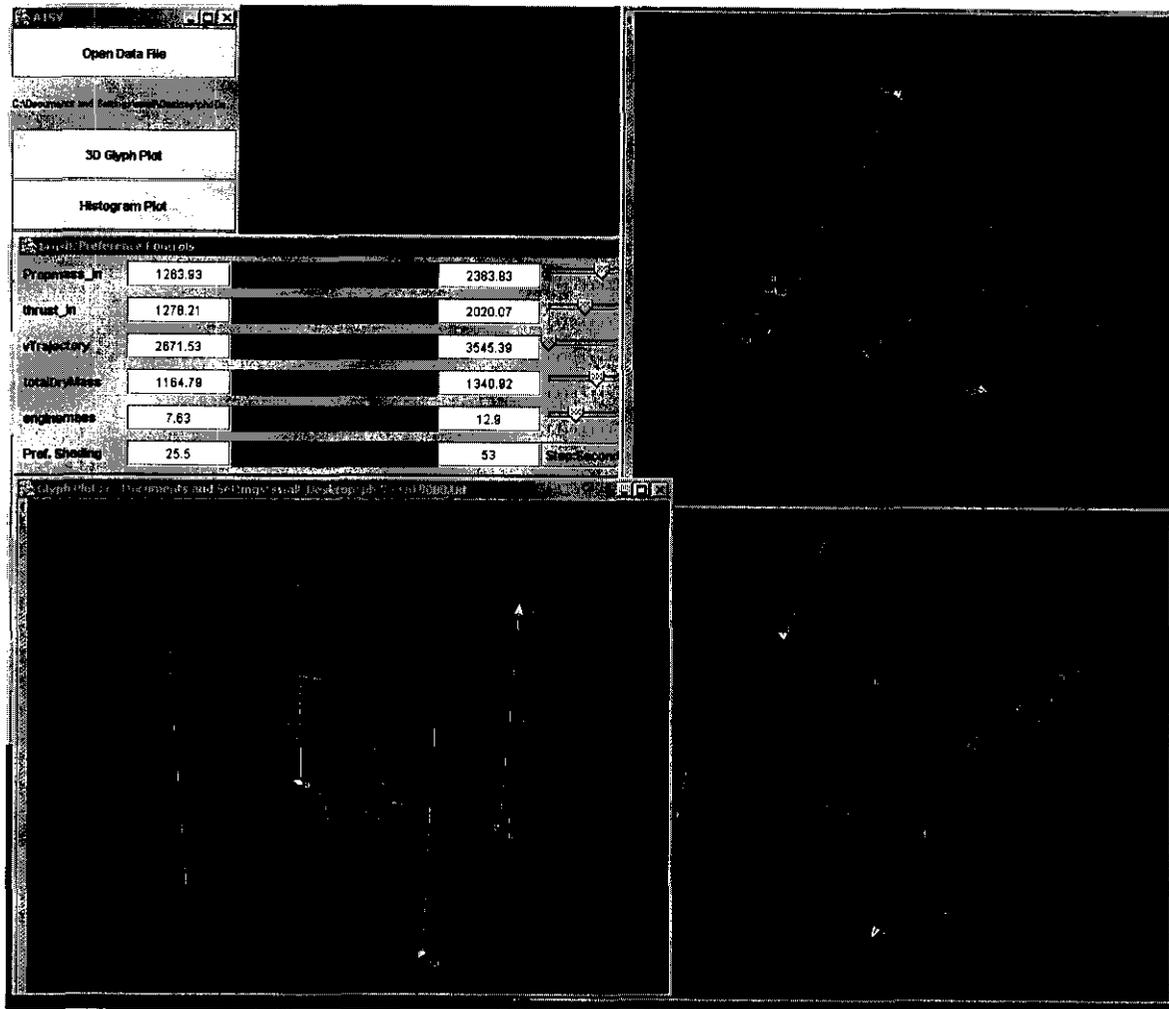


Figure 2 - Tradespace Visualization Tool

Translation of the tradespace into a virtually physical space makes understanding of the design problem similar to discerning and recognizing patterns in the other domains. In particular, techniques for multi-dimensional visual data mining can be brought to bear. If one or more objective (or merit) functions can be defined using some combination of the parameters in the space, shading or other graphic highlighting can highlight the designs that most satisfy that objective. Any of a number of classic optimization techniques can be applied and their optimization path plotted within this virtual space for team observation and comment. If no quantitative merit function can be defined, as is often the case when evaluation of science merit depends on consensus of teams of scientists, Pareto frontiers and other techniques for “shopping” within the tradespace can be applied [16, 17]. In either case, we anticipate that giving a design team the ability to visualize the costs and benefits of the populated tradespace will be highly valuable in the process of selecting an optimal design point.

A candidate solution, still at the 10-to-20-parameter level of fidelity, will be picked from the tradespace and

forwarded to Team X for its usual design and costing efforts. In the formulation phase, where fidelity of design increases and primary concerns become requirements development and evaluation of subsystem feasibility, the MBED initiative utilizes two model types: requirements models and subsystem models. Requirements models expand on the original concept [9] with the goal of expressing the requirement with a minimum of design. As a simplified example, consider this single requirement, stated in text, which might be a part of a typical level-2 automobile requirements package: “vehicle shall accelerate to 60 mph in less than 20 seconds using less than 6 ounces of fuel.”

Before looking at the model-based equivalent to this requirement, note three points about this statement and the associated code. First, although the requirement is basically functional, there is no explicit time-dependent behavior modeled: parameter values are simply changed. Second, it specifies a minimum of design, although it does imply the existence of an acceleration function. Third, it does not speak to the realizability of the requirement: presumably, some engineer would be asked to consider

signing a document containing this requirement if (and only if) he/she believed it to be realizable. The same requirement, retaining these same three points, could as well be stated in software, as shown in Figure 3. Execution of the code shown would, of course, guarantee the passage of 12 seconds of simulated time, the achievement of 60 mph, and the usage of 6 ounces of fuel.

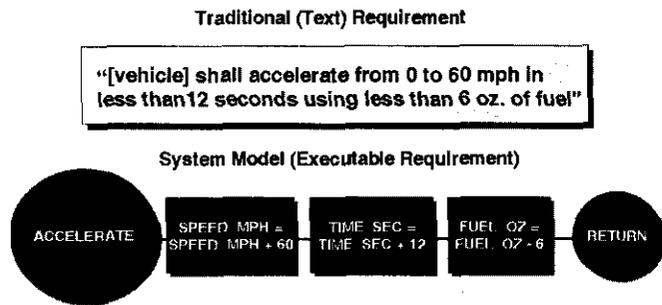


Figure 3 - Idealized Executable Requirement

In this simple example, we have ignored the issues associated with the inequalities in the text requirement and deal only with the minimum acceptable values. At a superficial level it is easy to deal with the inequalities in such a model, although in practice it is common to reduce requirements to single values (“zero-width” requirements), whether they are stated in text or in models. It is worth noting that development of a robust mechanism for handling “non-zero-width” requirements in the design process, albeit beyond the scope of this work, is underway (e.g., [18]).

What is the advantage of the code over the text? It still satisfies the three previously-noted conditions, but it has the additional property of testability because it can be executed. In an appropriate environment, and with sufficient knowledge of the road and driving directions (analogous to the mission scenario), one could test this model without any reference to subsystem models or to realizability to see if this vehicle could reach a given destination with a given-sized gas tank. With similar requirements models of braking and steering, a reasonable model of the entire vehicle could be written, and resource requirements could be developed. A typical space mission described at this level might consist of thousands of such requirements, and to analyze the capability of the described mission to satisfy a typical science scenario while still in the requirements stage would be a formidable task given text-based requirements. To analyze the effects on a scenario of a change in requirements, an all-too-common occurrence, is as difficult or more so.

In a recent attempt to extend this idea to more realistic practice, we used iLogix’s Statemate [19] to construct a requirements model of LightSAR, a simple spacecraft with a single radar sensor. Although it is primarily a state-based modeling tool, Statemate has the ability to produce a functional model by populating each of its “states” with

code similar to that used in the earlier example. Following the example, the spacecraft is represented as a series of “states,” each of which has transition rules for entering and exiting and a set of parameters that are set or modified as a consequence of being in the “state”. Each “state” represents a function, but the action of the function is only to force satisfaction of the requirements by changing parameter values. The radar “state” has a datarate and power associated with it, as do telecom, guidance_nav_cntl, etc. The model is driven by APGEN to simulate the activity required of the spacecraft and radar; power and data resource requirements are developed from this simulation. Trades among available resources, design, and scenario can easily be made.

As is the case with text-based requirements, it has been a challenge to completely avoid specification of some level of design, but the points made in the simpler example were taken as goals. We used this model to mature the LightSAR requirements in a manner analogous to the traditional functional allocation and requirements flowdown process, ending at a point that might be close to a full requirements-level design specification, perhaps the rough equivalent of four levels of text-based requirements documentation.

The model is depicted in Figures 4 and 5. Figure 4 is a top-level description of “states” that the spacecraft can occupy—more accurately, these are modes that the spacecraft will use to fulfill requirements. Note that these are functional requirements in state-like representations, each of which may or may not result in physical subsystems. Each is decomposable into lower levels (just as in the traditional hierarchical requirements allocation process), one of which is shown at its lowest level in Figure 5, which represents the radar command and telemetry functions. At this level, transition rules are also shown, analogous to the rectangular boxes in Figure 3. For example, transitioning from “radar-to-databus” state to “databus-to-radar” state is triggered by variable changes, as noted in the code adjacent to the transition arrow. A particular feature of this environment is that, like our simple example, it is executable. In the mock design pilot [13], science scenarios that describe data-taking and downlink events were used to simulate the operation of LightSAR and to make trades of onboard memory requirements against number of targets acquired, number of downlink stations required, and downlink data rate.

In later formulation phase, another type of model is necessary to transition into the use of detailed design tools such as CAD, ECAD, and software coding environments. Here MBED will adopt a set of “multimission,” or “MM” models that are more detailed representations, this time at the subsystem level [20]. An example of a user interface from the power subsystem model, Multimission Power Analysis Tool (MMPAT), is shown in Figure 6. These

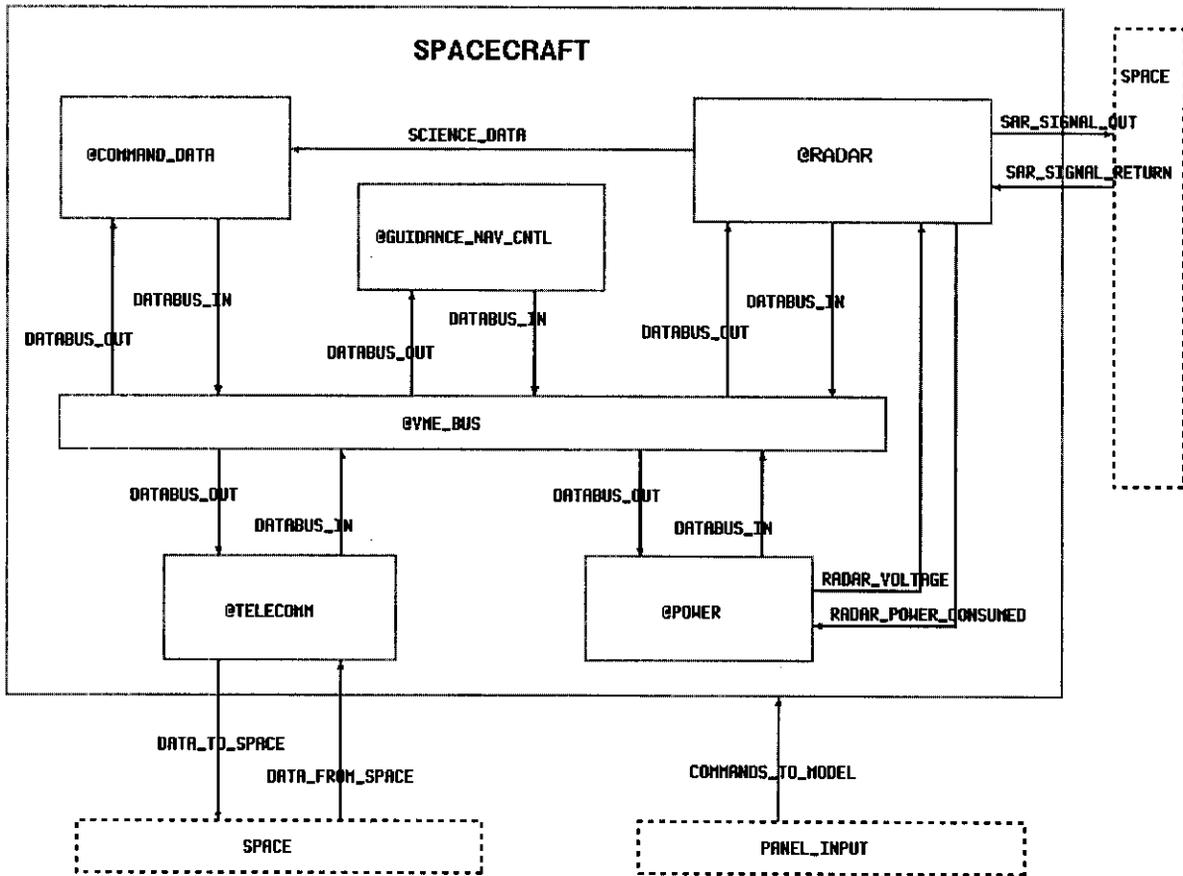


Figure 4 - Requirements Model of LightSAR in StateMate

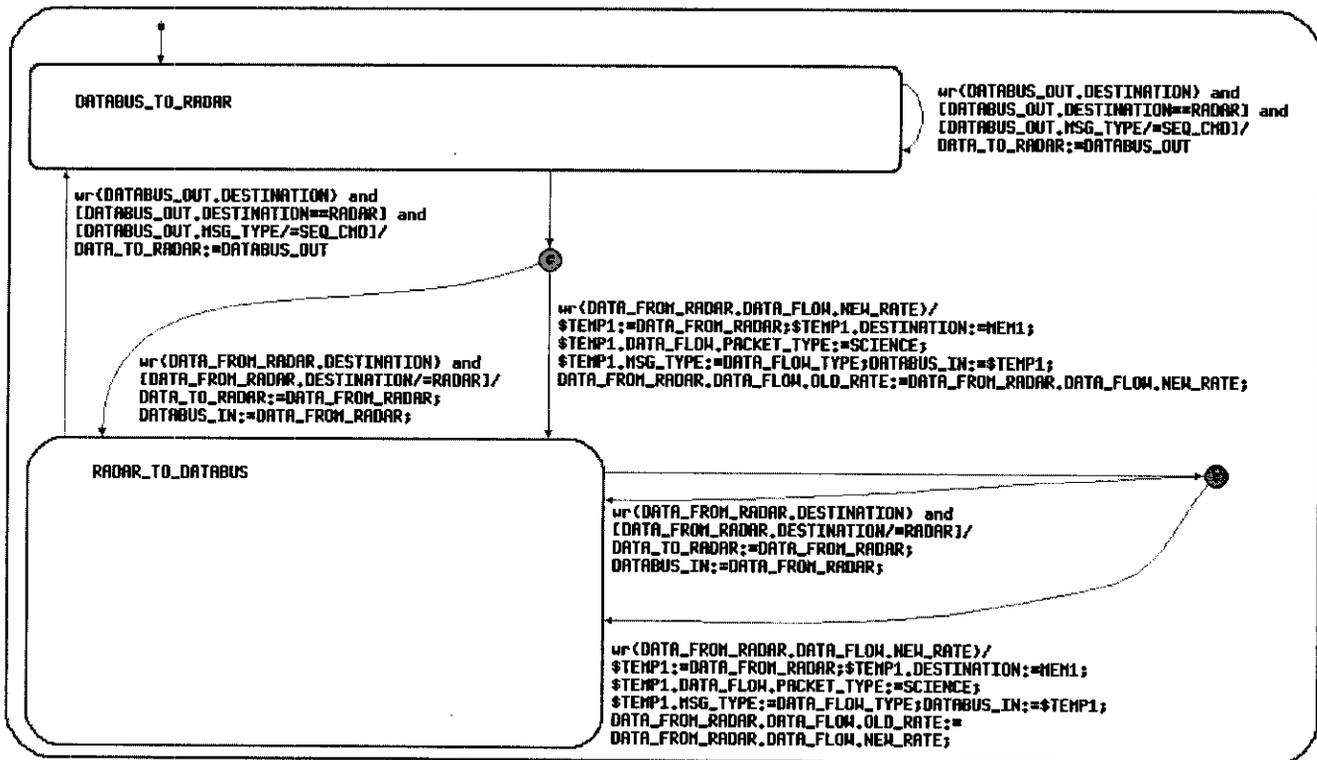


Figure 5 - Low-Level State Model of Radar Command and Telemetry Function

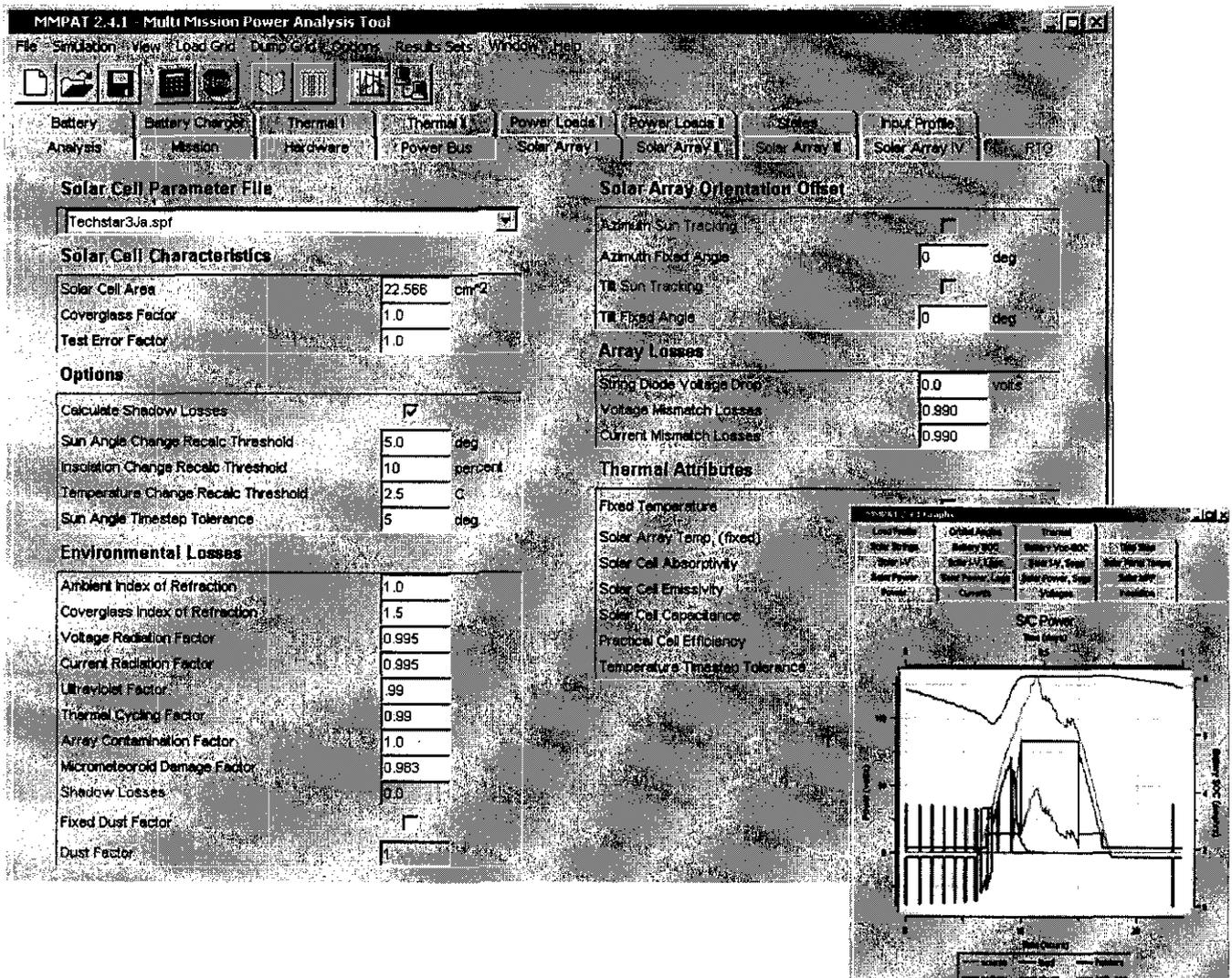


Figure 6 - Graphic User Interface for the MMPAT Tool (from [14]). Inset Shows Developed Resource Profile.

tools allow rapid selection of prototype components, parameters, and modes from menus, after which they can be driven by user scenarios from mission profiles (roughly equivalent to the concept of use cases) to provide dynamic simulation of the protosubsystem, at one step more detail than those generated with requirements models. In this way resource profiles can be generated (Figure 6, insert), and high-level trades can be made of user parameters against system design. With the addition of cost models, mission-wide trades can be analyzed. For example, in a planetary mapping mission, an important user parameter might be number of images, or surface resolution. Using scenario-driven MM models, trades of mission cost, solar panel selection, datarate, battery size, and number of images can be made with full knowledge of the subsystem design. As with the requirements models, if a merit function can be defined, formal optimizing techniques can be used at this lower level, or shopping techniques can be applied. Either way, subsystem design points are established, and the subsystem designer can transfer the prototype subsystem

design into the appropriate design tools to proceed to detailed design.

The MM subsystem models are currently under development at JPL. MMPAT is in limited use, and two additional models, for telecommunication and propulsion, are being evaluated for use. The full suite of models will describe all typical subsystems in space missions, and will include command and data handling and thermal subsystems.

5. SUMMARY AND CONCLUSIONS

In this paper we have described the logical progression from requirements models to subsystem models that form the heart of the model-based engineering design process envisioned for the MBED initiative. Requirements models are used to capture requirements with a minimum of design, as are text-based requirements, and with no attempt to address feasibility. Their chief advantage over text-based requirements is that they are executable—that is, they can be driven to simulate performance. Science or mission

scenarios, which represent users' operational requirements, are used to create the simulation and guarantee internal consistency of requirements, specifically addressing whether a system built to the system requirements can satisfy the user scenarios. By this method the entire tradespace of system, mission, and (with the addition of cost models) programmatic parameters can be explored. Trades can be made between user and system requirements, and between these and total mission cost.

Subsystem models are used primarily to demonstrate feasibility and to provide links to detailed design tools. The MM models described here will represent each of the classical subsystems of a spacecraft and allow rapid configuration of prototype components, parameters, and configurations. Like the requirements models, the MM models can be driven with scenarios to produce detailed resource usage profiles, this time testing whether the subsystems as configured will satisfy the user demands.

Model-based design has been a long time coming, and it still has a way to go before being accepted design practice. With the MBED initiative, we will assemble the models described in this paper and use them to extend the current JPL conceptual design process.

6. ACKNOWLEDGEMENTS

The author would like to thank H. Lykins and the Model Driven Design Working Group of INCOSE for inspiration leading to this work. Many JPL personnel have participated in the development of the concepts discussed here, including D. Smith, A. Freeman, J. Sercel, and others. I am especially indebted to E. Antonsson for many useful discussions, and to the continuing encouragement of E. Mahen. The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration and funded through the internal Research and Technology Development program.

REFERENCES

- [1] http://www.space.com/peopleinterviews/mccurdy_profile_000419.html
- [2] J. Sercel, T. Clymer, and W. Heinrichs, "The Product Attributes Database (PAD): First of a New Class of Productivity Tools for Product Development," *Proceedings of the IEEE Aerospace Conference*, 1998.
- [3] <http://www.geia.org/sstc/sstcmeeting/Jun2002/g47/nextgense.pdf>
- [4] S. Wall, "Use of Concurrent Engineering in Space Mission Design," *Proceedings of EuSEC 2000*, Munich, Germany, September 2000.

- [5] See, e. g., www.acq-ref.navy.mil/tools/turbo/topics/bk.cfm
- [6] JPL, TRW, Aerospace Corp, "DCE—a Breakthrough Process," *Proceedings of the IEEE Aerospace Conference*, 1999.
- [7] G. Mark, "Extreme Collaboration," *Communications of the ACM*, December 2001.
- [8] L. Koenig, D. Smith and S. Wall, "Team Efficiencies within a Model-Driven Design Process," *Ninth Annual International Symposium of the International Council on Systems Engineering*, 1999.
- [9] L. Baker, P. Clemente, R. Cohen, L. Permenter, B. Purves, and P. Salmon, "Foundational Concepts for Model-Driven System Design," *Proceedings of the INCOSE Model Driven Systems Design Working Group*, 1997.
- [10] C. Ramchandani, M. Maier, and T. McKendree, "Toward a Comprehensive System Architecture Representation Model" *Fifth Annual International Symposium of the International Council on Systems Engineering*, 1995.
- [11] "The Role of Modeling in the Development of Large Systems", J. Hodapp and S. Hyer, *Fifth Annual International Symposium of the International Council on Systems Engineering*, 1995.
- [12] S. Wall, "Reinventing the Design Process: Teams and Models," *International Astronautical Federation Specialist Symposium on Novel Concepts for Smaller, Faster and Better Space Missions*, Redondo Beach, CA, April 1999.
- [13] S. Wall, "The Mission System Design Center: A Pilot of Formulation-Phase Concurrent Engineering in Aerospace Design," *Tenth Annual International Symposium of the International Council on Systems Engineering*, 2000.
- [14] P. Maldague, A. Ko, D. Page T. Starbird, "APGEN: A Multi-Mission Semi-Automated Planning Tool," *International Workshop on Planning and Scheduling for Space Exploration and Science*, 1997.
- [15] E. N. Harris, M. Yukish, and C. Paredis, Final Report: "University-Based Research in Support of New Design Initiative" (Document No. NRO000-00-C-0146), Lockheed Martin Space Systems Company, Denver, Colorado, October 2001; and M. Yukish, personal communication.
- [16] B. Wilson, D. Cappelleri, T. Simpson, M. Frecker "Efficient Pareto Frontier Exploration Using Surrogate Approximations", *Optimization And Engineering* 2, 31–50, 2001.

[17] R. Balling, "Design by shopping: A new paradigm?," *Proc. Third World Congress of Structural and Multidisciplinary Optimization*, C. L. Bloebaum and K. E. Lewis et al., eds., Buffalo, NY, University at Buffalo vol. 1, May 17-21, 1999, pp. 295-297.

[18] E. Antonsson and K. Otto, "Imprecision in Engineering Design," *ASME Journal of Mechanical Design*, Volume 117(B) (1995), pages 25-32.

[19] <http://www.ilogix.com/products/magnum/index.cfm>

[20] M. Kordon and E. Wood, "Multi-Mission Space Vehicle Subsystem Analysis Tools," *Proceedings of the IEEE Aerospace Conference*, 2003.

BIOGRAPHY



Steve Wall is a Principal Engineer at the Jet Propulsion Laboratory, California Institute of Technology, where he leads JPL's Center for Space Mission Architecture and Design. Steve has participated in seven major space missions in design teams, science teams, operations teams, and management. Current research interests include advanced system design, concurrent engineering, and other rapid design methods. He holds a Masters in Optical Engineering from the University of Rochester and a BS in Physics from North Carolina State University. For his past work Steve has been awarded the NASA Exceptional Achievement Medal, the Exceptional Service Medal, and six NASA Group Achievement Awards.