

# MOS 2.0: The Next Generation in Mission Operations Systems

Duane L. Bindschadler,<sup>\*</sup> Carole A. Boyles,<sup>†</sup> Carlos Carrion,<sup>‡</sup> and Chris L. Delp<sup>§</sup>  
*Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 91109*

A Mission Operations System (MOS) or Ground System constitutes that portion of an overall space mission Enterprise that resides here on Earth. Over the past two decades, technological innovations in computing and software technologies have allowed an MOS to support ever more complex missions while consuming a decreasing fraction of Project development budgets. Despite (or perhaps, because of) such successes, it is routine to hear concerns about the cost of MOS development. At the same time, demand continues for Ground Systems which will plan more spacecraft activities with fewer commanding errors, provide scientists and engineers with more autonomous functionality, process and manage larger and more complex data more quickly, all while requiring fewer people to develop, deploy, operate and maintain them. One successful approach to such concerns over this period is a multimission approach, based on the reuse of portions (most often software) developed and used in previous missions. The Advanced Multi-Mission Operations System (AMMOS), developed for deep-space science missions, is one successful example of such an approach. Like many computing-intensive systems, it has grown up in a near-organic fashion from a relatively simple set of tools into a complexly interrelated set of capabilities. Such systems, like a city lacking any concept of urban planning, can and will grow in ways that are neither efficient nor particularly easy to sustain. To meet the growing demands and unyielding constraints placed on ground systems, a new approach is necessary. Under the aegis of a multi-year effort to revitalize the AMMOS's multimission operations capabilities, we are utilizing modern practices in systems architecting and model-based engineering to create the next step in Ground Systems: MOS 2.0. In this paper we outline our work (ongoing and planned) to architect and design a multimission MOS 2.0, describe our goals and measurable objectives, and discuss some of the benefits that this top-down, architectural approach holds for creating a more flexible and capable MOS for Missions while holding the line on cost.

## Nomenclature

*AMMOS* = Advanced MultiMission Operations System  
*ATLO* = Assembly, Test, and Launch Operations  
*MGSS* = Multimission Ground Systems and Services  
*MOS* = Mission Operations System

## I. Introduction

**T**HIS paper describes the vision and motivation behind an effort to formulate and implement a systematic approach to multimission operations for deep-space NASA missions. The effort is a multiyear initiative, referred to as Operations (Ops) Revitalization. It is funded by NASA via the AMMOS program, and managed by the MGSS program office at JPL. The initiative draws on systems architecting and model-based engineering approaches that have been applied successfully to a wide variety of software-intensive systems. The desired

---

<sup>\*</sup> Asst. Program Manager for Operations, MGSS, M/S 264-235, 4800 Oak Grove Dr., AIAA Senior Member.

<sup>†</sup> Asst. Program Manager for Planning & Development, MGSS, M/S 264-235, 4800 Oak Grove Dr.

<sup>‡</sup> Mission Operations System Engineer, System Engineering Section, MS 264-214, 4800 Oak Grove Dr.

<sup>§</sup> Architectural Engineer, Flight Software and Data Analysis Section, MS 264-214, 4800 Oak Grove Dr.

outcome is a set of products and practices that represent a significant improvement in the area of missions operations: MOS 2.0. A companion to this paper<sup>1</sup> describes the techniques and methodologies utilized, and the lessons learned to date in developing MOS 2.0.

This work is motivated by the ongoing desire to decrease the cost of providing Mission Operations Systems for deep-space missions, while improving MOS capabilities to plan, execute, and return science data, and continuing to do so at extremely low risk to mission success and achievement of science objectives. Reviewing and synthesizing some of the experiences of the past few decades of mission operations at the Jet Propulsion Laboratory, we find that factors that have previously enabled the decreasing MOS costs (even while increasing their capabilities) are providing less return, or are even becoming less effective. New approaches are therefore necessary.

We rely heavily on JPL's 40-plus years of experience in developing systems for deep-space operations and executing flagship missions such as Mariner, Viking, Voyager, Galileo, Cassini, Magellan, and the Mars Exploration Rovers, as well as missions rooted in the "faster, better, cheaper" approaches of the 1990's (Stardust, Genesis, MGS, Mars Odyssey and many others).

By intent, the effort is evolutionary, particularly in that many operating missions continue to rely on support from the current generation of AMMOS tools and services. But this is not to say that it will not disrupt specific practices or paradigms. Rather, one of the guiding intents is to establish the architectural framework within which changes can be identified and their impacts managed.

## II. Background

Viewed from a high-level perspective, the demands on an MOS have changed little over the past few decades. Simply put, a mission operations system is charged with performing the command and control functions of a mission that cannot be done autonomously by a flight system. In practice, this means formulating and specifying plans and commands for the flight system to execute, capturing and analyzing the data returned by the flight system, and maintaining the coupling (feedback) between these two sets of activities. An example of the functions performed by a MOS is shown in Figure 1.

In looking beyond the high-level functionality of an MOS, changes from the 1980's and 1990's become more apparent. Improvements in computer hardware and software, exploitation of the Internet and emerging web technology have allowed for significant automation. The cost of developing even large, complex software applications has decreased. Computing and networking hardware have become commodity items, with costs decreasing even as network and computing speeds increase and data storage capacities grow. Most past and current missions at JPL have benefitted significantly from these advances. Despite (or perhaps because of) such successes as well as the cost-consciousness created by competitive selection processes for missions, there are continued pressures to reduce MOS costs, both in development and operations. However, many areas where automation can easily be applied have already been exploited; cost savings due solely to cheaper computing will only yield incremental benefit.

The 1990's also saw the creation of a multimission approach to mission operations at JPL. Looking ahead to plans for multiple "faster, better, cheaper" missions to Mars, managers and engineers set out to create a single operations organization that would serve multiple flight projects. Utilizing standard adaptable processes, interfaces, software, and team structure, they designed and created the basis for the current approach to multimission operations at JPL. Extensive use of this approach and exploitation of new technologies enabled extremely low-cost mission operations for multiple Mars missions, and was also responsible for keeping MOS development and operations costs at low levels for missions such as Genesis, Stardust, Mars Global Surveyor, and Mars Odyssey. This approach has worked well for a particular set of missions, characterized by:

- A planetary orbiter or flyby mission concept
- A relatively small number of individual science instruments ( $\leq 4$ )
- A high-heritage spacecraft bus, built by a single system contractor (Lockheed Martin Space Systems)
- Spacecraft operations teams, operated by same contractor, and following a similar multimission operations model
- Use of a shared ground data system, with minimal customization required for the individual missions.

These missions make the most usage of the current AMMOS multimission capabilities, utilizing nearly all applicable multimission teams, services, and tools. We refer to them as "high-usage" missions. A second subset of missions makes selective use of AMMOS capabilities, including MRO, Spitzer, Deep Impact/EPOXI, and Dawn ("moderate-usage" missions). They typically use some but not all applicable multimission teams and services. They also commonly have ground data systems that include key, project-specific software, or highly customized

implementations of AMMOS tools. Such missions diverge from one or more of the characteristics noted above. A third group of Missions have made some usage of AMMOS tools, but relatively little usage of multimission teams

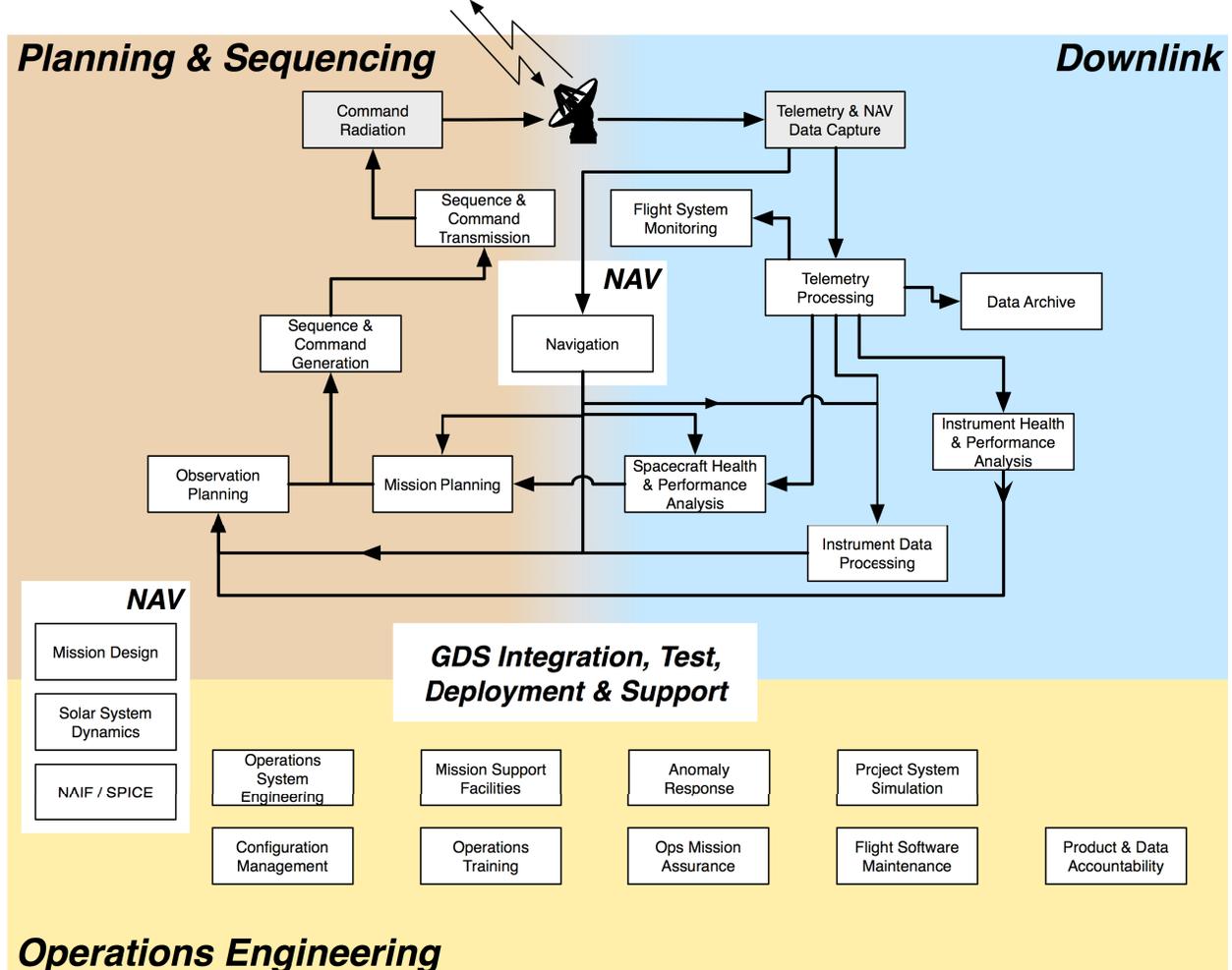


Figure 1. Common functions of a Mission Operations System for a NASA Deep Space Mission.

or services ("low-usage" missions). These include MER, Phoenix, and Cassini. For missions currently under development, both GRAIL and Juno make extensive use of the AMMOS. The MSL mission utilizes some AMMOS tools and will rely on the AMMOS Mars Relay services for relaying most commands and telemetry via MRO and Odyssey.

The current pattern of usage of the existing AMMOS capabilities suggests that it is well worth revisiting its underlying architecture. While mission decisions to adopt (or not) particular AMMOS services commonly reflect multiple technical and programmatic aspects, there are recurring themes. One in particular is that the AMMOS is inflexible and will therefore not serve the needs of Mission "X." For missions that are highly similar to previous high-usage customers (e.g., GRAIL, Juno), direct experience tends to refute such contentions. This is not the case for MSL, or for many of the other mission concepts that currently vie for NASA funding. Architectural approaches that explicitly consider the broad range of concerns, and models that allow those concerns to be traced to specific elements of the system are needed if the range of AMMOS-applicable missions is to be expanded. This is particularly true for rovers or landers (or other future mission types) that operate on a daily tactical planning/commanding cycle.

It is also true that multi mission approaches are in need of refurbishment. Much of the foundation for the multi mission teams utilized infrastructure and experienced personnel from previous missions such as Mars Observer, Galileo, and Magellan. Cost savings during "faster, better, cheaper" was partly achieved by minimizing documentation and thus the costs associated with maintaining and updating that documentation. Without sufficient

effort focused on maintaining and evolving multimission approaches, the knowledge developed has begun to dissipate as the natural turnover in personnel occurs.

### III. Definitions

For the purposes of this paper, it is useful to consider definitions for several key terms. For deep-space mission operations in the context of a particular mission, we consider a Ground System to include all hardware, software, networks, facilities, people, and processes used to operate a Flight System (or Systems) for a particular Project. The AMMOS finds its use as part of a Mission Operations System (MOS), which we define as that portion of the Ground System that is managed within a particular Project organization. These definitions, as well as relationships between AMMOS and other aspects of the mission operations domain, are illustrated in Figure 2.

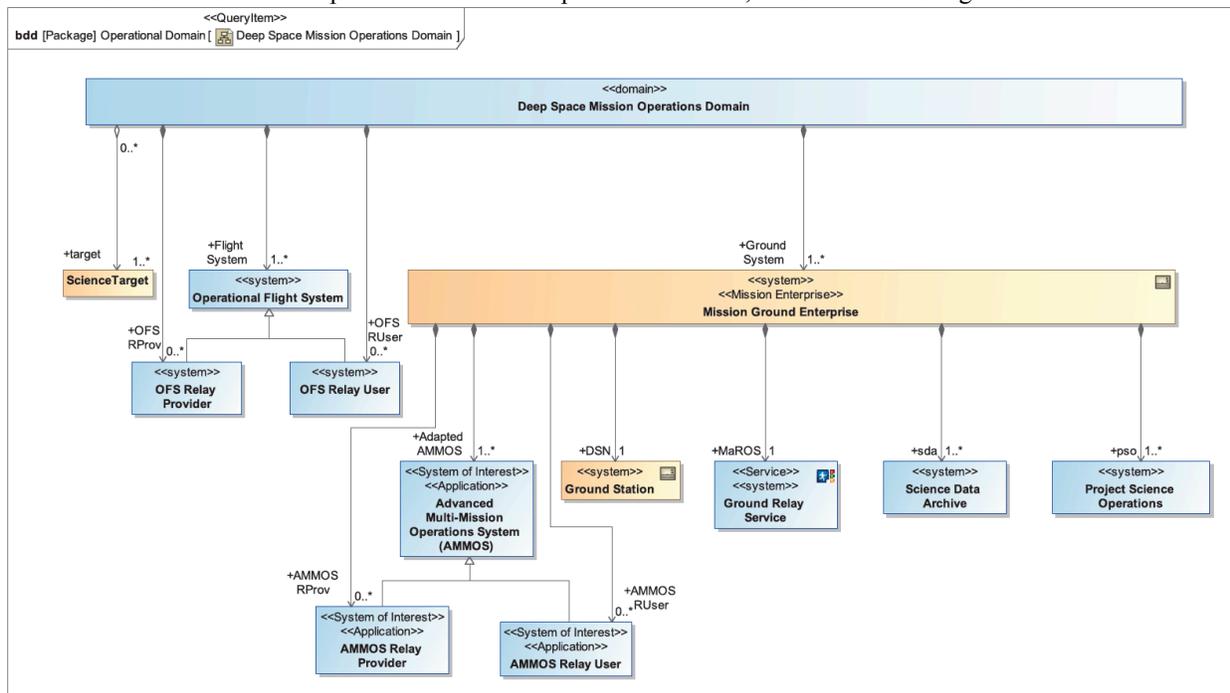


Figure 2. UML block diagram illustrating MOS within the overall context of the domain of deep-space mission operations.

### IV. Motivation

#### A. Why Architect and Model?

The rationale for our choice to address an MOS 2.0 by means of model-based techniques and from an enterprise and system architecting approach may be evident. Our experiences and findings to date in this initiative<sup>1</sup> certainly provide support for that choice. Nonetheless, it is worth noting that systems and enterprises have architectures, *whether or not they are explicitly documented and maintained*. Consideration of the architecture of a system of interest should be done with deliberate intent, with consideration of key stakeholder needs across relevant organizations. The alternative is that consideration of architectural concerns is instead accomplished piecemeal within organizational or functional stovepipes. Since ground systems are subject to continual change as technologies and mission needs evolve, architectures must be maintained and evolved over time. The lack of a defined and coordinated effort inevitably leads to decisions that may be locally optimal, but risks broader impacts that outweigh any local benefit.

The ability to predict the effects of local changes across a system is one of the reasons for using model-based techniques. Alan Brown notes the following rationale for modeling as part of IBM's materials on model-driven architectures:

Models provide abstractions of a physical system that allow engineers to reason about that system by ignoring extraneous details while focusing on relevant ones. All forms of engineering rely on models to understand complex, real-world systems. Models are used in many ways: to predict system qualities, reason about specific properties when aspects of the system are changed, and communicate key system characteristics to various stakeholders. The models may be developed as a precursor to implementing the physical system, or they may be derived from an existing system or a system in development as an aid to understanding its behavior.

Although there is no substitute for the basic discipline of a carefully considered and coordinated process for maintaining understanding of a system architecture and making informed decisions about change, model-based techniques can facilitate such processes. In particular, models can maintain and display the connections between system elements in ways that require less overhead than conventional paper documents. And as capabilities to simulate system behaviors become more accessible, model-based methods promise to significantly improve the ability to explore alternatives and "what if" scenarios.

## **B. Domain-Specific Motivations**

In our examination of the ground system enterprise for current and future NASA missions, it is useful to consider several driving factors, which compete for preeminence during development and operation of a typical MOS:

- The need for lower costs for developing an MOS and executing flight operations, in order to remain competitive (especially for competed missions such as those from the Discovery and New Frontiers programs).
- The need to have a flexible ground system to support Project Integration & Test (aka ATLO) activities.
- The need to assure mission safety and minimize risk of mission loss or compromise of mission objectives.
- The need for a maintainable system, which can be called upon to support operations through multiple mission extensions.

These are shown in approximate time order -- that is, cost considerations are pre-eminent in the proposal or pre-formulation phase, up through the early portions of implementation (e.g., Phase C in NASA's project lifecycle). By the time System I&T/ATLO begins, flexibility and responsiveness of the ground system to changing needs of the Flight System (especially flight software) become overriding concerns.

Approaching Launch, the focus for the MOS becomes risk. Are personnel trained appropriately? Is the risk of mission-ending errors (either procedural or due to software) acceptably low? Are the right plans in place for contingencies at Launch and during early operations? These and similar concerns commonly recur prior to mission-critical or other high-priority events (e.g., orbital insertion maneuvers, entry, descent, and landing (EDL), critical flybys of a science target).

During long cruise periods, or Extended (i.e., post-primary or -nominal) Mission phases, the overriding concern becomes the need to maintain the MOS. This must occur in the face of level or declining budgets, the inevitable Mission-external changes (e.g., hardware/software obsolescence, changing standards and interfaces), and internal changes (e.g., loss of personnel, changes in Flight System performance.).

Development costs for an MOS generally represent a relatively small fraction of the overall Mission development cost (~10-20%). The MOS has historically been accorded lower priority during development, and decisions made during that time tend to give more weight to flight hardware and software concerns or to those involving Project I&T activities. In the context of a single Flight Project, this is both necessary and understandable. But the effect is that MOS managers and engineers can find it difficult or impossible to make decisions based on a balanced, lifecycle perspective. And all too frequently, this has meant shifting of costs or risk from development into operations.

The charter of both this effort and the MGSS organization requires that we adopt a lifecycle perspective. A robust architectural description and approach are necessary if we are to support both Project-specific, relatively near-term needs while providing for long-term maintainability and cost effectiveness for NASA.

The ability to effectively gauge the impact of changes within or external to a system is one of the key aspects of an effective architectural model. In the design of buildings, architects and structural engineers routinely develop and interrogate models to assess the response of a proposed building or bridge to the stresses imposed by the structure

itself or by external forces such as wind or seismic shaking. Buildings are designed with particular uses in mind. Changes in usage (e.g. from office space to server rooms) must be assessed against the electrical, structural, and HVAC implications of such changes.

For a Mission Operations System, we seek to create analogous understanding and models to capture that understanding. What happens when a telemetry processing system is upgraded from a 1980's vintage to a modern Service Oriented Architecture (SOA) based implementation? How does an MOS design for a standalone Mars orbiter change if that mission must also support relay operations for multiple surface rovers or landers? What key changes to existing MOS tools and services are needed to support the next generation of potential missions (e.g., Mars, asteroid, or comet sample return, Venus or Titan landers or balloons, etc.)? To address fundamental issues of missions operations, a multimission architecture must facilitate answering such questions.

Beyond the Ground Enterprise (Fig. 2) there are potential long-term implications to a model-based, system-architected MOS 2.0. Similar approaches are increasingly being adopted for the development of Flight System capabilities. At the same time, technologies for flight system autonomy continue to improve, and have demonstrated significant value<sup>2</sup>. Complexity of hardware and software for spacecraft and instruments continue to increase in order to address ambitious science objectives. Looking to the future, missions must be able to choose whether particular capabilities and functions are better performed by flight software or by ground software. Such choices are complex, and include attendant consideration of lifecycle costs to the mission, mission risk, and ripple effects within and across Flight and Ground Systems.

Models can provide a common language for addressing Mission-level concerns across the Flight-Ground interface. And by utilizing the tools of modeling and the discipline of architecting, future missions will be able to make intelligent trades between flight and ground functionality that optimize (for example) science return per dollar spent. Developing MOS 2.0 puts the overall Ground Enterprise in a position to support the next generation of NASA Mission architectures.

## V. Constraints

There are two types of constraints on the Ops Revitalization initiative that must be considered. The first are those constraints related to the existing mission customers of the AMMOS, and the existing software that makes up the AMMOS GDS. Because we must continue to support missions that currently utilize AMMOS capabilities, changes generally must proceed in relatively small increments. This is particularly true for the “high-usage” AMMOS customers, whose mission teams and budgets tend to be small and would have little desire to support (for example) user acceptance testing of major new capabilities. And although improvements in AMMOS business processes will need support from software and technology architectures, such changes can incur significant costs for developing new capabilities. For the time being, we treat the existing software as a potential constraint on our business processes. One of the anticipated benefits of MOS 2.0 is to facilitate identification of “pain points” in our processes. Once identified, such problem areas can be prioritized and worked off over time.

The second major type of constraint comes from institutional policies and standards. These include such requirements and guidance as NASA’s 7120.5d, JPL’s Flight Project Practices and Design Principles, IT and physical security requirements, and even export compliance regulations such as ITAR. It is essential for the Ops Revitalization initiative to incorporate and comply with these items. In fact, architecting the MOS 2.0 to deal with such standards (and the fact that they are likely to evolve over time) represents another avenue for providing value to mission customers.

## VI. Vision, Goals, and Objectives

The primary mission statement and the justification for the AMMOS is summed up in the following:

The AMMOS exists to provide multi-mission tools and services that enable mission customers to operate at a lower total cost to NASA, with comparable or higher reliability and performance, than would be the case if these customers acquired their own unique tools and services.

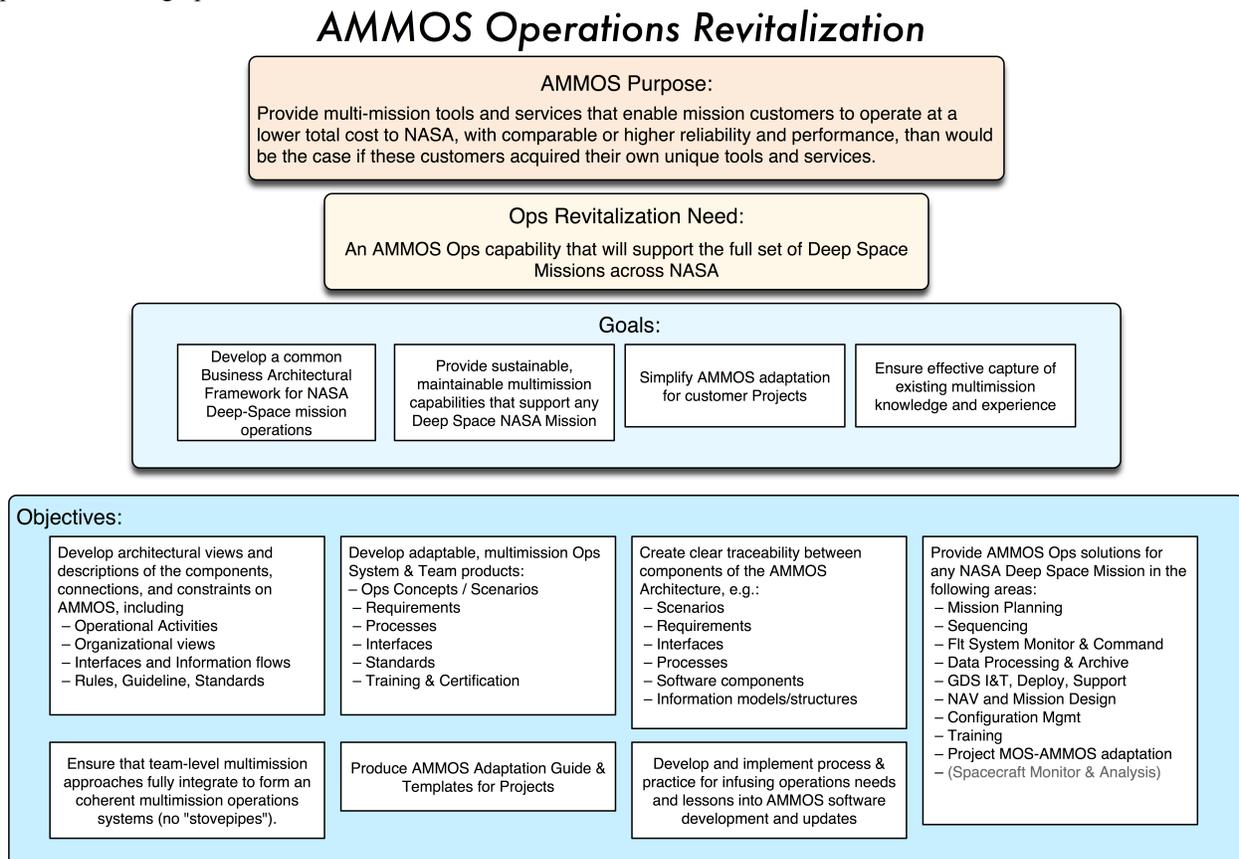
Within the context of that larger requirement, this Initiative satisfies the need to provide an AMMOS Operations capability that will support the full set of Deep Space Missions across NASA. In order to direct our efforts to create an MOS 2.0, we have set four high-level goals (see Figure 3). Below, we discuss each in turn.

## A. Develop a Common Business Architecture

The first point refers to a Business Architecture. The Open Group<sup>3</sup> defines an enterprise architecture as consisting of

- Business Architecture
- Data Architecture
- Applications Architecture
- Technology Architecture

This initiative is particularly focused on the Business Architecture and some aspects of the Data Architecture. The business of the AMMOS is the development and execution of multi mission operations for customer missions. The Business Architecture represents concerns about operational processes, team and individual roles and responsibilities, and the interfaces between teams, between operators and software, and with entities external to the MOS. The portion of the Data Architecture that is addressed identifies the types of information that are used and produced during operations.



**Figure 3. Goals and Objectives to be satisfied by MOS 2.0.**

## B. Provide Sustainable, Maintainable Multimission Capabilities

The goal of making MOS 2.0 a maintainable, sustainable system arises from previous experiences. Inadequate documentation of capabilities created during the "faster, better, cheaper" era has made it difficult or impossible to repeat or refine such capabilities. Highly useful solutions engineered for specific mission needs have turned out to have unforeseen costs or impacts when applied to new missions. One example is an extremely useful automation script within the AMMOS that has become integral to operations for several missions, but is becoming increasingly difficult and expensive to update. These difficulties trace back to the initial architecting and design of the script, which never envisioned the current breadth of usage.

### **C. Simplify AMMOS Adaptation for Customer Projects**

Simplifying adoption by Missions is significant if the AMMOS is to provide NASA with a significant cost benefit. Within the past two years, an AMMOS online catalog has been designed and implemented, and the negotiation of Service Level agreements with Missions has been added to our standard practices. These measures represent improvements, but there is much that can still be done to ensure that AMMOS capabilities and customer needs are mutually understood, and that Missions are aware of how to take maximum advantage of AMMOS teams and processes. For missions that differ significantly from high-usage AMMOS customers, it is particularly important to demonstrate clearly and credibly where the missions unique needs can be met by AMMOS capabilities, and where mission-unique capabilities will need to be developed. In addition, interface descriptions must be clear and readily available, ranging from detailed file interface specifications and API's to operational interface agreements (OIAs) that indicate the relative responsibilities of teams and individuals participating in AMMOS processes such as uplink planning and sequencing.

### **D. Ensure Effective Capture of Existing Multimission Knowledge and Experience**

Effective capture of previous experience is important, particularly if MOS 2.0 is to avoid "re-inventing the wheel." A companion paper<sup>1</sup> describes the development of operational scenarios that have made significant gains toward achieving this goal, capturing a wealth of information and experience into operational scenarios, and mining it in order to populate the system model.

### **E. Operations Revitalization Objectives**

Objectives are specified as means to achieving our high-level Goals. They are intended to guide the efforts of the development team to develop the specific deliverables and capabilities necessary to accomplish the Goals. Many of the anticipated products of this initiative are indicated in the Objectives. These include documentation for each of the various capabilities listed in the Objectives (Mission Planning, Flight System Monitoring & Commanding, etc.) that have been architected to function together as a coherent Mission Operations System, and which are adaptable to the specific needs of customer Projects. As part of our methodology, the objectives are being used to write requirement statements that will guide individual team-level implementations, and will ultimately be used as verification criteria for MOS 2.0.

## **VII. Conclusion**

As noted previously, details of our approach, results to date, and some of our lessons learned are detailed elsewhere<sup>1</sup>. Our work to date shows significant promise in providing improved MOS capabilities at lower lifecycle cost to NASA than is currently achievable. At the time of this writing, a preliminary Architectural Description Document has been generated from the MOS 2.0 Operations Domain Architecture Model, and we are in the midst of preparation for an expert peer review of that document.

Our next steps include completion of the physical architecture; analogous to preliminary design for an MOS. Based on results to date we will begin pilot implementations at the team level for a few select teams in the spring, anticipating that there will be some iteration between the system- and team-level designs. We also will begin work on the Development and Deployment Domains of the architectural model during the next year. Subject to funding levels, we anticipate being able to provide results from either pilot or actual operations usage of parts of the MOS 2.0 at the next SpaceOps Conference.

## **Acknowledgments**

The authors wish to acknowledge the support of MGSS Program Management and the many useful suggestions and insights of MOS experts at JPL. This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

## **References**

<sup>1</sup> Carrion, C., Delp, C.L., Illsley, J., and Liepack, O., "Use of Operational Scenarios in Architecting MOS 2.0," *SpaceOps 2010 Conference Proceedings* (to be published).

<sup>2</sup> Chien, S., *et al.*, "The autonomous sciencecraft embedded systems architecture," *Systems, 2005 IEEE International Conference on Man and Cybernetics (0-7803-9298-1)*, 10-12 Oct. 2005, Vol.4, p. 3927.

<sup>3</sup> The Open Group, "The Open Group Architecture Framework (TOGAF) - Version 9, Enterprise Edition," URL: <http://www.opengroup.org/architecture/togaf9-doc/arch/>, cited 15 February 2010.