MAKING ADAPTABLE SYSTEMS WORK FOR MISSION OPERATIONS: A CASE STUDY

Barbara E. Holder
Michael E. Levesque

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Mail Stop 171-243
Pasadena, CA 91109
USA
barbara@jpl-devvax.jpl.nasa.gov
mikel@jpl-devvax.jpl.nasa.gov

ABSTRACT

The Advanced Multimission Operations System (AMMOS) at NASA's Jet Propulsion Laboratory is based on a highly adaptable multimission ground data system (MGDS) for mission operations. The goal for MGDS is to support current flight project science and engineering personnel and to meet the demands of future missions while reducing associated operations and software development costs. MGDS has become a powerful and flexible mission operations system by using a network of heterogeneous workstations, emerging open system standards, and selecting an adaptable tools-based architecture. This paper describes challenges in developing adaptable systems for mission operations and the benefits of this approach.

Key Words: User-configurable systems, ground data systems, mission operations.

1. INTRODUCTION

The Advanced Multimission Operations System (AMMOS) at NASA's Jet Propulsion Laboratory is based on a multimission ground data system (MGDS) that is adaptable to individual flight projects and thus eliminates the need for each flight project to build its own separate ground data system. MGDS goals [1] are to:

- Support each new mission with all of its mission-unique processing, as a ready adaptation to a baseline set of functional capabilities.
- Simplify data distribution and access services so analysts can readily locate and analyze their data using MGDS.
- Maximize multimission practices, in order to reduce training and operational costs, by using similar equipment and user interfaces across functions and missions.
- Support operations for 10-15 years, by evolving the MGDS in response to hardware improvements and changing mission needs or types.

The MGDS architecture is a set of layers of individual components that are added, replaced, or adapted, as technologies, and mission scenarios change [2].

The layers, shown in Figure 1, are:
- Applications
- Organization specific standards
- Industry standards
- Open system standards.

The components within the layers are the system's building blocks: libraries, tools, tables, scripts, and operations scenarios that may be reused by different flight projects and software developers to reduce development and maintenance costs. Most functions are provided via the tools [Figure 2]. A tool is software easily configurable to become an application with utility to users. The tools themselves are rarely modified, but by modifying scripts, tables, and operational

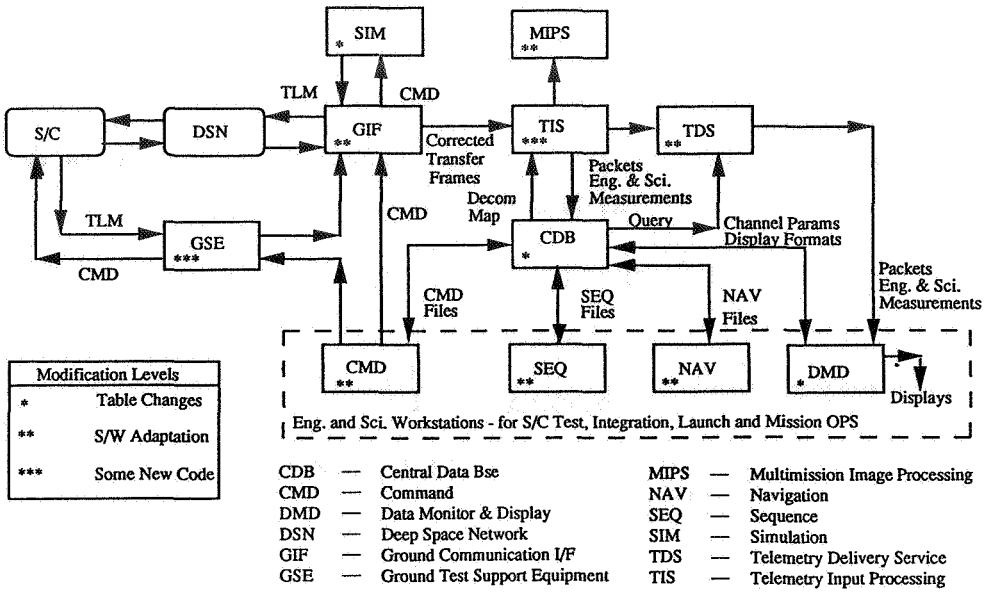

Figure 2. AMMOS Tool Suite

A system based on adaptable tools delivers high capability per development dollar, because the users evolve the system by modifying the scripts and tables to suit their needs. As a result, users discover highly efficient methods for accomplishing their tasks that were never anticipated by the developers or managers.

For example, the Magellan spacecraft team regularly did a variety of tasks on a daily, weekly, and one-time basis. The users automated repetitive tasks by writing scripts that performed the task for them. Analysts could then spend more time doing real work: mission analysis and anomaly resolution.

3.2 The Turn-key Example

Despite the success of Magellan's adaptation, it was clear that MGDS lacked the usability of a turn-key system. Many project managers and users thought they should not have to be involved in project adaptation. There was a discrepancy between what the developers gave the projects (an adaptable system) and what the projects were expecting (a turn-key system). To resolve this conflict, the development organization added a turn-key system delivery capability:

- adaptation teams as discussed earlier;
- a system user interface for novice users, based on the X.Desktop software package.

The system user interface (SUI) enabled novice users to become familiar with the tools and UNIX in a simplified user environment. However, once they became more familiar with MGDS and UNIX, they often abandoned SUI. MGDS's turn-key environment proved invaluable for novice users, but not for advanced users.

4. ADAPTATION DETAILS

4.1 System Adaptation

To meet MGDS's design goals of supporting operations for the next 10-15 years and evolving with hardware improvements and changing mission needs or types, MGDS was built on open system technology so it could be portable to other hardware and software environments. Figure 1 illustrates the open system layering [6] used in the MGDS design. Application programs (the tools) are not forced to change when new hardware and operating system software environments change.

As open system standards evolve, they will be incorporated into MGDS, and less capability will need to be provided by

scenarios the tools can be modified to meet a mission's specific needs.

MGDS users may customize their work environment themselves or with the assistance of a developer. The system evolves into an efficient mission operations environment when users are empowered to modify the building blocks or combine them in various ways to meet their mission-unique needs.

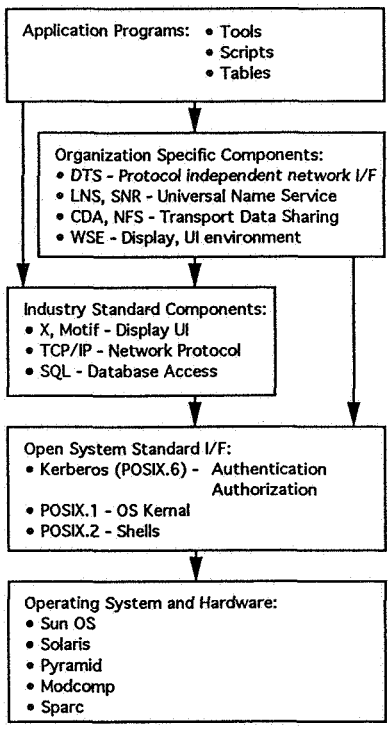

Figure 1. Open System Layer Model

2. MISSION OPERATIONS

MGDS is a suite of adaptable tools that support functions common to all missions. Each tool supports a generic mission operations function that may be needed by any mission.

When the tool suite is delivered to a flight project, it is adapted from the generic version to a project version. The adaptation process is provided as a service to the flight projects by the mission operations program office. The initial project adaptation is made by software developers and integrators, to meet project requirements and scenarios. The second adaptation is made by either a handyman user or an adaptation team comprising users and programmers; to meet general end-user and flight team needs. The final adaptation is made by the end-users who may further adapt the tools to meet their individual or team requirements.

An adaptation team creates an atmosphere of cooperation between the development organization and the operations organization. The adaptation team concept is described by MacLean et al [4] as a *handyman user* who facilitates communication between operators and developers, and responds to the users' immediate needs. The adaptation team or handyman user communicates user needs to programmers for longer-term or more complex development.

3. HOW MGDS IS ADAPTED

3.1 The Handyman Example

MGDS functions are normally made available to projects as adaptable tools [Figure 2]. Users decide how to use the tools to do their tasks.

For example, the capability to process and display decommutated telemetry is provided by the Data Monitor and Display (DMD) tool. The Magellan spacecraft team used DMD to monitor real-time telemetry via the spacecraft team workstations [4]. Each spacecraft subsystem team had a DMD environment that was adapted to subsystem requirements and operator needs by a handyman user. Each spacecraft subsystem had different data display formats, processed different channels, and needed different scripts to alter the processing and display of the data.

Engineers defined their own display screens or templates by using a special template definition language (TDL). Although TDL is limited, it provided users with the tools required to create their own imaginative displays. Users could also produce derived telemetry channels via the channel conversion language. The MGN flight team developed 250 derived channels on their own.

organization-specific components. However, even with organization-specific standard components, as application programs become more numerous, or are enhanced, the open systems layered model will provide the framework for increased portability.

4.2 Primary Project Adaptation

The initial project adaptation is made by software developers and system integrators.

The developers ensure that the functional requirements of the project are met in the delivered tool suite. For instance, a developer modifies the DMD to process mission specific data types as part of the initial adaptation. Since these data types follow a baseline data format standard, the required changes are easily made by updating the DMD tables. Unanticipated adaptations will increase development costs and make the project adaptation more difficult.

The integrators test tools by writing scripts that set up end to end data processing to emulate mission scenarios.

The adaptable system design makes software development and integration easy because tools are built on common system components that rarely change.

4.3 Secondary Adaptation for Users and Teams

We found the adaptation team or handyman approach worked better than the turn-key approach for meeting mission operations requirements. The success of the adaptation rests on how well the team or handyman supports the users. Most of the handyman time must be spent in the missions support area, working with users at their workstations so that user environments are configured on user request with immediate feedback from users. Adaptation engineers are then able to inform developers of user problems, new mission requirements, and user requests that must be implemented in the future.

4.4 Is That It?

Designers will always miss something. That's why adaptable systems really succeed at the user level. Once the initial and secondary adaptations are made, the users will evolve the system into its most workable configuration for operations.

User adaptations can be as simple as grouping a set of tool commands into a script or as complex as writing a UNIX shell script to process data through several tools via named pipes. Although the users can make the system work for their tasks, they do not always choose the most productive methods [5]. Still, users often change it in ways that were never anticipated by the developers, integrators or adaptation engineers.

4.5 Adaptation is a Continuous Process

Managers must encourage a cooperative development and operational environment and define tangible system goals and benchmarks. Creating and maintaining a culture for adaptation is a major component of adaptable system development. Developers and users must all accept responsibility for the success of the adaptation. Feedback and interaction among developer, tester, and operator is indispensable. System teamwork is the key to solutions that are based on a better understanding of system-wide tradeoffs.

5. LOOKING AHEAD

The adaptable nature of MGDS will readily support new missions and operational scenarios. New missions will be able to use the tools provided by MGDS as part of their low-cost plan. Adaptations are more cost effective if missions consider and adopt MGDS baseline capabilities and standards prior to defining mission unique requirements.

5.1 Lessons Learned

•The handyman approach works better than the turn-key approach.

•The MGDS design is a good approach for creating a powerful, flexible, open system for mission operations.

•The adaptation process is only as good as the communication between operations and development organizations. Management must encourage teamwork and provide a structure for making system tradeoffs.

•Adaptable systems work best with experienced users who are eager to migrate the system into optimal mission operations. The users are the key to success and should not be automated out of the system. Rather, the users should automate the system to better serve them and science.

6. ACKNOWLEDGMENTS

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. The authors would like to thank Scott Burleigh, Steve Jenkins and Mike Tankenson for their valuable comments on earlier drafts of this paper and Robyn Lefler for producing the figures.

7. REFERENCES

1. SFOC[1] System Engineering, SFOC System Functional Design Document. *JPL Internal, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109,* November 1986.

2. A.W. Bucher. Using SFOC to Fly the Magellan Venus Mapping Mission. *Magellan Forum Series, JPL Internal, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109,* July 1992.

3. A.J. Gainsborough, J.A. Holladay, P.H. Ray. A Distributed Computer Environment for a Multimission Spacecraft Operations and Control Center. *Proceedings of ESA Symposium: Ground Data Systems for Spacecraft Control,* Darnstadt, FRG, 26-29, June 1990.

4. A. MacLean, K. Carter, L. Lovstrand and T. Moran. User-Tailorable Systems: Pressing the Issues with Buttons. *Human Factors in Computing Systems, CHI'90 Conference Proceedings (Seattle, WA),* ACM, New York, 1990, pp. 175-182.

5. E. Nilsen, H. Jong. Method Engineering: From Data to Model to Practice. *Human Factors in Computing Systems, CHI '92 Conference Proceedings (Monterey, CA)* , ACM, New York, 1992, pp. 313-319.

6. Technical Committee on Operating Systems and Application Environments of the IEEE Computer Society. Standards Project Draft Guide to the POSIX Open Systems Environment, P1003.0. *Institute of Electrical and Electronics Engineers, Inc.* New York, 1991.

[1] The Advanced Multimission Operations System (AMMOS) was formerly named the Space Flight Operations Center (SFOC).