Ground Systems Standardization and Commonality: Continuing the Dialogue

Deane Sibol
JHU/APL Space Department
Ground Applications Group

GSAW 2008

1-3 April 2008

The Johns Hopkins University
APPLIED PHYSICS LABORATORY

Presentation Outline

- Introduction
- Overview of Common Ground Approach
- Common Ground on MESSENGER, STEREO and New Horizons
- Benefits
- Factors in Achieving Cost-Savings
- Lessons Learned
- Standardization of Interfaces
- Management
- Summary

Introduction

- JHU/APL, located in Laurel, MD, USA
- Developed and currently operate 4 NASA Missions:
  - TIMED (LEO)
  - MESSENGER (DS) en route to Mercury
  - STEREO (DS) (2 observatories) observing coronal mass ejections
  - New Horizons (DS) en route to Pluto
- Developing NASA’s Radiation Belt Storm Probes (RBSP)
- MESSENGER, STEREO and New Horizons were the first missions to develop their ground software using the Common Ground Approach. TIMED preceded the Common Ground Approach, but its ground software did contribute to the common ground software code base.

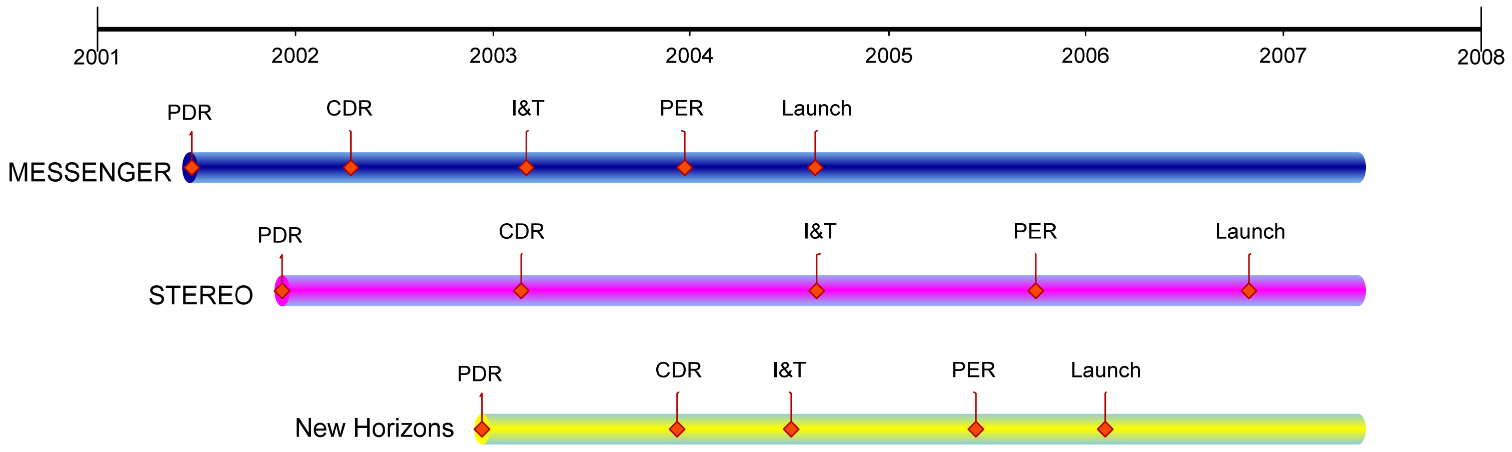
Overview of Common Ground Approach

- Organizational restructuring aligned with Ground Software CSCIs:
  - Common Ground Software Lead Engineer
  - 5 CSCI Leads for Commanding, Telemetry, Assessment, Planning, and Tools

- Primary responsibility for all leads was to identify commonality among missions and lobby for decisions that would keep ground software mission-independent, when feasible

- Requirements, design elements and code modules were each identified as mission-specific or mission-independent

Common Ground on MESSENGER, STEREO and New Horizons



- **All 3 missions equally shared in the initial cost of developing the requirements, architecture, design, implementation and testing of the Common Ground software.**

Benefit: Faster Time-to-Market

- Hardware-checkout GSW development from 8 SM to 2 SM
- All mission-specific requirements, design and code are clearly identified, so developers can quickly plan necessary modifications
- Subsequent GSW releases became easier to estimate and produce
- GSW never entered the mission schedule’s critical path

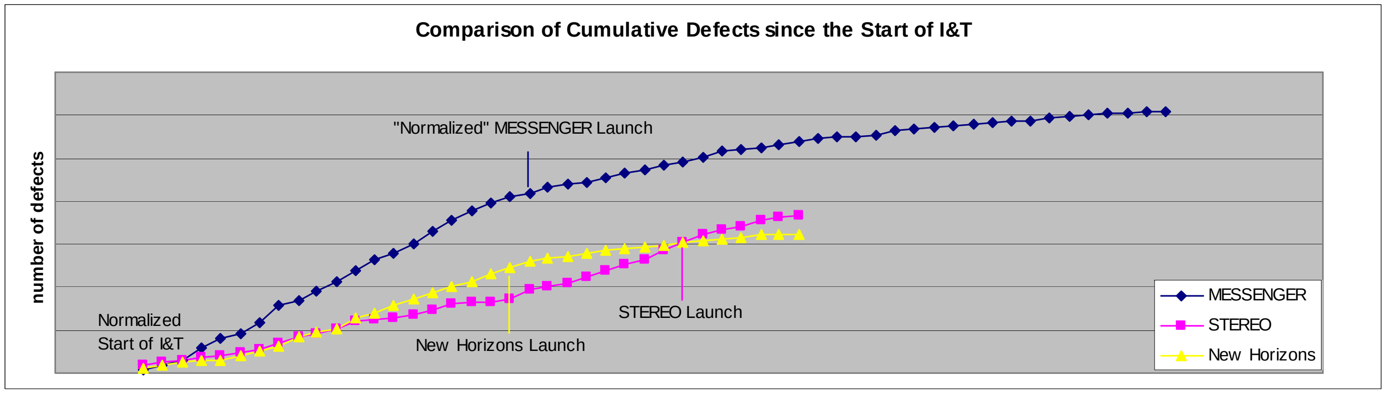
Benefit: Economies of Scale

- 3 missions contributed equally to the development of the architecture, component designs and the coding and testing of mission-independent software

- 3 missions benefited from software updates containing fixes and enhancements reported by any mission

- Total cost for maintenance of mission-independent software was less per mission

Benefit: Decrease in Overall Number of Defects



- **Defects reported by subsequent missions (STEREO and New Horizons) substantially decreased**
- **Concurrent use of mission-independent software yields reliable applications**
- **Fewer defects = fewer work-arounds, reduced down-time during critical spacecraft development phases**

Factors in Achieving Cost-Savings

- **60-70% Mission-Independent Requirements**
- **“Common” Design, with some mission-specifics configurable at runtime**
- **Of 200K SLOC, ~80% Mission-Independent Code**
- **Re-use of some test plans and test cases**
- **Reduced learning curve for testers and users**
- **“Common” Documentation**

Lessons Learned: Common CCB

- Mission GSW is not a COTS product, but a mission asset
- Mission CCB dictates changes
- Without a "Common CCB", Common Ground Approach relied on concurrence among 3 independent CCB
- Common Ground Approach also assumed that once a code freeze was lifted, missions would eventually sync to the latest-and-greatest software
- Branching is possible, but benefits and cost-savings are lost

- Even when a CCB insists on branching, mission operators continue to later ask for tools and features deployed to other missions other than their own.

- **A common CCB, much like that used by vendors of COTS products, is necessary to maximize commonality and aids in standardizing concepts of operations, requirements and tool features.**

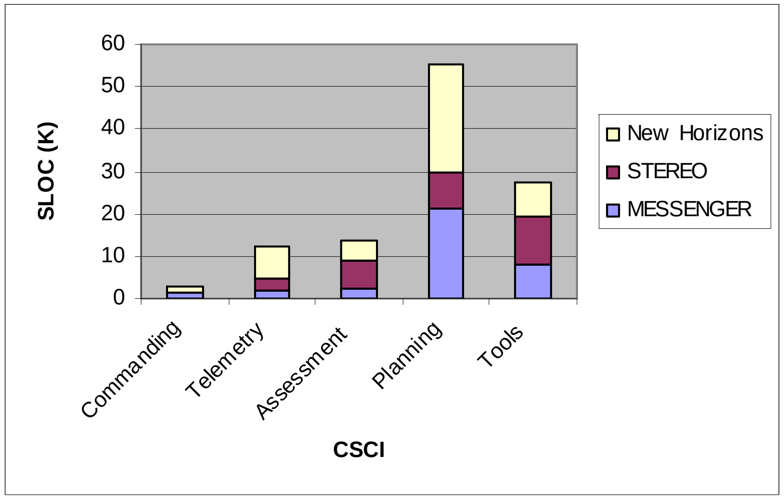
Lessons Learned: Identifying Mission-Independent Attributes

- **Impossible to predict future mission deviations**
- **Overestimating mission-independence: functionality must be inherited and overridden with mission-specific implementations**
- **Underestimating mission-independence: increased lines of code marked mission-specific, yet unchanged from mission to mission**

- **Planning for commonality in the initial development cannot be 100% accurate, but is the best approach.**
- **Creating commonality in later stages of development or after the fact causes more rework than if initially planned in – especially if the designs of the applications/systems were independent and diverse.**

Lessons Learned: Obstacles to a Fully Mission-Independent Ground Software

- Planning CSCI contains the most mission-specific code
- Mission-specifics already factored into:
  - Run-time configuration files or command-line arguments
  - Mission-specific constants in header files
  - Mission-specific derived classes



Standardization of Interfaces: Current Status

- In Use
  - CCSDS Telemetry – Version 1 Transfer Frame (not yet Version 2 VCDU)
  - CCSDS Commanding
  - CCSDS File Delivery Protocol (CDFP)
  - CCSDS Space Link Extension (SLE) Forward CLTU and RAF/RCF

- Considered / Watching
  - XML Telemetric and Command Exchange (XTCE) - no driver for portable format with I&T and Operations in-house
  - Cross-Support: Ground Station Monitoring and Control Interfaces – emerging
  - Cross-Support: Tracking and Orbit Propagation Services
    - Tracking Data Message (TDM) – beginning to be used by GS providers, need OTS tools to support as format option
    - Orbit Data Message (OEM) – beginning to be used by GS providers, need OTS tools to support as format option
  - Cross-Support: Ground Station Planning Interfaces – no standard

Standardization of Interfaces: Overcoming Obstacles

- Rarely get buy-in on use of standards until the standard is widely adopted, is mission-enabling, or is required via an interface agreement
- Perceived cost and risk of implementing/integrating a new standard usually outweighs any benefit

- Drive cost and risk down by:
  - Publishing standards with more than one reference implementation
  - Supplying simulators for each end of the interface
  - Teaming with other organizations to share these costs and risks

Management: Costing Future Missions

- Flight software's design and code re-use has direct impact of ground software re-use
- Focus cost estimation on known mission-specifics and assume mission-independent design and code are re-used with no cost
- Include 'tax' to bring current baseline forward, so that it does not remain stale

Management: Best Practices

- Get mission system engineer and program management buy-in to support commonality and standardization early and retain through management changes

- Enlist support of Missions Manager and Chief Engineer(s) having no direct mission responsibility

- Manage vendors and subcontracts separately per mission, but coordinate centrally

Summary

- Common Ground Approach to maximize commonality was successful in reducing cost and time-to-market
  - By identifying requirements and code that is mission-specific, making it clear to developers what needed to be changed from one mission to the next
  - By naming leads that were responsible for maintaining commonality among missions, when feasible
  - By reducing mission-independent defects

- Adoption of standards also aided in maximizing re-use among missions
  - Standardized Planning and Ground Station Interfaces (beyond Command and Telemetry) are needed

- Common CCB and support from non-mission managers required for success

References

- ***"Complete Ground Software Re-Use: The Common Ground Approach to a Re-Usable, Shared Ground System", Proceedings of the 5th International Symposium on Reducing the Cost of Spacecraft Ground Systems and Operations (RCSGSO), Paper 103-A0052, July 2003.***

- ***While the preceding paper proposed the Common Ground Approach and predicted cost-savings, the following paper…***

  "Software Re-Use Review: The Benefits and Future of the Common ground Approach", D. Sibol, P. McKerracher, R.M. Furrow, W. Stratton, paper and presentation to the 7th International Symposium on Reducing the Costs of Spacecraft Ground Systems and Operations (RCSGSO), Moscow, Russia, June, 2007.

  reports the results and lessons learned using 4 years of metrics.