Developing a Telemetry Archiving Architecture to Support Multiple, Simultaneous Missions

Leeha Herrera
The Johns Hopkins Applied Physics Laboratory
GSAW2004
Introduction

- It is possible to support multiple missions with a single telemetry archiving architecture
  - Archive: repository for long-term storage of telemetry in packet format

- Development costs are shared across missions

- Initial cost is significant but is then drastically reduced for subsequent missions
Overview

- Background
- High Level Architecture
- Benefits
- Conclusion
Response to Multiple Missions

- Resources were divided into common functional areas vs. the traditional mission-based teams to reduce redundancy
- Migrated to a Common Ground Software approach to support the simultaneous development of multiple NASA missions
  - Effort to develop applications that can be used on all missions w/ little to no customization
Migration of Legacy Code

• Common Ground Software code infrastructure was designed
  – used across all functional areas for all missions
• Inherited code evaluated for adaptation to support multiple missions in functionality and code structure
• Risk areas identified
  – Telemetry Archiving Architecture
Goals for the Archiving System

- Faster and consistent response and processing
- A robust and flexible system
  - Key for development for multiple missions
  - Allow mission specific modifications
  - Lower maintenance costs and support time
- Allow mission specific requirements and configurations
- Support wide variety of users: Developers, I&T, MOps, Science Centers
- Interface w/ EPOCH T&C
Key Design Decisions

• Provide 3 types of access to data
  – Real-time, Instant Playback, Long-term Playback
• Ability to “plug in” different processes to convert data from multiple sources and formats
• Long running, file based processing vs sockets
  – applications monitor input directories for new files
• Archive built from real-time and off-line file processing
• Access to available archive data while new data is added
System Data Flow

Ground Station

Data Router

Real-time Interface

Spool data to files

Instant Replay Interface

As Received Data Files

Sort & Create Index & Data Files (filter)

Sort & Create Index & Data Files (filter)

Sort & Create Index & Data Files

Merge new data w/ current archive

Assessment Processes

Playback Interface for archived data

Sorted Archive Files

Assessment Processes

Data Converter

Ground Station Data Files

Data Converter

Front End Data Files

Playback Interface

Real-time Data Client

Instant Replay Data Client

EPOCH

Data Files

Ground Station

Data Files

Sorterd Index & Data Files

Sorterd Index & Data Files

Spool data to files

Sorterd Index & Data Files

Playback Interface for archived data

Sorted Archive Files

Assessment Processes

March 2004
L Herrera
Component Based Architecture

• Allows for flexibility in configuration
  – Each mission can customize the system
  – Only necessary “plug in” components used

• Small components focused on a single function have proven to be easier to develop, test, and maintain

• Future components can be added to create additional layers of functionality
Archiving Architecture

• Seamless merging of data from multiple sources with configurable filtering capabilities
• Allows multiple points of access to data while processing it into the long term archive
• Archive files can be used directly as data products to science centers
• Ability to archive by ground receipt and/or spacecraft time
• Efficient processing & quick turn around of large amounts of data
Common Architecture

• Provides common look & feel to all users
  – Mission Operations personnel will be shared across missions

• Increases confidence in architecture and code
  – Tested on multiple missions
  – Fixes are shared across missions

• Allows large re-use, reduced time and cost for development of new missions
  – A telemetry archiving system can be brought on-line for a new mission in a week
Conclusion

• Lower development time & cost
  – core system developed w/ 1.5 resources in ~8 months
• Supports the needs of all current missions
  – All benefit from optimization/features required for one
• Experienced minimal user reported defects
• Changed the nature of developer support
  – Increase confidence in code due to smaller, robust processes
  – Complexity of configuration remains an issue
    • Most problems now reported due to configuration not application defects