Implementing Centralized Command Management in a Distributed Plug-and-Play Software Architecture

Christopher Monaco, Mark Reid
Johns Hopkins University, Applied Physics Laboratory
11100 Johns Hopkins Road, Laurel MD 20723
Christopher.Monaco@jhuapl.edu
Mark.Reid@jhuapl.edu
Contents

- Introduction
- STEREO Heritage Command Executive
- Plug-and-Play Concepts and NASA’s core Flight Executive (cFE)
- Side by Side Comparison of Architectures
- RBSP Prototype Command Manager
- Managing Distributed Command Execution
- Evaluation and Performance
- Conclusions
Introduction

- APL has developed spacecraft for multiple NASA missions
- Most recently launched STEREO (Oct 2006)
  - Twin Spacecraft providing stereo images of the Sun
- STEREO utilizes heritage APL Flight Software
  - TIMED, MESSENGER, New Horizons

- APL is currently defining the architecture for the RBSP mission
- Planned launch date (March 2012)
  - Twin Spacecraft investigating the Earth’s Radiation Belts
- Software team is evaluating Plug-and-Play concepts to facilitate software reuse
STEREO Heritage Command Executive
Plug-and-Play Concepts and NASA’s core Flight Executive (cFE)

**Publish/Subscribe:**
- Components communicate over a Standards-based Message-oriented Middleware/Software Bus.
- The Middleware/Software Bus uses a run-time Publish/Subscribe model. Message source has no knowledge of destination.
- No inherent component start up dependencies

**Benefits:**
- Minimizes interdependencies
- Supports HW and SW runtime “plug and play”
- Speeds development and integration.
- Enables dynamic component distribution and interconnection.

* Overview of Core Flight System (CFS), Wilmot/Bartholomew, Briefing to APL - Feb. 2006
Side by Side Comparison of Architectures

Legacy Single Application System

- Package 1
  - Function 1
  - Function 2
  - Function 3

- Package 2
  - Function 1
  - Function 2
  - Function 3

- Package 3
  - Function 1
  - Function 2
  - Function 3

Prototype Layered Plug-N-Play System

- Application A
  - Messaging Services
  - Core Services
  - OS Abstraction
  - Operating System

- Application B

- Application C
RBSP Prototype Command Manager

Command Manager (CM)
Task Context

- Autonomy Engine (AUT)
- Scheduler Application (SCH)
- Real-Time Command Ingest (CI)
- Time Tag Commanding (TT)
- <CM> Cmd
- <25 Hz Wakeup> Msg
- <CM> Cmd
- <App Ack/Nak> Msg
- <App> Cmd
- <App> Cmd
- <App> Cmd
- <CM Status> Tlm
- <App> Cmd
- Any Application

CM Application

- Macros
- <App> Cmds
- Cmd Buffers, Source and Pending Lists
- CM Status Telemetry
Managing Distributed Command Execution

- Ground
  - Real-time cmd (app cmd)
  - Package the Application Command within a Command Manager Packet and write it to the SW Bus

- CI Application
  - CM Pkt containing app cmd via SW Bus
  - Receive cmd, Extract app cmd, Link into cmd data structs, Locate highest priority cmd and send to SW Bus
  - App cmd to SW Bus
  - Execute cmd Issue Ack/Nak

- CM Application
  - Wait for next Scheduler Wakeup
  - no cmds
  - no cmds
  - no cmds
  - Wakeup

- Application
  - Ack/Nak via SW Bus
  - Process Ack/Nak, Update cmd history log
  - Wakeup
Evaluation and Performance

- CM architecture provides nearly the same command management functionality as existing software.
  - Prototype does not perform command verification at load time as traditionally done.
  - Command verification at load time can easily be included by extending the command protocol to include a “command verification” performed within the target application.

- Command throughput is dependent upon end application design.
  - Applications should not only wakeup upon SCH wakeup messages but also application commands.
  - With this design consideration command throughput equivalent to the CM application execution rate is attainable.

- Timing measurements indicate that the command data transmission overhead is small and not greatly affected by command packet size.
Conclusions

- CM prototype satisfies APL heritage architecture commanding requirements.
  - Maintains the familiar aspects of command execution: guaranteed command execution order and command prioritization.

- More flexible development environment with less coupling and less complexity.

- Application development can largely occur without the existence of the CM.

- Internal command data transmission overhead does not significantly impact processor utilization.
  - This architecture can easily support command rates equal to the heritage STEREO software without significant processing overhead.
Acknowledgments, Announcements, Questions

- We would like to recognize some of the people who have supported this effort
  - The STEREO Project
  - The RBSP Project
  - The APL Space Department Management
  - The APL Ground Applications Group (SIG)
  - NASA Goddard Space Flight Center, Flight Software Branch
  - NASA Goddard Space Flight Center, Technology Transfer Office

- We would also like to extend an invitation to the Spacecraft Flight Software Workshop hosted by JHU/APL and the AIAA
  - Dates: Nov. 5-6, 2007
  - http://www.flightsoftware.org/

- Questions?