GSAW2005 Tutorial C:

Mission Data System: A Unified Systems and Software Engineering Solution for Complex Control Systems

Length: Full Day

Overview:

As the challenges of space missions have grown over time, we have seen a steady trend toward greater automation, with a growing portion assumed by the spacecraft. This trend is accelerating rapidly, spurred by mounting complexity in mission objectives and the systems required to achieve them. In fact, the advent of truly self-directed space robots is not just an imminent possibility, but an economic necessity, if we are to continue our progress into space.

What is clear now, however, is that spacecraft system complexity is reaching a threshold where customary methods of control are no longer affordable or sufficiently reliable. At the heart of this problem are conventional approaches to systems and software engineering. Divide-and-conquer strategies based on subsystem-level functional decomposition fail to scale in the tangled web of interactions typically encountered in complex spacecraft designs, where subsystems are inevitably very tightly coupled through their need to share limited resources and perform robustly in uncertain environments. A straightforward extrapolation of past methods has neither the conceptual reach nor the analytical depth to address the challenges associated with future space exploration objectives.

Furthermore, there is a fundamental gap between the requirements on software specified by systems engineers and the implementation of these requirements by software engineers. Software engineers must perform the translation of requirements into software code, hoping to accurately capture the systems engineer’s understanding of the system behavior, which is not always specified explicitly or completely. This gap opens up the possibility for misinterpretation by the software engineer of the systems engineer’s intent, potentially leading to software errors. Similar problems can arise during operations.

These challenges have motivated the development of the Mission Data System (MDS), a unified software architecture for distributed, automated flight, ground, and test systems, and a novel systems engineering methodology, called State Analysis. State Analysis addresses these challenges by asserting the following basic principles.

Instructors: Robert Rasmussen, Michel Ingham, and Daniel Dvorak, Jet Propulsion Laboratory

Biography:

Robert Rasmussen, who holds a BS, MS, and Ph.D. in Electrical Engineering from Iowa State University, is a principal engineer in the Systems and Software division at JPL. He has been a member of the technical staff at JPL since 1975, gaining extensive experience in spacecraft attitude control and computer systems, test and flight operations, and automation and autonomy - particularly in the area of spacecraft fault protection. His flight mission adventures include systems engineering for Voyager and
Galileo, as well support to several other missions. Most recently, he was Cognizant Engineer for the Attitude and Articulation Control Subsystem on the Cassini mission to Saturn. Numerous awards for these efforts include NASA’s Exceptional Service and Exceptional Engineering Achievement Medals. He was also instrumental in initiating and advising the Remote Agent experiment on DS-1, an early demonstration of high level spacecraft autonomy, for which he shared NASA's Software of the Year award in 1999. He is currently chief architect of the Mission Data System, a new software and systems engineering framework.

Michel D. Ingham is a member of the Senior Technical Staff at the Caltech Jet Propulsion Laboratory, where he works in the Flight Software Development and Technology Group as a software systems engineer and software architect for the Mission Data System (MDS) project. He received his Sc.D. and S.M. degrees from MIT's Department of Aeronautics and Astronautics, and his B.Eng. in Honours Mechanical Engineering from McGill University, in Montreal, Canada. At JPL, Dr. Ingham is leading an integrated team of flight system engineers and software engineers, responsible for the end-to-end development of a software prototype for the entry, descent and landing system of the Mars Science Laboratory 2009 rover mission. As a member of the MDS software architecture team, his roles include coordination of the design and implementation of the Mission Planning and Execution (MPE) software system, participation in the design of ground and flight software architectures, and development of a state-based systems engineering methodology ("State Analysis"). Dr. Ingham is also the Principal Investigator of a joint JPL/MIT research project to integrate model-based execution technologies into the MDS software architecture.

Daniel Dvorak is a principal engineer in the Systems and Software Division at the Jet Propulsion Laboratory, California Institute of Technology. He is deputy architect for the Mission Data System (MDS) and Division Technologist for the Software and Systems Division. He worked on the Remote Agent experiment that flew on Deep Space 1. His interests include software architecture, autonomy and robotics, fault protection, and verification of autonomous systems. Previously he worked at AT&T Bell Laboratories on a number of projects including: R++, a rule-based extension to C++; a system for monitoring the health of 4ESS electronic switching systems (for toll telephone traffic); an expert system for performance monitoring and diagnosis of the UUCP network; and Datakit, a low-latency packet network. He holds a Ph.D. in computer science from The University of Texas at Austin, an MS in computer engineering from Stanford University, and a BS in electrical engineering from Rose-Hulman Institute of Technology.

**Who Should Attend:**

Systems engineers — especially those who deal with...
- The operational functionality of system hardware
- Integration of system functions
- Environmental effects on system performance
- The allocation and management of system resources
- Information flow throughout the system
- Software’s role in overall system behavior
- Validation of the system’s capabilities
• Modeling and simulation of the system, or
• Operation of the system

Software engineers — especially those...
• Negotiating or implementing functional requirements
• Integrating or testing software
• Dealing with hardware/software interfaces, or
• Acting as liaison to the greater project effort

Managers — especially those responsible for the...
• Delivery of software
• Operational behavior and performance of the system