Acquisition Strategies for Service-Oriented Architected Spacecraft Command and Control Systems

Geri Chaudhri
Geraldine.a.Chaudhri@aero.org
(310) 336-1091
The Aerospace Corporation
March 1, 2011
Introduction

• Trend for Service-Oriented Architecture (SOA) being embraced by space community

• CCSDS has published draft standards for a SOA based Spacecraft Command and Control Systems

• Programs such as GPS in process of acquiring a SOA based Command and Control System

• Challenges exist for the acquisition organization

• Strategies to address challenges
SOA Defined

• A paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains

• It provides a uniform means to offer, discover, interact with and use capabilities to produce the desired effects consistent with measurable preconditions and expectations

• In its simplest form, a Service-Oriented Architecture is a collection of services
  – Services are defined as intrinsically unassociated units of functionality, which have no calls to each other embedded in them
EXAMPLE SOA using Web Services

Spacecraft Command and Control Services

- Mission Planning
- Spacecraft Commanding
- Telemetry Processing
- Orbit Determination and Analysis
- Payload Management
- Simulation
- Procedure Development and Execution
- Data Analysis
Challenges

• Developing the Right Set of Requirements
  – Need to adapt requirements to available products and services

• Developing a System Architecture that is Compatible with Diverse Products
  – Large effort required to standardize product interfaces as well as keeping system secure

• Integrating Many Diverse Products
  – Allocating sufficient time and resources to this effort may require rework of existing scheduling models.
Strategies

• Organize the team around the products or services

Architecture

Commanding  Telemetry  Mission Planning  Orbit Determination  Data Analysis  Payload Management
Strategies

• Developing the “Right Set” of Requirements
  – Keep system requirements at a high level
  – Specifying too much detail in the system requirements limits the solution set and makes decomposing the requirements very difficult, causing redundancy at lower levels.
  – Prototype often to test out the feasibility of requirements with regard to goals, risk and stakeholder buy-in
  – Involve the end users in prototype demonstrations
  – Prioritize the requirements from most to least mission critical
  – Delay the establishment of the requirements baseline until the necessary prototyping is completed
Strategies

• Developing a System Architecture that is Compatible with Diverse Products
  – Designate a dedicated group to define or oversee the selection of the middleware or communication protocol
  – Describe the architecture in the simplest means possible
  – Define performance and security requirements up front
  – Perform trade studies while prototyping various architectures against defined criteria
  – Select products and vendors that can easily conform to selected architecture
  – Maximize use of COTS and/or reuse and legacy software when possible
Strategies

• Integrating Many Diverse Products
  – Allocate at least 50% of the development schedule to integration and test
  – Develop a formal plan for integration and test and adhere to it
  – Ensure that best practices and standards are used while executing integration and test
  – Use proven test methods and certified test equipment
  – Employ an independent test organization to verify the functionality
Software Acquisition – Capability Maturity Model*

• 5 levels
  • Initial – Ad hoc acquisition process
  • Repeatable – Processes are in place to plan, manage, evaluate and transition the product
  • Defined – Acquisition process is documented and standardized
  • Quantitative - Detailed measures of the acquisition processes and processes are collected
  • Optimizing - Continuous process improvement is fostered by quantitative feedback from the process

* Software Acquisition Capability Maturity Model (SA-CMM) Version 1.03
Copyright: Software Engineering Institute – Carnegie Mellon
Software Acquisition – Capability Maturity Model*

• Process Areas
  • Level 1 – Initial: None

  • Level 2 – Repeatable
    • Software Acquisition Planning
    • Solicitation
    • Requirements Development and Management
    • Project Management
    • Contract Tracking and Oversight
    • Evaluation
    • Transition to Support

* Software Acquisition Capability Maturity Model (SA-CMM) Version 1.03
Copyright: Software Engineering Institute – Carnegie Mellon
Software Acquisition – Capability Maturity Model*

• Process Areas (continued)
  • Level 3 – Defined:
    • Process Definition and Maintenance
    • User Requirements
    • Project Performance Management
    • Contract Performance Management
    • Acquisition Risk Management
    • Training Program Management

• Level 4 – Quantitative:
  • Quantitative Process Management
  • Quantitative Acquisition Management

• Level 5 – Optimizing:
  • Continuous Process Improvement
  • Acquisition Innovation Management

* Software Acquisition Capability Maturity Model (SA-CMM) Version 1.03
Copyright: Software Engineering Institute – Carnegie Mellon
Software Acquisition – Capability Maturity Model - Tailored

• If Level 2 add “Acquisition Risk Management” and “User Requirements”

• Merge “Requirements Development and Management” and “User Requirements” into one process area

• Inform end user of acquisition strategy

• Hold off on placing the requirements under change control until the necessary prototyping is done

• Performance and security requirements should be established early and placed under change control before the prototyping effort begins
Conclusions

• A SOA based system offers multiple benefits
  – Interoperability, reusability, expandability, maintainability
  – Cost reductions and improved service to users

• Challenges to acquisition organization can be overcome by:
  – Involving the end user early
  – Adapting requirements to products
  – Prototyping before baselining requirements
  – Selecting vendors that will bear the cost of standardization
  – Tailoring Software Acquisition – Capability Maturity Model