Enterprise Service Bus for Ground Systems Integration

Raytheon Intelligence and Information Systems

G Todd Kaiser
Chief Architect, Government Systems
303-344-6915
gtkaiser@raytheon.com

March 1, 2005
Agenda

- Introduction to Raytheon Aurora
- Current Industry Environment/Challenges
- Approaches & Technologies
- SOA/ESB Overview
- Migrating An Existing Telemetry System to an ESB
- The “Bigger” Picture
- Conclusions
Raytheon Space Systems - Aurora

Mission Management

Command and Control

Ground Terminals

Orbital Determination

Information and Network Management

Data Processing
Environment & Challenges

Industry Environment

• Various ground architectures
• Old and new technologies
• Multiple contractors with various products
• Redundant assets through stove-piped acquisitions
• Disparate products from single vendors
• New capabilities needed by customers “yesterday”
• Systems increasing in size and complexity

Challenges

Shorter Schedules
More Capabilities
Smaller Budgets

“Chief Architect”
Answering The Challenge

How do we tackle this?

We need an approach that includes standards, processes, and technologies used to leverage existing assets quickly, while still upholding quality and providing value.
Approaches & Technologies

Architectural Approaches
- Ad-hoc
- Structured
- Object-Oriented
- Distributed
- EAI/MOM
- SOA

Corresponding Technologies
- Integration Brokers
- ESB
- J2EE
- CORBA
- RPC
- Wheel
- Fire

Freshest “Buzz!”

3/18/2005 Page 6
Service-Oriented Architecture (SOA)

Goals

• Align information systems with mission goals
• Provide agile infrastructure that can change quickly as mission requirements change
• Provide a graceful (and self-paced) evolution for legacy systems

Key Aspects

• Services, NOT functions
• Separation of the interface from the implementation
• Business logic moved from the applications to the middle tier
• Services are published and discoverable

Example: NOAA’s National Digital Forecast Database
Enterprise Service Bus

Key Characteristics
- Transformation of data between different formats
- Messaging (usually through JMS, though not exclusively)
- Intelligent Routing
- Work-flow management (a.k.a., orchestration)
- Common data model
- Data synchronization

Integration Capabilities
- Standards-based
- Container-based service management
- Existing adaptors for legacy systems (CORBA, JMS, .NET, J2EE, etc.)
- On-the-fly data routing changes through work-flow management
- Transformation of data allows for integration of legacy services

Current Industry Best-Practice for Implementing an SOA
Migrating to an SOA with an ESB

**To Do List:**

1. **Determine Core Services**
2. **Build Higher Level Services**
3. **Model Business Processes using Services**

- **Core services** are atomic—they don’t depend on other services.
- **Higher level services** are composed of core services and are published to service directories.
- **Business processes** are realized using the new services.
A New Way Forward

• Transform existing assets into SOA services
  – Allows for reuse of existing investments
  – Provides SLAs that will allow aging technologies to be phased out on customer’s schedules and budgets

• Center new system design on SOAs with SLAs
  – System engineering starts building SLAs, not ICDs and IRSs, and thinks in terms of capabilities and benefits, not functions and features
  – Allows separate contractors to reuse their intellectual property in order to deliver a service to customers

• Use Enterprise Service Bus as integration technology
  – Characteristics of ESB allow for self-paced evolution of existing systems
  – Also allows rapid (and cheaper) reconfiguration of systems to meet changing mission needs
  – Standards-based integration technologies allow for greater collaboration amongst teammates – allowing systems to scale better through “divide-and-conquer” approach
An Example - Migrating Existing Telemetry System...

Key:
- Service Container
- Management Framework
- ESB Endpoint
- Client Interface
- Methods
- ESB Connection

Enterprise Service Bus

Decomm A  Decomm B  Engineering Conversion

Only 6 interfaces to manage!
The “Bigger” Picture

• Going beyond applications – integrate the entire enterprise
  – The puzzle is recursive – applications, missions, enterprises
  – Mission Planning, flight dynamics, telemetry analysis, factory test & support, mission users

• Mission-to-mission integration
  – Allow independently acquired and managed missions to collaborate by integrating them with ESB technology (standards-based, etc.)
  – Develop Service Level Agreements for each mission, NOT ICDs or IRSs
  – ESBs allow for security integration, geographically separated mission stations
  – Asynchronous messaging allows for messaging in transient environments such as the battlespace, air/space platforms, etc.
Conclusions

• The keys to making a service-oriented architecture work are as follows:
  – Change the mindset from traditional systems engineering of ICDs/IRSs to service level agreements.
  – Pick the right services – come up with a list of key characteristics for what defines a service and judge each possible service by this criteria.
  – Determine the granularity of a service for the given problem. Some services might be too fine-grained for some problem spaces.
  – Understand the limitations of the technologies. An ESB will not solve world hunger, but it should make getting to a solution easier.

• Standards, standards, standards
  – Community should start to standardize on data formats for messaging in a SOA-like architecture (e.g., a standard XML-based telemetry format)
Backup Slides
Raytheon Aurora
Delivered Ground Systems

Government Systems

Common Ground Systems

Commercial Systems

• Hughes Communications Galaxy USA
• AT&T - USA
• BrasilSat COCC Brazil
• Morelos 1 Mexico
• Solaridad Mexico
• Bsky - B UK
• OPTUS Australia
• Palapa B (Telkom) Indonesia
• Palapa C (Permutel) Indonesia

• THOR 1
• Norway, Finland, and England
• MEASAT 1
• Malaysia
• Sirius 1
• Sweden
• ASIASAT 1
• Hong Kong
• Astra
• Luxembourg
• APSTAR
• China
• BrasilSat B3
• Brazil
• MEASAT 2
• Malaysia
• THAICOM
• Thailand

• CHINASAT China
• JCSAT 4 & 5 Japan
• Superbird - C Japan
• Morelos 3 Mexico
• BSAT Japan
• THOR 2 Norway
• DCCS
• CCT
• Polar EHF

• Sirius 3 Sweden
• THOR 3 Norway
• PANAMSAT 5 US
• ASIASAT China
• Palapa Indonesia
• Orion Hawaii, Korea, and US
• AFSCN
• TDRSS

• Several Classified Systems

• ICO - USA
• ICO - Australia
• ICO - India
• ICO - Chile
• ICO - South Africa
• ICO - Germany
• ICO SCC - UK
• Bonum I Russia
• Thuraya UAE
• Superbird 4 Japan
• MCP
• STE
Migrating A Telemetry Subsystem

• Goal: Single, integrated ground system for various missions
  – Common API for services
• Telemetry
  – De-commutation of telemetry IS unique
  – Historical telemetry storage and retrieval IS NOT unique
  – Display of telemetry IS NOT unique
• Step 1: Core Services
  – Provide de-commutation service for each telemetry format
  – Provide a single telemetry display service (HMI integrated onto the ESB)
  – Provide a single history storage and retrieval service (tricky here – all one service or separate for real-time and off-line telemetry?)
  – Derived service: Telemetry distribution service
  – Develop a mission “agnostic” data format for telemetry – format accommodates all missions, but is not mission specific

Understand the capabilities required! Don’t expose services that are not needed!
Telemetry Subsystem (cont)

• Step 2: Higher-level services
  – Combine telemetry de-commutation, telemetry distribution, HMI, telemetry history into “real-time” telemetry service

• Step 3: Business Process Creation
  – Use intelligent message routing to route raw telemetry messages to correct de-commutation engine (messages tagged at source)
  – Use transformation services to transform raw telemetry formats into correct format for each de-commutation engine