Tools for Describing Space Data Systems Reference Architecture

19 May 2004

Peter Shames, NASA/JPL, Takahiro Yamada, JAXA/ISAS
Agenda

• Introduce the Reference Architecture for Space Data Systems (RASDS)

• Show some examples of how it can be used to model space data systems

• Define the RASDS requirements on formal methodologies & tools

• Describe our analysis of using SysML to provide the means to formally describe RASDS models
A Physical View of a Space Data System

One or More Spacecraft

Commodity Space Communication Systems
Commodity Space Navigation Systems

One or More Instruments

A Ground Tracking Network

A Spacecraft Control Center

A Space Tracking Network

An Instrument Control Center

A Science Facility

Source: A. Hooke, NASA/JPL

8/27/04

CCSDS Architecture WG 3
Reference Architecture

Purpose

• Establish an overall CCSDS approach to architecting and to developing domain specific architectures

• Define common language and representation so that challenges, requirements, and solutions in the area of space data systems can be readily communicated

• Provide a kit of architect’s tools that domain experts will use to construct many different complex space system architectures

• Facilitate development of standards in a consistent way so that any standard can be used with other appropriate standards in a system

• Present the standards developed by CCSDS in a systematic way so that their functionality, applicability, and interoperability may be clearly understood
Technical Approach

• Develop a methodology for describing systems, and systems of systems from several viewpoints
  – Initial focus was CCSDS, but it is more generally applicable to space data systems
  – Derived from Reference Model of Open Distributed processing (RM-ODP), which is ISO 10746
  – Adapted to meet requirements and constraints of space data systems

• Define the needed viewpoints for space data system architecture description
  – Does not specifically include all elements of RM-ODP engineering and technology views, assume use of RM-ODP for these
  – Does not encompass all aspects of Space Systems, i.e. power, propulsion, thermal, structure, does not preclude them either

• Define a representational methodology
  – Applicable throughout design & development lifecycle
  – Capture architecture & design artifacts in a machinable form, able to support analysis and even simulation of performance
  – Validate methodology by applying it to several existing CCSDS reference models and existing systems

• Identify relevant existing commercial methodologies
  – Evaluate UML 2.0 and SysML, now in progress
  – Explore applicability of selected methodology & tools to RASDS
Space Data System
Several Architectural Viewpoints

Enterprise
Business Concerns
Organizational perspective

Connectivity
Physical Concerns
Node & Link perspective

Functional
Computational Concerns
Functional composition

Information
Data Concerns
Relationships and transformations

Communications
Protocol Concerns
Communications stack perspective

Derived from: RM-ODP
8/27/04
CCSDS Architecture WG
Space Data System
Architectural Notation

Object

Object with Interface

Object Encapsulation

Node (physical location)

Node Encapsulation (physical aggregation)

Logical Link

Physical Link

Space Link (rf or optical)

Management

Service

External Concerns
Unified Object Representation

Management Interfaces:
How objects are configured controlled, and reported upon

Object

Service Interfaces:
How services are requested & supplied

Core Functions
What the object does

External Interfaces:
How external elements are controlled

Concerns:
Issues
Resources
Policies
Enterprise View
Federated Enterprises with Enterprise Objects

Agency ABC
Mission A
GTN B
Prog C
Enterprise Concerns:
Objectives
Roles
Policies
Activities
Configuration
Contracts
Lifecycle / Phases

Mars Exploration Program Federation
Cross-Support Agreement

Mission AX
GTN Y
Proj X
Service Z

Mission BFD
Development & Operations Domain

Company XYZ

Agency QRS
Mission Q
Proj R
Instr S
Instrument Integration

Organization PDQ
Mission BFD
Operations Contract
Functional View

Example Functional Objects & Interactions

- Monitor & Control
  - Directive Generation
  - Directive Management
  - Directive Execution
- Data Repository
- Data Acquisition
- Orbit Determination
- Radiometric Data Collection
- Tracking
- LT Data Repository
- Spacecraft Analysis
- Mission Analysis
- Mission Planning

Functional Concerns:

Behaviors
Interactions
Interfaces
Constraints
Connectivity View
Nodes & Links

Connectivity Concerns:
Distribution
Communication
Physical Environment
Behaviors
Constraints
Configuration
Connectivity & Functional View
Mapping Functions to Nodes

Science Spacecraft
- Monitor & Control
- Directive Execution
- Attitude Control
- Radiometric Data Collect
- Data Acquisition
- Data Repository
- Comm Mgmt
- Tracking

Science Institute
- Mission Planning
- Directive Generation
- Data Repository
- LT Data Repository (Archive)
- Directive Management
- Directive Generation

Tracking Station
- Radiometric Data Collect
- Monitoring Mgmt
- Tracking

S/C Control Center
- Spacecraft Analysis
- Mission Analysis
- Orbit Determ
- Traj Design
- Monitor & Control
- Directive Management
- Directive Generation
- Data Repository

Combined View:
End to End Behavior
Performance
Throughput
Trade studies

8/27/04
CCSDS Architecture WG
Information Objects
Relationship to Functional View

S/C Event Plans
Observation
Plans

Directive
Generation

Directive
Execution

Command
Execution

Operation Plans

Commands

Operations Plan
Schema &
Structure
Definition

Command
Schema &
Structure
Definition

Actual Data
Objects

Data Models

Abstract
Data Architecture
Meta-models

Information Objects
are exchanged among
Functional Objects

Information Concerns:
Structure
Semantics
Relationships
Permanence
Rules
Communications Viewpoint
Protocol Objects
End-To-End Command Processing

Communications Concerns:
- Standards
- Interfaces
- Protocols
- Technology
- Interoperability
- Suitability
Security Analyses
Multiple Viewpoints & Relationships

Enterprise Security Domains

Functional Allocations

Connectivity & Communications

Trust relationships
Policies
Privacy / proprietary issues

Access control
Authentication

Firewalls
Encryption
Boundary access points

Combined View:
Relationships
Allocations
Performance
Trade studies

Data Acquisition
Monitor & Control
Directive Execution
Data Management
Attitude Control
Radiometric Data Collect
Tracking

Enterprise Security Domains

Mission A
Spacecraft
Mission A
Instrument Control Center
Spacecraft Control Center C
Ground Tracking Network B

Combined View:
Relationships
Allocations
Performance
Trade studies

Enterprise Security Domains

Mission A
Spacecraft
Mission A
Instrument Control Center
Spacecraft Control Center C
Ground Tracking Network B

Connectivity & Communications

Science Spacecraft
Science Institute
Tracking Station
S/C Control Center

Combined View:
Relationships
Allocations
Performance
Trade studies

Data Acquisition
Monitor & Control
Directive Execution
Data Management
Attitude Control
Radiometric Data Collect
Tracking

Connectivity & Communications

Science Spacecraft
Science Institute
Tracking Station
S/C Control Center
High Level RASDS
Methodology / Tool Requirements

• Meta-model and model language that is independent of specific tool environments and implementations
  – Models can be exchanged and imported into other tool suites

• Tool suite with a graphical interface that enables creation, manipulation, display, archiving, and versioning of meta-models, component and connector type templates, and instance models

• Support development of machine readable, portable architecture meta-model for RASDS

• Support development of instance models for specific space systems deployments

• Provide a framework that supports coarse grained simulation of behavior and performance characteristics instance models
Formal Method Evaluation

• Studied UML 2.0, SysML, xADL
• Unified Modeling Language (UML 2.0)
  – Too focused on software systems
  – Includes elements that are not needed for RASDS
  – Some commercial tool support now
• System Modeling Language (SysML)
  – Has most of the required features (and more)
  – Needs some extensions for RASDS viewpoints and details
  – Commercial tools support expected late 2004 / early 2005
• xADL
  – Extensible approach that can accommodate RASDS
  – xADL needs to be customized, not interoperable w/ XMI
  – Tool support from UCI and USC, academic quality
SysML Background

- Informal partnership of modeling tool users, vendors, etc.
  - Organized in May 2003 to respond to UML for Systems Engineering RFP
  - Includes many aerospace companies and major UML tool vendors
- Charter
  - The SysML Partners are collaborating to define a modeling language for systems engineering applications, called Systems Modeling Language™ (SysML™). SysML will customize UML 2 to support the specification, analysis, design, verification and validation of complex systems that may include hardware, software, data, personnel, procedures, and facilities.
- References:
  - SysML Partners Web Site  http://www.sysml.org
  - See also SysML Specification Draft v0.3 on this web site

Source: SysML Partners
SysML Language Architecture

<<metamodel>>
SysML

UseCases

Interactions

Components

Classes

Auxiliary Constructs

Profiles

Actions

StateMachine

Activities

Parametrics

Requirements

Verification

Source: SysML Partners

8/27/04

CCSDS Architecture WG
Mapping RASDS into SysML

- No simple one for one mapping
- RASDS uses Viewpoints to expose different concerns of a single system
- SysML uses specific diagrams to capture system structure, behavior, parameters and requirements
- Several SysML diagrams, focused on different object classes, may be usefully applied to any given RASDS Viewpoint
- Extended SysML Views may be used to define the relationships between Viewpoints and Diagrams
- SysML will support more accurate fine grained modeling of structure, relationships and behavior than was expected of RASDS
Mapping RASDS into SysML

- **Enterprise**
  - Organizational component & collaboration diagrams
  - Use case, interaction overview diagrams
  - Requirements & constraints for rules, policies & agreements
- **Connectivity**
  - Physical component, composition, collaboration & class diagrams
  - Parametric diagram for physical link characterization
- **Functional**
  - Logical component, collaboration & class diagrams
  - Activity, state chart, parametric, & timing diagrams
- **Informational**
  - Information class & parametric diagrams
- **Communication**
  - Protocol component & collaboration diagrams
  - State machine, sequence, activity & timing diagrams
Enterprise View Using SysML
Use Case Diagram
Connectivity View (Nodes & Links) Using SysML Components (Spacecraft)

Derived from: SysML Partners
8/27/04
CCSDS Architecture WG 23
Connectivity View (Nodes & Links) Using SysML Components (MOS & TT&C Systems)

MOS : MissionOpsSystem
- dm : DataManager
- MOP : Mission OpsPlanning
- Cmnd Port
- SC : SendCmnd

ObsReq Port

TelemDonePort

TT&C : TrackTelemCommand
- TM : Telemetry
- TC : Telecommand
- ReTransport
- UL Port
- dl : DownLink
- ul : UpLink
- Ant: Mechanical

TelemPort

TelemData

CmndData

PointingData
Connectivity View (Composition)
Using SysML Components

Global structure inherited by each kind of Spacecraft …
… and constrained for each kind
Preliminary Functional View Using SysML Activity Diagram

- Showing component allocations (optional)
Informational View Using SysML Class Diagram

• Reusable, refinable information structure:

```
<<info obj>>
InstrCmdnFile
```

```
<<data obj>>
InstrCmdnList
```

```
<<data obj>>
InstrCmdn
```

```
<<metadata obj>>
InstrCmdnMetaData
```

```
<<metadata obj>>
InstrCmdnSemantics
```

```
<<metadata obj>>
InstrCmdnStructure
```

Global representation inherited by each kind of Information Object
Communication View (Protocol Objects) Using SysML Component Diagram

MOS : MissionOpsSystem

SC : SendCmd

MOP : Mission OpsPlanning

<protocol>
TP: TransportLayer

<protocol>
IP: InternetLayer

<protocol>
LL: LinkLayer

<protocol>
PL: PhysLayer

RF: link [Xband]

SC : SpaceCraft

ECU : Execution Control Unit

TC : TeleCmd

<protocol>
TP: TransportLayer

<protocol>
IP: InternetLayer

<protocol>
LL: LinkLayer

<protocol>
PL: PhysLayer

CmndData:RF

<<controls>>

Derived from: SysML Partners
8/27/04
CCSDS Architecture WG
Communication View Using SysML State Machine Diagram

<<protocol>>
TP: TransportLayer

Idle

Sending

Waiting

evSend / transmitCount=0

evDoneSend / ++transmitCount

evACK[isValid]

tm(Wait Time) Throw("Unable to Send")

tm(Wait Time) [transmitCount<LIMIT]

Protocol specifications inherited by each instance of Protocol Objects
Acknowledgements

• This task was carried out as part of the program of work of Consultative Committee for Space Data Systems (CCSDS).
• It was performed by the Architecture Working group (AWG), chaired by Takahiro Yamada, ISAS
• Other AWG members who actively participated are listed below:

  – Fred Brosi, NASA/GST
  – Dan Crichton, NASA/JPL
  – Adrian Hooke, NASA/JPL
  – Steve Hughes, NASA/JPL
  – Niklas Lindman, ESA/ESOC
  – Nestor Peccia, ESA/ESOC
  – Lou Reich, NASA/CSC
  – Don Sawyer, NASA/GSFC
  – Peter Shames, NASA/JPL
  – Anthony Walsh, ESA/Vega
BACKUP SLIDES
Enterprise View (Enterprise Objects)
Connectivity View (Nodes and Links)

Node A

Node B

Node C

Link 1 (Physical Connection)  Link 2 (Physical Connection)
Functional View (Functional Objects)

Connectivity + Functional View (Nodes, Links and Functional Objects)
Information View (Information Objects)

- Abstract Data Architecture
- Meta-models
- Defined Data Models
- Actual Data Objects

Functional+Information View (Functional Objects and Information Objects)

- Application A
  - Info Mgmt A
  - Data A
- Application B
  - Info Mgmt B
  - Data B
- Application C
  - Info Mgmt C
  - Data C
Connectivity+Functional+Communication View (Nodes, Links, Functional Objects and Communications Objects)
SysML Motivation

• Systems Engineers need a standard language for analyzing, specifying, designing, verifying and validating systems

• Many different modeling techniques
  – Behavior diagrams, IDEF0, N2 charts, …

• Lack broad based standard that supports general purpose systems modeling needs
  – satisfies broad set of modeling requirements (behavior, structure, performance, …)
  – integrates with other disciplines (SW, HW, ..)
  – scalable
  – adaptable to different SE domains
  – supported by multiple tools

Source: SysML Partners
UML 2.0 / SysML
Architectural Alignment

<<metamodel>>
UML

Common Behaviors
UseCases
Activities
Actions
StateMachines
Interactions
Composite Structures
Components
Deployments

<<metamodel>>
SysML

UseCases
Activities
Actions
StateMachines
Interactions
Components
Parametrics
Requirements
Verification

Classes
Profiles
Auxiliary Constructs

Source: SysML Partners
8/27/04
Analysis of Using SysML for RASDS

• Analyzed requirements in UML for Systems Engineering RFP and SysML Draft Response (January 25, 2004)

• Initial analysis indicates that SysML meets or exceeds the requirements for RASDS, with some specific exceptions:
  – The ability to explicitly relate model elements in multiple model viewpoints is partially addressed by SysML
  – Must be augmented by RASDS methodology specific views, relationships and constraints.

• **SysML specification is still being finalized**, elements are expected to further evolve before completion. SysML is expected to be adopted by the OMG & INCOSE in late 2004, tool support will follow shortly.

• SysML Partners view RASDS exercise as a useful set of test cases to validate their approach, has driven evolution of some elements (views)
Further Considerations on Use of SysML

- Viewpoint in SysML is a constrained set of elements from *meta-model* selected for a particular purpose.
- View in SysML is a constrained set of elements selected from a *user model* for a particular purpose.
- These align with RASDS usage, but details of specifying RASDS views are yet to be finalized.
- The behavior and executability aspects of SysML are outside current RASDS scope, but are expected to prove useful.
- Requirements and parametric diagrams are not currently required for RASDS, but are likely to be useful in the long run.
- SysML provides behavioral specifications in state charts and activity diagrams, but some model behavior specifications may require use of some TBD script language.
- SysML is expected to support evolution of mission designs via elaboration of successive levels of detail in a model.
Derived from: SysML Partners

Function 1

<<allocatedTo>>

<<logical>>
Logical Comp 1
Operation 1

<<allocatedTo>>

<<logical>>
Logical Comp 2
Operation 2

<<allocatedTo>>

<<physical>>
Phys Comp A
Logical Comp 1
Logical Comp 2

Functional – Logical – Physical Allocation: Viewpoint Relationships
Next Steps

• Validate SysML modeling approach
  – Complete analysis of RASDS to SysML mapping
  – Validate with SysML Partners
  – Seek concurrence with CCSDS SAWG community

*IFF agreed, then:*

• Adopt an agreed RASDS formalism
  – Select specific formal methods from SysML for describing RASDS architectures and systems
  – Agree to final common representation and methods

• Generate baseline RASDS approach
  – Develop agreed SysML meta-models for Viewpoints
  – Define extensible library of component instances
Enterprise View Using SysML
Class Diagram

• Mission Organization Options:

- Mission_Ops_Team
- Space_Asset
- Gnd_Tracking_Facility
- Rover
- Orbiter
- Govt Network
- Commercial Network

Derived from: SysML Partners
8/27/04
CCSDS Architecture WG
1. **Support Architectural Modeling** – provide means for developing, validating, extending, and sharing RASDS compliant models

2. **Flexibility** – allow multiple approaches to be explored at the same time

3. **Model Integrity** – provide means for ensuring model integrity by checking relationships across views and updating them automatically (or flagging problems)

4. **Model Validation** – provide means for validating model completeness and well formedness

5. **Relative Ease of Use** – exhibit good ergonomics, be easy to learn and use, and provide other ease of use features like contextual help

6. **Repository / Model Sharing** – provide means for storing complete models, model elements, fragments, and templates, and for sharing these across a working group. Facilitate re-use and sharing
RASDS Requirements, contd

• Other Considerations:

  – Selected set of mission / space systems deployments must be developed and agreed
  – Mission lifecycle views, concept, design, development, launch, operation
  – Architectural model lifecycle, abstract to concrete, relationship to design
  – Extracting "suitable for framing" viewpoints for different audiences from models
  – Development of prototypes of various architecture elements and approaches
  – Explore means to do trade space evaluation driven by architecture models