Integration Analysis via Software Architecture Verification A Multi-Dimensional Approach

Presented by:
Dan Sivertson

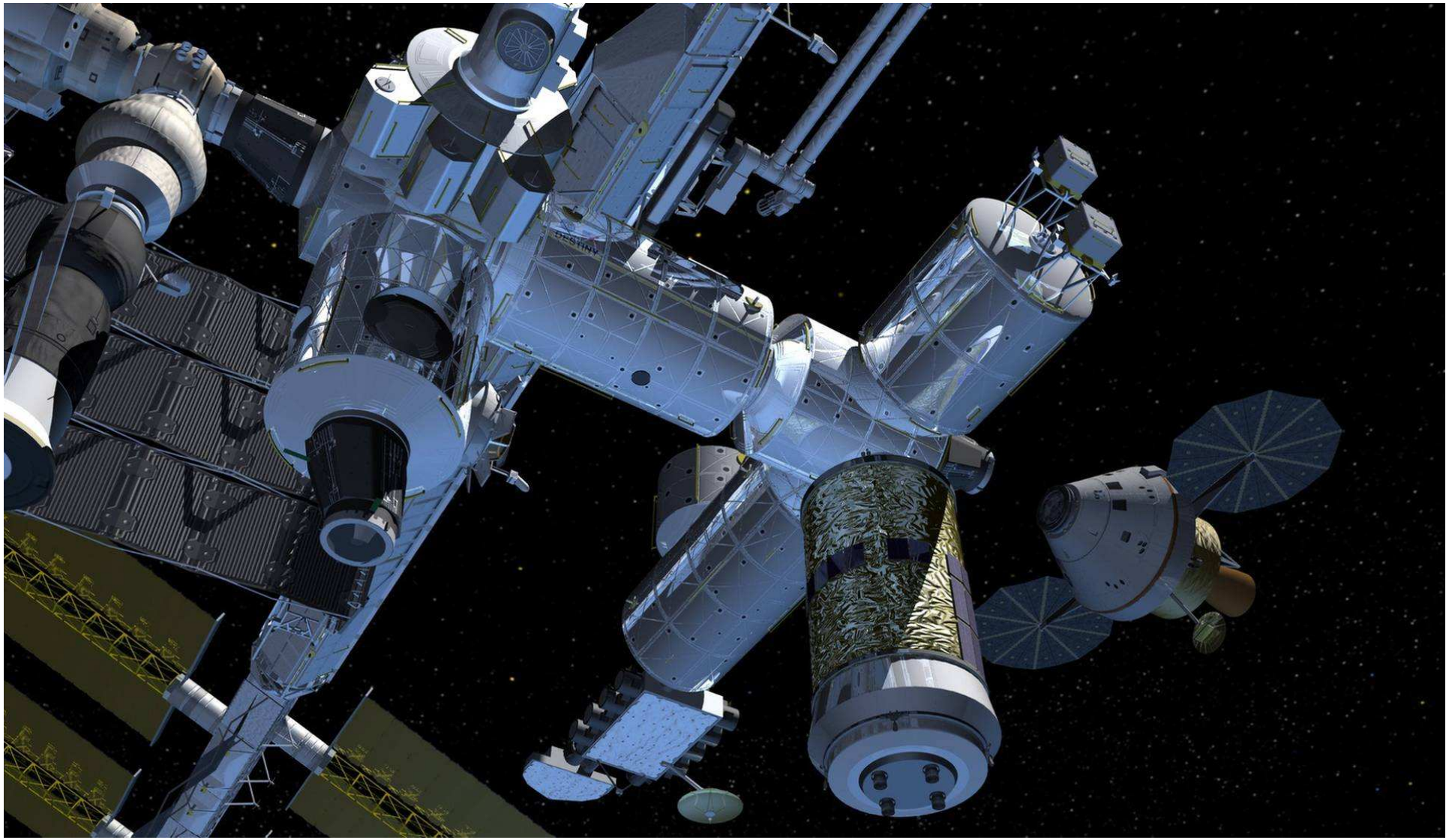
NASA IV&V Facility
Fairmont, WV

16 Sep 2010

Overview

- Orion Crew Exploration Vehicle
- NASA Software IV&V Technical Framework excerpts pertaining to Software Architecture/Design verification and our focus on standards, threads, and traces as a means of addressing architecture verification.
- A quick overview of software architecture, architecture descriptions, and DoDAF 2.0
- A run down of our "Orion Risk #34 – Inadequate Software Architecture" against DoDAF 2.0 dimensions as a set of observations
- Results of submitting our Orion Risk #34 at the Orion SW PDR.
- Addressing Architecture Risk
- Essential Architecture Verification Properties – laying the groundwork for our prototype Event Integration Analysis method
- Software Architecture Verification Matrix – Analysis Dimensions vs. Essential Properties and their coverage by Event Integration Analysis
- Event Integration Analysis method overview
- Preliminary Event Integration Analysis Results
- Conclusions

Orion Crew Exploration Vehicle



Orion's avionics suite will embody complex software behaviors

NASA Software IV&V Technical Framework Q&A

- How do we know that:
  - The system architecture contains necessary items to carry out mission and satisfy user needs?

Standards

- A: Use architecture standards as means to judge whether the necessary views exist in the software design – allowing the project to succeed.

  - The system's software requirements are high quality (correct, consistent, complete, accurate, readable, and testable), and will adequately meet the needs of the system and expectations of its customers and users, considering its operational environment under nominal and off-nominal conditions, and that no unintended features are introduced?

Threads

- A: Examine integrated threads of execution across the system to ensure than nothing is missing including redundant paths as required for mission critical scenarios.

  - The complete, integrated system complies with its specified system requirements allocated to software and to validate whether the system meets its original objectives?

Traces

- A: Trace software components through threads of execution to ensure that the correct components are in place and that the system will meet performance requirements. Also, trace requirements, architecture, and design artifacts through these threads of execution to ensure compliance with the original objectives.

Standards

- IEEE 1471-2000
  - Systems and software engineering – Recommended practice for architectural description of software-intensive systems
- DoDAF 2.0
  - DoD Architecture Framework 2.0

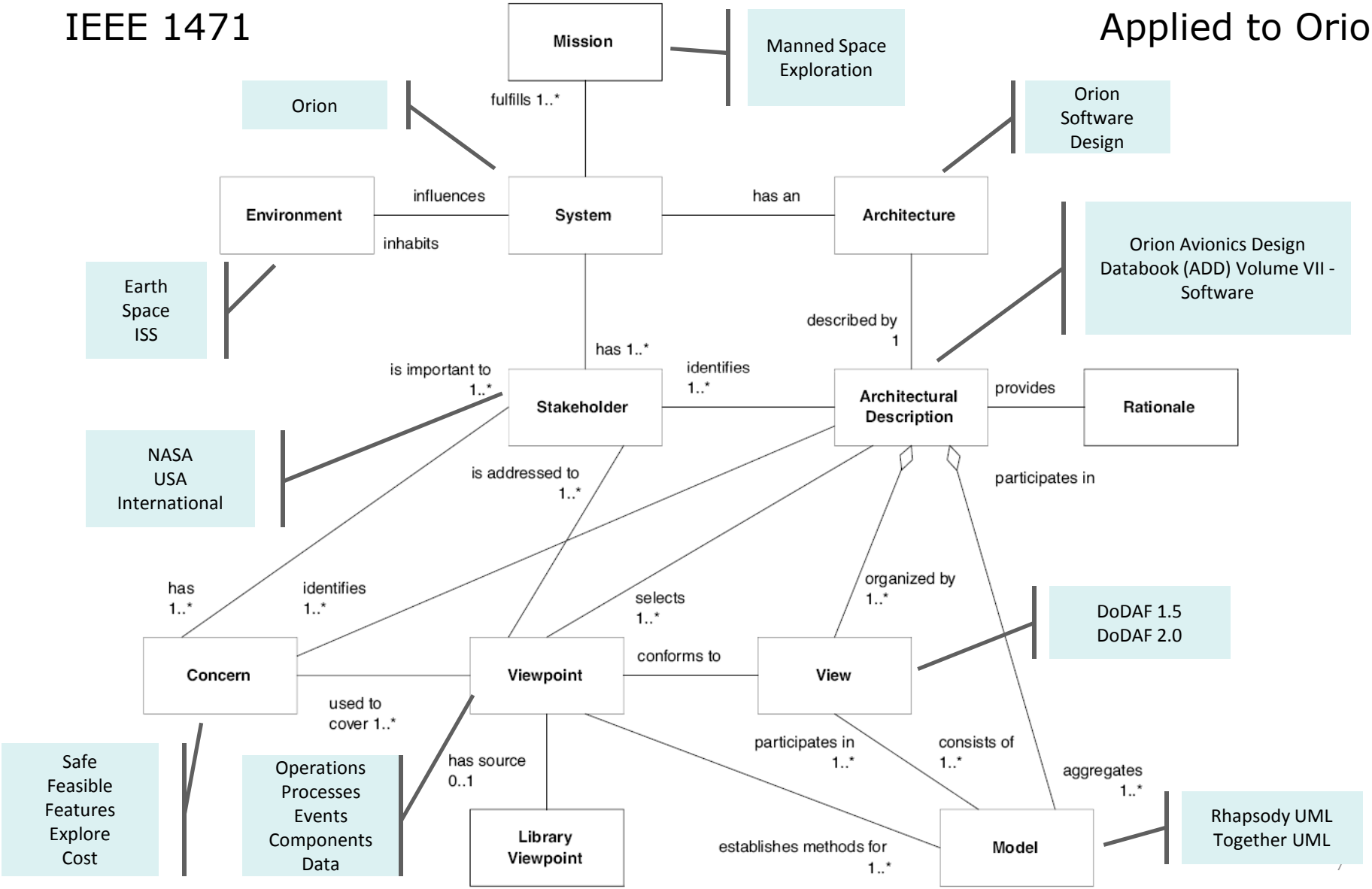
What is software architecture?

- IEEE Standard Glossary of Software Engineering Terminology:
  - *architecture*: The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution.
  - *architectural description* (AD): A collection of products to document an architecture.
  - *view*: A representation of a whole system from the perspective of a related set of concerns.
  - *viewpoint*: A specification of the conventions for constructing and using a view. A pattern or template from which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis.

Conceptual Model of an Architectural Description

IEEE 1471

Applied to Orion



DoDAF 2.0 Viewpoints

Project Viewpoint
Describes the relationships between operational and capability requirements and the various projects being implemented; Details dependencies between capability management and the Defense Acquisition System process.

Capability Viewpoint
Articulate the capability requirement, delivery timing, and deployed capability

Operational Viewpoint
Articulate operational scenarios, processes, activities & requirements

Services Viewpoint
Articulate the performers, activities, services, and their exchanges providing for, or supporting, DoD functions

Systems Viewpoint
Articulate the legacy systems or independent systems, their composition, interconnectivity, and context providing for, or supporting, DoD functions

Standards Viewpoint
Articulate applicable Operational, Business, Technical, and Industry policy, standards, guidance, constraints, and forecasts

Data and Information Viewpoint
Articulate the data relationships and alignment structures in the architecture content

All Viewpoint
Overarching aspects of architecture context that relate to all views

Legend: = Orion Architecture Risk Areas

Orion Risk #34 – Inadequate SW Architecture

Areas of Weakness vs. DoDAF 2.0 Views

DIV-1:Conceptual Data Model	The required high level data concepts and their relationships.
DIV-2: Logical Data Model	The documentation of the data requirements and structural business process (activity) rules. In DoDAF V1.5, this was the OV-7.
DIV-3: Physical Data Model	The physical implementation format of the Logical Data Model entities, e.g., message formats, file structures, physical schema. In DoDAF V1.5, this was the SV-11.

Observation: While the physical data models delivered for Orion PDR were strong, there was virtually no conceptual or logical data model tying Orion into the Constellation system data exchanges. Orion is a data-driven spacecraft – reconfigurable via configuration files. A strong data architecture is just as important as a sound software architecture.

Orion Risk #34 – Inadequate SW Architecture

OV-5a: Operational Activity Decomposition Tree	The capabilities and activities (operational activities) organized in an hierarchal structure.
OV-5b: Operational Activity Model	The context of capabilities and activities (operational activities) and their relationships among activities, inputs, and outputs; Additional data can show cost, performers or other pertinent information.
SvcV-2 Services Resource Flow Description	A description of resource flows exchanged between services.

Observation: Many low-level use cases and activity diagrams are being developed for Orion, however higher-level operational activity diagrams from a crew or ground-ops viewpoint are missing. This is critical for defining sound threads of operation for Orion command and control.

Orion Risk #34 – Inadequate SW Architecture

OV-6b: State Transition Description	One of three models used to describe operational activity (activity). It identifies business process (activity) responses to events (usually, very short activities).
OV-6c: Event-Trace Description	One of three models used to describe operational activity (activity). It traces actions in a scenario or sequence of events.

Observation: Clear, clean state transition descriptions and event-trace descriptions for critical non-recoverable Orion events such as Crew Module / Service Module separation via major software components are lacking.

Orion Risk #34 – Inadequate SW Architecture

SV-6 Systems Resource Flow Matrix	Provides details of system resource flow elements being exchanged between systems and the attributes of that exchange.

Observation: While interfaces have been defined between Orion partitions, the actual data flowing across these interfaces remains largely undefined as well as the rates of data transfer. Without such information, it is difficult to predict whether Orion processors will meet their throughput constraints or whether they will become I/O-bound.

Orion Risk #34 - Results

- Orion Risk #34 was the #1 Action Item at Orion Software PDR
- Orion developer responded quickly with a significant upgrade to the Orion Avionics Design Databook (ADD) Volume 7 – Software Architecture
- While the ADD upgrade still falls short in the DoDAF views mentioned in this presentation, it was a great improvement over the previous ADD version.
- Orion developer should deliver an update to the ADD Volume 7 for Orion Software CDR filling in many gaps in the current document.
- Risk has gone from red to yellow
- The PDR Action Item is now closed at the by Orion Avionics, Power, and Software Office

Addressing Architecture Risk

- In light of Orion Risk #34, our Orion IV&V team decided to try some new approaches for “Integration Analysis” across key Orion threads.
- What follows is a breakdown of what one of our architecture verification task teams is doing to further investigate critical non-recoverable events within Orion as “Integration Threads”
- This is a prototype method that is under trial right now on the Orion IV&V team.

SW Architecture Verification - Essential Properties

- What is the software supposed to do?
  - Specification – the nominal path - the system capabilities
- What is the software not supposed to do?
  - Unexpected emergent behavior – boundary specification – guards and inhibits
- What is the software supposed to do under adverse conditions?
  - FDIR – system biases – tolerances – off-nominal paths
- What are the safety considerations?
  - Crew – mission – public
- What are the integration or interfacing considerations?
  - System of systems – system – module – CSCI – component – HW/SW – partitions – network – parallelism – compatibility
- What are the dependability considerations?
  - Performance – capacity – complexity – stability

Standard IV&V questions plus safety, integration, and dependability

SW Architecture Verification – Examples 1

Analysis Dimensions	Intended behavior	Unacceptable Behavior	Adverse Conditions	Safety	Integration	Dependability
Interface Analysis	Control algorithms, protocols, data content/format, performance	Unacceptable Performance Thresholds	Control algorithm recovery, performance mitigation measures	Control Algorithms	Protocols, data format	Performance
Interface Verification	Services required across interface, service agreements	Violated Interface preconditions / post-conditions, invariants; service interruption	Guards, inhibits	Preconditions, Post-Conditions, Mission Phase Interlocks, Event Control	Design coupling, robustness	Invariants, service interruption recovery, quality of service
End-to-End Performance Analysis	Service delivery	Consequences of service failure, unexpected emergent behavior	Distributed redundancy and fail-over	Crew, Mission	System of Systems	System of Systems
Timing Requirements	Nominal mission phases and segments	Blown performance margins, missing timing requirements	Safety and Recoverability, timing margins	Timeline management, phase transition interlocks	Constellation	Orion
Security Requirements	Secure communication	Compromised communication	Encryption methods, redundant communication channels	Flight termination system	Communication encryption standards	Redundancy

Legend: ● = Event integration analysis

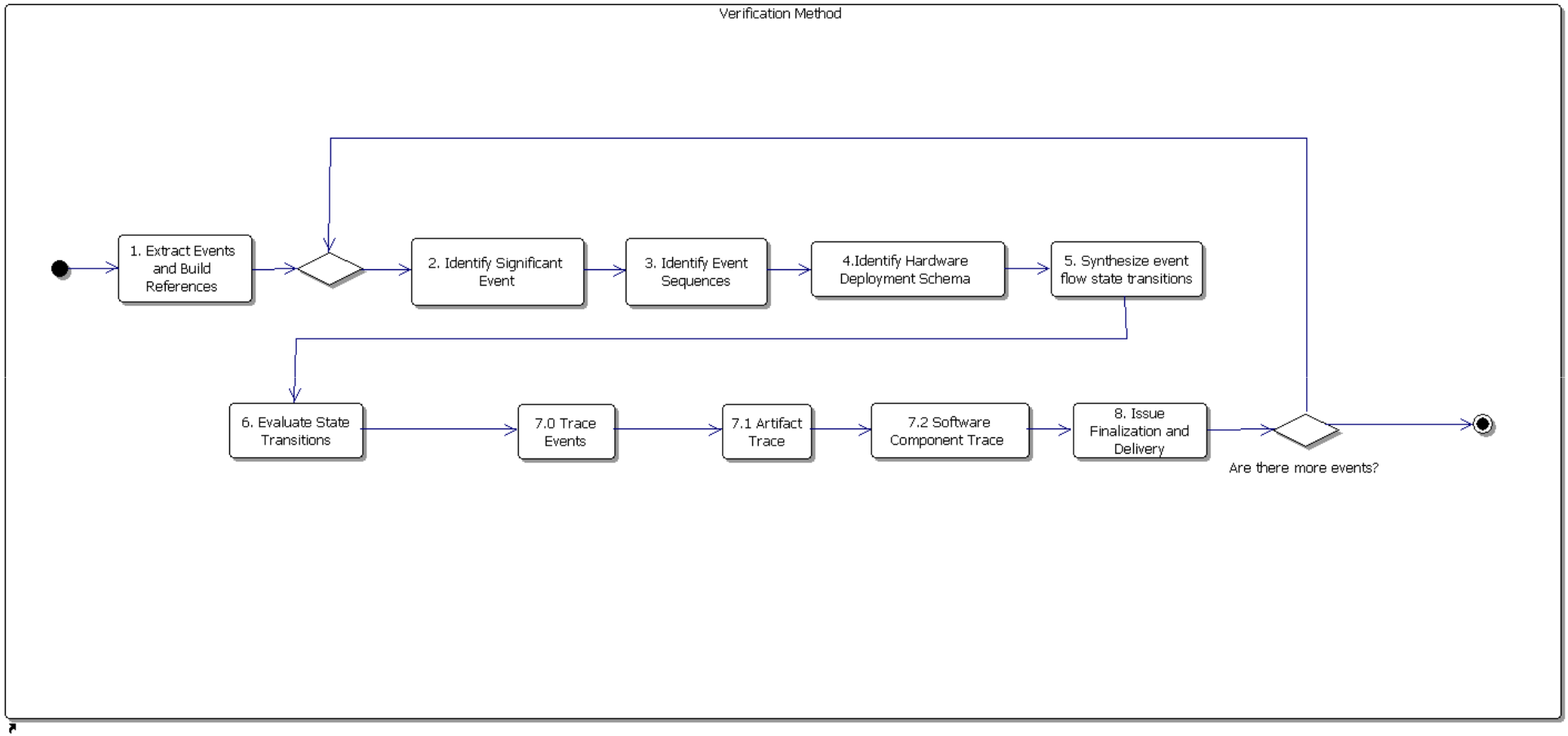
Legend: ○ = Future Targets

SW Architecture Verification – Examples 2

Analysis Dimensions	Intended behavior	Unacceptable Behavior	Adverse Conditions	Safety	Integration	Dependability
Suitability for Mission	Crew Safety, Mission Success ●	Loss of Crew, Loss of Mission ●	Degraded modes ●	Launch Abort System	Crew Office, MS, GS	Stability, Maturity, Control
Testing and Verification Requirements	Nominal modes	Fault Trees	Off-Nominal Modes	LOC/LOM prevention scenarios	Kedalion lab	Consistent, Complete, Unambiguous

Legend: ● = Event integration analysis

Event Integration Analysis



Event Integration Analysis Steps

1. **Extract Event References** -- Extract relevant event references from artifacts. Example event - CM/SM separation Use Orion CONOP, CM Spec, SM Spec, Little ADD V7, TMG SRS, TMG SDD, VMG SRS, VMG SDD, Pyro SRS, Pyro SDD, other documents as required or are relevant.
2. **Identify Significant Event** -- Elucidate necessary event operations.
3. **Identify Event Sequences** -- Show event leading up to event being analyzed, and the Domains that are involved.
4. **Identify Hardware Deployment Schema** -- Develop hardware diagram that depicts all relevant computing hardware involved in the subject event. (Example for CM/SM pyrotechnic event includes VCM, FCM, DCM, PDUs, etc.)
5. **Synthesize event flow state transitions** -- Show and describe event transition flows.
6. **Evaluate State Transitions** -- Evaluate event flow and state transitions for completeness and consistency using UML diagrams and artifact inputs. Use model animation to find weaknesses in state transition diagrams.
7. **Trace Events** -- Trace event threads from top down and from bottom up to look for gaps in requirements, design, and relevant documentation.
   1. **Artifact Trace** – Trace through artifacts looking for widows, orphans, etc. via CONOP, ADD V7, Subsystem Spec, SRS, SDD.
   2. **Software Component Trace** -- Trace through software components: domains, partitions, processes, displays, and others as needed.
8. **Issue Finalization and Summary** -- Wrap up issues in PITS and move all issue to ready for review.

Preliminary Event Integration Analysis Results

- We are about ½ way through our trial period with the new method for the Orion CM/SM Separation Event
- Modeling Orion domains and assigning critical event operations to those domains has improved our understanding of Orion.
- Examining pyrotechnic schematic diagrams has helped us better understand the emerging software architecture.
- Findings are emerging
  - Inconsistencies between the Timeline Manager, System Manager, and Vehicle Manager – division of responsibilities in requirements and design
  - The impact of the chain of latencies built into the pyrotechnic sequencing is unclear.
  - Vehicle proxy management may not be adequately specified
  - Pyrotechnic timing sequences appear to be incomplete across Orion CSCIs
  - FDIR for pyrotechnic event failure appears to be incomplete
- Areas where we need better understanding
  - The full set of safe transitions for CM/SM Separation
  - How the vehicle configuration is effected by CM/SM Separation
  - How Orion Configuration Data Sets are associated with critical events
  - Commanding redundancy and it's role in unrecoverable Orion events
  - Manual vs. Automatic event initiation

Conclusions

- Using an architecture framework such as DoDAF 2.0 works well for finding deficiencies in emerging software architectures. Orion Risk #34 is achieving good results.
- In developing a new method, it's important to work to NASA and industry standards. Don't reinvent the wheel. Use the knowledge base that already exists.
- Preliminary results from Event Integration Analysis appear promising – examining integration threads that lead to a non-recoverable event such as CM/SM separation provides focus for evaluating many aspects of the emerging Orion software architecture such as:
  - Software component synchronization
  - System latency analysis
  - Command redundancy
  - Fault Detection, Isolation, and Recovery (FDIR)
  - Event planning
  - Safety precautions: Inhibit, Enable
  - Data architecture and data-driven vehicle reconfiguration
  - Vehicle ordnance control
  - Manual vs. Automatic commanding
  - Hardware control via software
  - Cross-CSCI threads of control

Thank-you!