Software Assurance in a System of Systems World: Interoperability Challenges – Reports from the Field

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Carol A. Sledge, Ph.D.
October 2009

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

Essential Characteristics of Systems of Systems

Maier's Characterization of Systems of Systems

Autonomous constituents with independent operations and management

- Includes people, organizations, software agents, etc.
- Source of independent actions and decisions

Evolution…

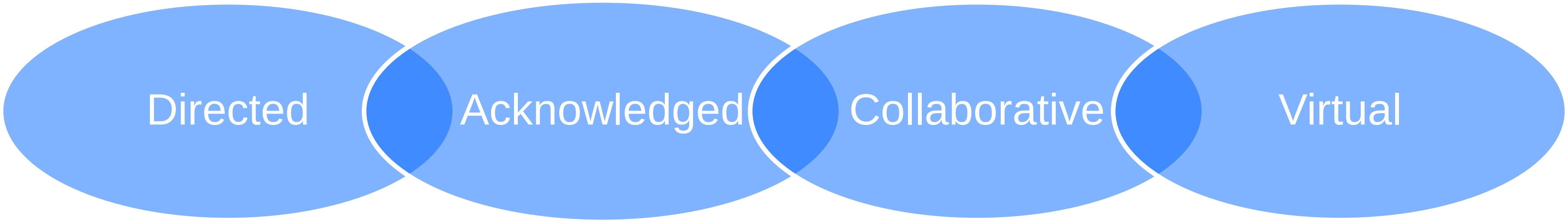
- Independent evolution of each constituent to respond to new technology and mission needs at its own pace and direction
- Evolution of the whole in response to changing demand

Emergent behavior

- "Whole is different than the sum of the parts"
- Indirect and cumulative effects of influences, actions, interactions

Maier, Mark W. "Architecting Principles for Systems of Systems," *Systems Engineering 1*, 4 (1998): 267–284.

Types of SoS*



Directed

- Integrated SoS, built and managed to fulfill specific purposes
- Centrally managed to maintain and evolve
- Constituents independent but subordinated to centrally managed purpose

Acknowledged

- Recognized objectives, designated manager and resources
- Constituents maintain independent ownership, objectives, funding, etc
- Changes based on collaboration between the SoS and the constituent

Collaborative

- Constituents interact more or less voluntarily to fulfill agreed central purposes

Virtual

- Lack central management authority and centrally agreed purpose
- Rely on relatively invisible mechanisms to maintain it

* DoD System Engineering Guide for System of Systems Engineering (Version 1.0, August 2008) & Maier

System of Systems Software Assurance Initiative

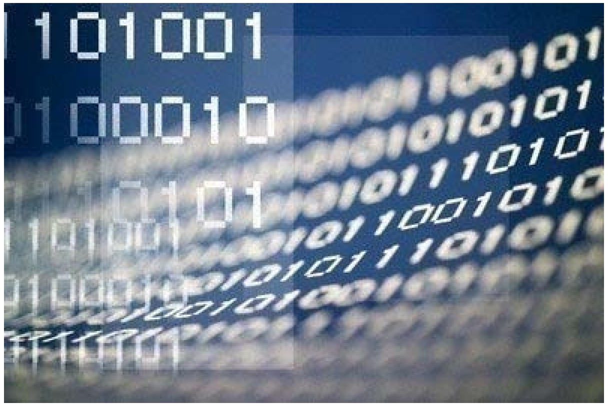
SoSSA Assurance Focus

System Assurance

- The justified confidence that a system functions as intended and is free of exploitable vulnerabilities, either intentionally or unintentionally designed or inserted as part of the system at any time during the life cycle*

Software Assurance

- Software's contribution to system and SoS assurance
  - Software assurance in the context of a system's mission and use



*** Engineering for System Assurance, NDIA System Assurance Committee, 2008, www.acq.osd.mil/sse/pg/guidance.html**

Initiative Scope and Goal

Scope

- Large-scale multi-user adaptive information management and C2 systems of systems (SoSs)

Goal: Methods and practices to provide

- Justified confidence that systems of systems will function as intended in their actual environment of use despite
  - The inevitable presence of various undiscovered defects and vulnerabilities
  - Unanticipated usage, environmental conditions, reconfiguration, or evolution
- Speedier delivery of fielded SoS capability

Integration & Interoperability

One Aspect: Integration and Interoperability

Currently, primarily interoperability issues surfaced at integration of the SoS for test and evaluation prior to fielding

- far too late in the systems engineering lifecycle to effectively and efficiently deal with the issues

Additional challenges with SoS

- underlying constituent systems in an SoS are constantly and independently evolving
  - producing a constant state of evolutionary and continual deployment

Need to surface (and mitigate) interoperability and integration issues earlier in the SoS lifecycle

Premise

Leverage insight from prior and existing DoD SoS

- DoD and industry sources

Re

- interoperability “failures” (and how to surface interoperability and integration issues earlier in the SoS lifecycle)
- what practices have facilitated better and quicker integration
- were there software approaches that could have helped mitigate the issues
- were there associated DoD policy, acquisition, and procedure challenges/barriers/incentives

Assumed anonymity/“genericized” unless explicit permission given

Overall Findings

Reluctance to discuss SoS interoperability “failures”/challenges, even with anonymity

Lack of “higher level” sharing of knowledge

- Software engineering issues, risks and lessons learned
- Organizational, management and governance
- Analysis, capture and dissemination
  - Experience (over years)
  - What has worked and what has not (post mortem)
- Time, cost and “not in the mainstream”

Magnification by SoS of existing, known software system problems plus new and emergent problems

Some Specific Comments from Interviews

Interoperability claimed but …

Find problems, do workarounds but then forget about problems – to be discovered again

No good processes that look at interoperability issues (id, avoid or mitigate them, disseminate solution (collection agency or repository))

Interoperability “personality” driven

- Individual takes it on to identify, document and work with programs to get it resolved

Different standards, interfaces, etc.

- Surface interoperability issues much earlier and develop mitigations or solutions (especially cross service)
  - Find the right people, at the right time, at the right level
- Even within service, may have different types of equipment that can’t talk to one another
  - Trying to avoid dependence on one company (fair share)

Specific Comments: Leveraging the Learning Curve

Positive experience – in sustainment, doing things early, being proactive

After action reports, other lessons learned, “knowledge base”

- Sometimes the knowledge base is a person (personality and social networks)
  - “Human interoperability”
- Attempting to institutionalize it

Earlier in the life cycle – going against grain

- Still dealing with hardware, beginnings of software engineering, do some preliminary software interoperability
  - Not in contract, far down in WBS
- Knowledgeable people “on board” earlier – avoid mistakes or consider what has happened in similar situations

Artifacts

Currency, existence, completeness, and accessibility

- Architecture
- Design
- Rationale
- Assumptions
  - Implicit assumptions
  - Not machine-checkable
- Data and information
  - Semantic/lexicons
- Access to and incompatibility of information
  - Different tools
- Level of detail
  - Critical information not captured in artifacts
  - What is critical, what becomes critical (based on changes)

Identified Issues for Architecture/Architects

Do not have adequate software architecture documentation in place

- Modification to what the system is interfacing to
  - Time and money to bring “as is” architecture documentation up to date and still do the “to be” architecture documentation

Architect needs to talk directly with customer(s) to understand expected use

- Uncover interoperability issues

Similarly architect requires timely access to internal corporate subject matter experts

- Share expertise

Identified Testing Issues

Mission threads do not reflect current operational environment reality

Poor systems level testing done

Changes to various systems

- How do those changes affect the threads and tests

Core systems - one simple change of interface standard by a core system, caused many problems in other systems

Challenge: processes, artifacts, and collaborations in systems of systems are dynamic and ongoing, not static.

- Implies continual integration and test are necessary
  - Interim and incremental demonstration of interoperability, SoS functionality, and SoS capability

Evaluation and leveraging of evidence become increasing important

Identified Practice Issues

Integration, interoperability – mostly considered late in life cycle

- Earlier integration
  - Allow systems to come to test floor/op. environment prior to formal integration
  - Interoperability risk reduction exercises
    - C4ISR On-The-Move (integrated technology demonstration)
    - Tactical Network Topology (field experiment exercise environment)

Specific guidance (usually lower level)

- Net-Centric Enterprise Solutions for Interoperability [NESI]
  - Cross service effort (Army, Navy, DISA); http://nesipublic.spawar.navy.mil
  - “Body of architectural and engineering knowledge that guides
    - Design, implementation, maintenance evolution and use of IT portion of net-centric solutions for defense applications”
  - E.g. information interoperability: “To be able to share information, applications must be able to share data and to agree on its meaning” (access to data, semantic match)

DoD Policy, Acquisition, and Procedure Challenges/Barriers/Incentives

Most SoS are not Programs of Record

- Usually no specific SoS funding, authority, management or engineering
- At best, influence the new, or changes, upgrades

Individual systems do not consider larger context (interfaces, interdependencies, etc.)

Constant SoS evolution, continual deployment

- Coordination, collaboration amid change and turnover
- (Re)certification

Incentives and rewards focus on system, not SoS

- What is best or better for SoS, may not be optimal or desired for an individual system
- Challenges to meet system milestones/deliverables
- (Early)Dissemination of (potential) changes/problems to others detrimental to program/contractor

Request to Audience (from a SoS Point of View)

Pointers to and access to DoD and industry sources to leverage insight re

- Interoperability “failures” (and how to surface interoperability and integration issues earlier in the SoS lifecycle)
- What practices have facilitated better and quicker integration
- Were there software approaches that could have helped mitigate the issues
- Were there associated DoD policy, acquisition, and procedure challenges/barriers/incentives

Additionally seeking insight and information

- How conclusions about (software) system interoperability could be developed faster & more accurately by taking advantage of evidence gathered throughout the lifecycle
- Determine what evidence could be provided at different stages and how it could be used to develop justified predictions that a fielded system will not experience certain types of interoperability problems

Contact Information

Presenter:

Carol A. Sledge, Ph.D.

Research, Technology, and System Solutions

Telephone: +1 412-268-7708

Email: cas@sei.cmu.edu

World Wide Web:

www.sei.cmu.edu

U.S. mail:

Software Engineering Institute

Carnegie Mellon University

4500 Fifth Avenue

Pittsburgh, PA 15213-2612

USA

Visit the SEI display at the Regatta Pavilion

Software Engineering Institute | Carnegie Mellon