Common Software Platforms in System-of-Systems Architectures: The State of the Practice

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

John Klein
Sholom Cohen
Rick Kazman

31 Oct 2013

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM-0000683

Outline

Context
- Many meanings of “platform”
- Study Goals

Research Method

Interview Questions

Results and Discussion
- Response Groupings
- Architecture Framing and Processes
- Challenges and Patterns of Success
- Constraints on a Solution

Conclusions

Context

System of systems distinguished by

- Operational independence of constituent systems
- Managerial independence of constituent systems

Interoperability is a primary architecture concern

“SoS Platform”

- Provides services and functions to all constituent systems
- One strategy to promote interoperability

“Platform”

Several definitions:

- Vehicle that transports systems and provides services such as power and cooling [Greenert 2012]
- Common elements reused across a product line or product family
  - “Product Platform” [Cusumano 2010]
  - “Platform-based Engineering” [Madni 2012]
- “Industry Platform” [Cusumano 2010] or “SoS Platform” [Klein 2012]
  - Provides services to an open set of systems that interact to form a SoS
  - General-purpose services, e.g., directory and authentication
  - Domain-specific services, e.g., geospatial information processing for a command and control SoS

SoS Platforms

SoS Platforms address architecture concerns

- Interoperation
  - Common communication mechanisms
  - Common information models (semantics)
  - Patterns or sequences of interaction
- Reduce time and effort to develop or modify systems for use in the SoS
  - Reference or concrete implementations of services
  - Replace point-to-point integration with system-to-platform integration
  - Reduced barrier to entry for new systems to join SoS
- These enable modular substitution of constituent systems in the SoS
  - An SoS platform supports an “ecosystem”

SoS Platform Examples

Commercial ecosystem-enabling platforms include:

- Facebook
- Apple (iOS, OS X, iCloud, App Store, etc.)
- Salesforce.com

Defense system domain examples include:

- Future Avionics Capability Environment (FACE)
- US Army Common Operating Environment (COE)

Goals of this Study

Many examples of successful commercial SoS platforms, but success in defense and other domains has been elusive.

We wanted to answer the following questions:

- What processes are used to develop SoS architectures, and how are software elements of the architecture treated in the processes used?
- What challenges do SoS programs face in developing architectures; performing test, integration, and assurance; managing runtime configuration and operation; and evolving the SoS? What approaches have been used in successful programs to overcome these challenges?
- What are the constraints on new approaches to developing, using, and evolving these SoS architectures?
- What are the important differences between practices used to create commercial SoS architectures and military SoS architectures?

Research Method

First Attempt – Convene a Workshop

- Recruited participants from the professional network of research team members
- Inclusion criterion – direct experience as an architect or systems engineering lead for at least one SoS
- Only one invitee agreed to participate – some responses implied a reluctance to publicly share experience

Led us to develop an interview protocol that reported responses anonymously

Interviewee Demographics:

- 14 initial participants, 2 withdrew → 12 interviewees
- 10-25 years of professional experience
- 8 participants worked on 4 or more SoS projects, 2 worked on 1 SoS project
- 4 participants from commercial organizations, 5 from military system development contractors, 3 from government side of military system development

Interview Questions

1. SoS architecture development processes, including
   - How did interviewee define SoS?
   - Conflicts between classical systems engineering perspective (“is part of” decomposition) and software architecture perspective (layers, “is used by”)
   - System vs. SoS concerns and tradeoffs
2. Challenges in various SDLC phases
   - Development; Test, Integration, and Assurance; Runtime Configuration and Management; and Sustainment and Evolution
   - Examples of successes and gaps in current practice
3. Constraints on new design and analysis methods

Complete question set available at http://www.andrew.cmu.edu/user/jklein2/SoS-Study-Interview-Questions.pdf

Results – Response Groupings

Responses separated into three broad groups, aligned with participants' experience:

- Commercial SoS Platforms
- Command and Control SoS
- Military SoS Platforms

For example, in defining and framing the SoS platform:

- Command and Control SoS group and Military SoS Platform group defined the platform in terms of "what it is":
  - technology characteristics such as APIs and programming language bindings
  - Services provided
- Commercial SoS Platform group defined the platform in terms of "what it does" – creates network effects that support an ecosystem

Results – Architecture Framing

Two scenarios identified:

- New SoS comprised of new systems (“directed SoS”)
  - Success with both top-down and bottom-up approaches to defining the SoS Platform
    - Top-down - platform emerges as commonalities are discovered as systems are designed
    - Bottom-up – platform defined first, then systems are designed
- Existing systems integrated to create SoS (“acknowledged SoS”)
  - Architecture is much more constrained
  - Less conceptual integrity across the constituent systems

Functional vs. Quality Attribute Requirements

- Command and Control SoS group reported recent projects explicitly frame decisions in terms of both functional and quality attribute requirements
- Commercial SoS group considered both types of requirement in making decisions, and did not explicitly distinguish between them

Results – Architecture Processes

No reports of particular methods or approaches for SoS or SoS Platform architecture development or analysis

No use of economic modeling to analyze design alternatives

- Commercial SoS group driven by time-to-market
- Military SoS group performed trade studies, but often "backed into" architecture decisions based on development constraints

All reported that deep domain knowledge is critical to making timely decisions

- Identify the most important tradeoffs
- Eliminate ineffective parts of the solution space

Tradeoffs between short-term and long-term concerns

- Command and Control SoS group prioritized operational performance over downstream lifecycle costs and sustainment costs
- Commercial and Military SoS Platform groups reported a need to consider future needs
  - Commercial organizations identified future needs through market analysis
  - Military platforms informed future needs using S&T investment roadmaps

Results – Primary Challenges

Primary challenges in developing and evolving SoS architectures are due to non-technical factors, including:

- Misalignment of development organization and lines of authority with the architecture
- Misalignment of system and SoS goals
- Reluctance to introduce dependence on a SoS platform into the constituent system architectures
- Regulations and Policy (for systems acquired by the US government) that diminish the value of a SoS platform

Difficult to justify migrating existing systems to replace organic feature with SoS platform services – this may not produce new capabilities

- Incurs short-term cost, but produces long-term value due to reduced integration and sustainment costs
- Commercial platforms can introduce incentives, military platforms rely on top-down mandates

Many similarities to adopting, developing, and sustaining Software Product Lines and Platform-based Engineering approaches

Results – Other Challenges

Many participants reported issues related to insufficient documentation of constituent systems

- Extensive documentation available, but focused on independent system operation
- Need to address interoperation concerns such as resource scheduling, resource sharing, and interface error/exception handling

SoS scale and complexity challenge the initial instantiation of the SoS platform

- V-model leads to long delay from architecture definition to system integration
- Some DoD projects using iterative/agile development approaches, but there were questions about when to start iterations and how to plan iterations
- Two proto-patterns identified in commercial organizations to bootstrap the initial SoS platform instantiation

Results – Proto-patterns for new platforms

Evolving the architecture of a new platform – allows the platform to be created quickly, and workflows to emerge based on actual platform use:

- Define individual platform message types and schemas, with no attention to workflows (sequences of messages to perform a particular business process)
- Initially, all workflow is organically built into the systems that use the platform
- Later, add workflow orchestration to the platform, with workflow versioning that includes endpoint roles, sequences, and transaction support.
- Migrate systems to use platform-provided workflow

Scaling up performance and availability:

- Platform provides features to maintain workflow state only in the participating endpoints, not in the platform infrastructure
- Refinement of SOA stateless services approach

Results – Challenge of Backward Compatibility

Commercial platforms use extensive test automation

- "Nearly all" tests are automated
- "Tens of thousands" of tests allow maintaining full backward compatibility through 20+ versions of the platform

Proto-pattern – 3-step process for introducing new features

- Pilot with limited set of selected users, with special IT operations processes in place to monitor usage and quality attributes such as performance
- Stabilize the features with the limited set of selected users, and transition to standard IT operations processes
- Make feature generally available in production environment

Solution Constraints

Integration with existing tools

- Architecture modeling, analysis, and documentation tooling
- Project management tooling – efficiently translate architecture decisions into cost, schedule, and risk metrics

Alignment with assurance and certification processes

Resolve conflict with US government acquisition policies and regulations

- There are activities addressing this, e.g., US Navy Open Architecture Business Innovation Initiative

Conclusions – Areas for Additional Research

Selection of features for SoS Platform

- Critical stakeholder concerns include time-to-market, ease of adoption, support for future needs, alignment with assurance/certification processes
- Problem space perspective – identify candidate features and assess value using mission thread analysis and domain analysis
- Solution space perspective – cost to implement and maintain features using economic modeling supported by cataloged architecture knowledge

Application of agile methods for developing platform-based SoS

- Modeling complicated dependencies in SoS architectures
- Accommodate managerial independence of constituent systems

Documentation of constituent systems to address SoS concerns

- Identify relevant concerns and appropriate modeling approaches
- Formalize in 42010-style architecture viewpoint, map to DoDAF

For more information

Contact:

John Klein, Senior Member of the Technical Staff

jklein@sei.cmu.edu

See our full paper about this study at

- http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=68315

Patterns for SoS software architectures

- http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6575257
- Upcoming SEI Technical Report at http://resources.sei.cmu.edu/library/

Adoption of Agile Methods in DoD

- http://www.sei.cmu.edu/acquisition/research/
- Contact Mary Ann Lapham (mlapham@sei.cmu.edu)