A Uniform Approach for System of Systems Architecture Evaluation

Michael Gagliardi, William G. Wood, John Klein, and John Morley
Software Engineering Institute

For a large-scale system of systems (SoS), severe integration and run-time problems can arise due to inconsistencies, ambiguities, and gaps in how quality attributes (such as reliability) are addressed in the underlying systems. This is exacerbated in contexts where major system and software elements of the SoS are developed concurrently and oftentimes independently. Using a defense system scenario, this article outlines a uniform approach for capturing quality attribute requirements as augmentations to mission threads early in the development process and for analyzing SoS, system, and software architectures against these mission thread augmentations.

An SoS program, in government or industry, can suffer severe integration and run-time problems that can result in costly rework, schedule overruns, and the failure to achieve performance goals [1]. For example:

- In 2005, NASA's Demonstration of Autonomous Rendezvous Technology (DART) spacecraft collided with its target rather than achieving orbit, scuttling a $110-million mission. NASA's official investigation cited a number of contributing factors, including the failure to uncover a number of design issues prior to SoS integration [2].
- A year before the DART debacle, the Ford Motor Company killed a new supply chain system—after spending four years and about $400 million on its development—and returned to a set of "custom-written mainframe applications" for purchasing. "Poor performance" was cited as the culprit [3].
- A few years before Ford's loss, the self-developed billing and claims-processing system at Oxford Health Plans failed miserably after deployment, triggering a drop of more than $3 billion in the corporation's value. The system couldn't keep pace with the organization's needs [4].

One significant underlying cause for problems like these is a lack of attention to quality attributes (such as interoperability, sustainability, performance, and reliability[1]) early in the development life cycle, when their implications on the SoS architecture can be dealt with more easily. A recent study by the National Defense Industrial Association found that one of the top issues hindering the acquisition and successful deployment of SoS is that "insufficient systems engineering is applied early in the program life cycle, compromising the foundation for initial requirements and architecture development" [5].

® The Architecture Tradeoff Analysis Method and ATAM are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

The typical SoS context—where major system and software elements have their own architecture documentation created by different contractors using diverse tools and notations—makes it more difficult to focus on quality attributes. In these contexts, program managers and SoS architects need a way to promote consistency, clarity, and completeness of quality attribute requirements throughout the development of the SoS.

This article describes a two-pronged, uniform approach for the early identification of quality attribute inconsistencies, ambiguities, and gaps within SoS and system architectures. Using an approach that focuses on SoS quality attribute considerations can produce benefits such as:

- Improved SoS architecture.
- Early identification of significant architectural challenges and risks.
- Better communication between the SoS, system, and software stakeholders.
- More predictable integration of component systems.
- More effective root cause analysis of problem areas.

After defining the uniform approach, we use a defense system illustration to show how this uniform approach captures considerations about reliability early in the life cycle and influences risk management throughout the SoS, system, and software development.

Uniform Approach Described

The two-pronged, uniform approach includes:

- A methodology to perform a *first pass* identification of architectural risks[2] at the SoS level, using existing mission threads that are augmented with quality attribute concerns. This is provided through a series of mission thread workshops (MTWs) and SoS architecture evaluations. The results are organized into a number of risk themes, and then individual systems are associated with these risk themes.
- Further evaluation of the problematic constituent systems can be performed using the augmented mission threads from the SoS architecture evaluations and employing an extension of the Carnegie Mellon SEI Architecture Tradeoff Analysis Method® (ATAM®) for system and software architecture evaluation (System and Software ATAM)[3].

This approach is based on successful SEI methods and techniques for addressing key quality attributes, their relationships, and trade-offs at the software architecture level. Two well-established and widely used methods are the SEI Quality Attribute Workshop (QAW) and the ATAM. The QAW helps acquirers and developers identify and characterize the key quality attributes for a system. The ATAM enables software developers and acquirers to evaluate software architecture against required quality attributes and business/mission goals before the system is actually developed. Over the past decade, many DoD programs have used the ATAM to evaluate their mission-critical software architectures.

Through this approach, architectural risks are identified early in the life cycle, promoting more efficient and effective risk management. As shown in Figure 1, the MTW takes warfare vignettes, business/mission drivers, mission threads, and SoS architecture plans as input and produces mission threads augmented with quality attribute requirements and a set of SoS architectural challenges. These augmented mission threads and architectural challenges should then be used in the development of an SoS architecture. When the SoS architecture is somewhat mature, the augmented mission threads then serve as the basis for an SoS architecture evaluation, which uncovers SoS architecture risks and points to individual systems that may pose difficulties in meeting the quality attribute concerns

reflected in the mission threads. Although the emphasis in these activities is firmly on quality attributes, the evaluation often exposes functional gaps, inconsistent Concept of Operations, and process ambiguities. Then, through a System and Software ATAM, architectural risks at the system and software levels are identified.

Mission Thread Workshop

Mission threads are associated with one or more warfare vignettes. A vignette describes the overall environment, including the geography; its own force structure and mission, strategies, and tactics; and the enemy's forces, attack strategies, and tactics, including timing.

A mission thread has been defined as "sequence of activities and events beginning with an opportunity to detect a threat or element that ought to be attacked and ending with a commander's assessment of damage after an attack" [7]. Our mission thread-based example vignette is as follows:

> An enemy tank platoon is threatening a lightly protected company and comes into the field of view of an unattended ground sensor (UGS), which connects to and informs a manned ground vehicle for command and control (MGVC2). An MGVC2 identifies the enemy tanks. The MGVC2 assigns an unmanned missile launcher (UML) to engage the tank platoon. The UML engages and destroys the enemy. The MGVC2 determines that the threat has been eliminated, based on subsequent UGS signals.

The MTW is a facilitated process that brings together SoS stakeholders to both augment existing mission threads with quality attribute considerations that will shape the SoS architecture and identify SoS architectural challenges. These stakeholders include, but are not limited to, the following:

- Lead SoS architects (for the contractor and program management office [PMO]).
- Lead system and software architects (for the contractors and PMO).
- Program managers (for the contractors and PMO).
- Key representatives from integration and test, requirements, users, maintenance, installation, independent verification and validation, modeling and simulation, and other areas.

There are three main stages to the MTW: 1) preparation, 2) execution, and 3) roll-up and follow-up. In the preparation stage, the SoS program manager develops an overview presentation on the SoS mission/business drivers, and the SoS architect develops an overview presentation on the SoS architecture plans. The facilitation team meets with the SoS program manager and architect to plan the MTW, provide feedback on the two presentations, reach agreement on mission threads and types, and identify stakeholders. In this first stage, the facilitation team may need to decompose warfare vignettes into activity-oriented mission thread steps by considering:

- External actors, who may not be explicit in the vignette.
- Functionality and capability and their distribution.
- Command structure.
- Manual versus automated activation.

During the execution stage of the MTW, the SoS program manager delivers the presentation on the SoS business/mission drivers, including the business/programmatic context, the plan for development, as well as high-level functional requirements, constraints, and quality attribute requirements. The SoS architect then presents the SoS architecture plans, including: key business/programmatic requirements, key technical requirements and constraints that will drive architectural decisions, existing context diagrams, high-level SoS diagrams and descriptions, development spirals, and an integration schedule.

The bulk of the MTW execution stage is spent augmenting the mission threads with quality attribute considerations and identifying SoS architectural challenges based on stakeholder inputs and the dialogue between the stakeholders and the architects. During the augmentation, overarching quality attribute considerations—as well as step-specific quality attribute considerations for each quality attribute of the SoS—are elicited and documented for each mission thread. The architects can also describe how the planned architecture satisfies the quality attribute considerations in each step for each mission thread

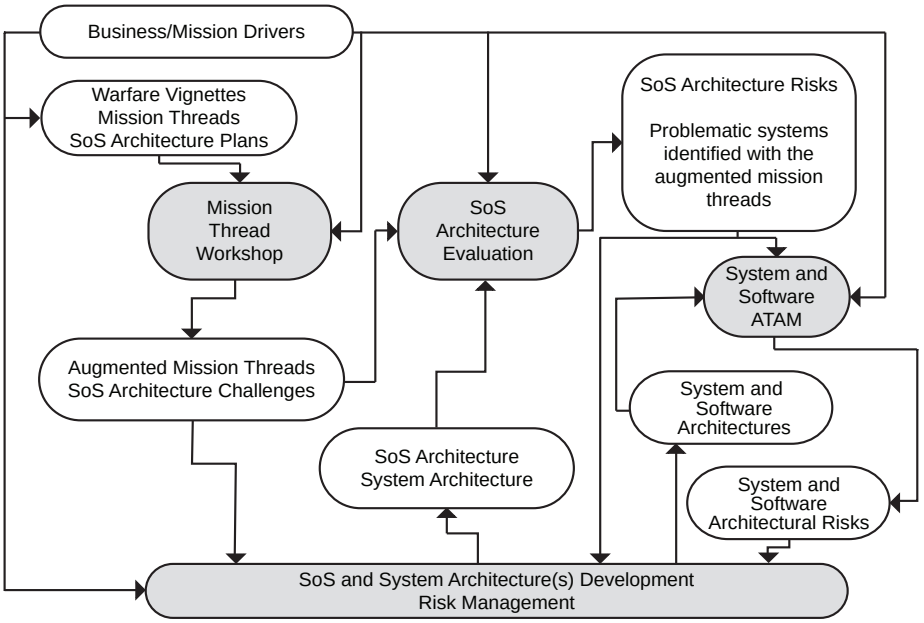

Figure 1: *A Uniform Approach for Identifying SoS Integration Problems Early*

Table 1: *Mission Thread Elements Augmented by Reliability Requirements*

Mission Thread Element	Augmentation for Reliability
An enemy tank platoon is threatening a lightly protected company and comes into the field of view of a UGS, which connects to and informs an MGVC2.	UGS-MGVC2 connection fails one second after connection; reconnects after two minutes.
An MGVC2 identifies the enemy tanks.	Takes too long (over five minutes) to deconflict and identify the enemy tanks.
The MGVC2 assigns a UML to engage the tank platoon.	Communication transmissions to UML are missing their deadlines.
The UML engages and destroys the enemy.	Assigned UML fails to launch.
The MGVC2 determines that the threat has been eliminated from subsequent UGS signals.	Signals from UGS have ceased. Communication is lost between the MGVC2 and UGS.

selected for the workshop. Table 1 (on the previous page) shows some possible requirements for reliability expressed as augmentations on the steps in the example mission thread presented earlier.

The process used in the execution stage fosters a dialogue between architects and stakeholders regarding the issues and challenges that are captured during the workshop. In the context of our example mission thread, there could be issues such as:

- The connection with the UGS and the MGVC2 is dynamic—that is, the sensor must announce the presence of an object of interest when it senses one. More than one MGVC2 platform may be notified by the UGS, and a mechanism is needed to choose which platform will act on the data.
- The connection between the UML and the MGVC2 is also somewhat dynamic, but the MGVC2 may have an indirect connection to the UML.
- In some cases, the MGVC2 may not have the authority to engage a UML, but will have to report to a higher level command post.
- The missiles are guided in flight by the MGVC2, which may not have a direct connection.
- The tracks are created for the regional fusion engine, which may not be the MGVC2 and is not directly represented in the mission thread.

In the third and final stage, roll-up and follow-up, two reports are produced from the activity in the execution stage and answers to action items are assigned in that stage. One report includes the quality attribute considerations for each step of each mission thread selected for the MTW and overarching quality attribute considerations for each mission thread. The other report describes the SoS architectural challenges.

Experience executing QAWs (the basis of MTWs) shows that the most effective sessions bring together no more than 20 stakeholders for one to two days. This allows each MTW to tackle the augmentation of just one or two mission thread types; through a series of MTWs, the variety of mission thread types involved in an SoS are examined. At the end of the series of MTWs, an annotated summary briefing rolls up SoS architectural challenges and strengths, non-architectural issues (if any are uncovered), and recommendations.

MTWs have been used in conjunction with two large, complex DoD SoS programs. These MTWs assist architects in identifying SoS architecture challenge areas and the areas to focus their prototyping and proof-of-concept activities.

SoS Architecture Evaluation

In conjunction with the MTW, the SoS architecture evaluation provides a *first pass identification* of SoS architectural risks and quality attribute inconsistencies across the constituent systems. An SoS architecture evaluation:

- Uses outputs of the MTWs, including augmented mission threads and SoS architecture challenges.
- Incorporates the expertise of a trained evaluation team and SoS stakeholders, including the SoS and system architects.
- Probes architecture at the areas where the systems interact to identify risks.
- Organizes the individual risks into risk themes that can be comprehended (and mitigated later) by program management.
- Assesses the sufficiency of architecture documentation.
- Identifies potentially problematic systems for focused follow-on evaluations using the specific augmented mission threads.

In the SoS architecture evaluation, the SoS architect walks each augmented mission thread through the SoS architecture, describing how the SoS architecture and constituent system architectures satisfy the thread's functional and quality attribute considerations. For each step in the mission thread, the evaluation team and stakeholders probe the SoS architecture (and system architecture, if necessary) with a focus on the quality attribute augmentations and SoS challenges; risks, issues, and strengths are also identified.

At the end of each evaluation, the evaluation team delivers an outbrief that includes SoS architectural risk themes and strengths, non-architectural issues discovered, an identification of potentially problematic systems, and recommended next steps.

System and Software ATAM

As a follow-on to the SoS architecture evaluation in this approach, the System and Software ATAM keys in on the problematic systems identified. This evaluation uses the augmented mission threads to examine the system and software architecture and produces a set of software and system architectural risks that trace back to the quality attributes identified in the augmented mission threads. The System and Software ATAM is built on the format and approach of the ATAM.

Summary

Problems that do not surface until integration or deployment can have ruinous effects on the cost, schedule, and performance of SoS programs. Through the uniform approach outlined in this article, the SoS architects can use the augmented mission threads as one input for SoS architecture development. The augmented mission threads can also be used to identify any SoS architectural risks related to the quality attributes needed to accomplish the mission/business objectives early in the life cycle, promoting more efficient and effective risk management.

Table 2: *The Uniform Approach*

	MTW	SoS Architecture Evaluation	System and Software ATAM
Input	• Selected existing mission threads • Warfare vignettes • SoS architecture plans overview • SoS business/mission drivers	• Augmented mission threads • SoS architecture challenges • SoS business/mission drivers • SoS architecture • System architecture	• SoS architecture risks • Problematic systems identified with the augmented mission threads • SoS and system business/mission drivers • System architectures • Software architectures
Output	• Augmented mission threads that reflect quality attribute considerations • Architectural challenges • Issues and questions concerning the mission threads	• SoS architecture risks • Problematic systems identified with the augmented mission threads	• System architecture risks • Software architecture risks

This approach involves performing a series of MTWs and SoS architecture evaluations to identify inconsistencies, ambiguities, and gaps across the constituent systems and at the SoS level, using existing mission threads that are augmented with quality attribute concerns. It also includes further evaluation of problematic constituent systems using the augmented mission threads in a system and software version of the SEI ATAM. Table 2 summarizes the way the methods are integrated for analyzing SoS, system, and software architectures against augmented mission threads in order to expose technical risks at an early stage of development when mitigation can be done cost-effectively.

By using this uniform approach—focused on SoS quality attribute considerations early in the development life cycle—program managers and SoS architects can realize an improved SoS architecture, identification of significant architectural challenges and risks at a time when it is less costly to fix them, better communication between SoS, system, and software stakeholders, more predictable integration of component systems, and more effective root cause analysis of problem areas.◆

Notes

1. A suitable list of SoS quality attributes also includes backward compatibility, testability, and usability.
2. A risk is a potentially problematic architectural decision indicating that the system or SoS built from the architecture may not completely satisfy one or more business/mission goals.
3. The ATAM exposes architectural risks that potentially inhibit the achievement of an organization's business goals. The ATAM gets its name because it not only reveals how well an architecture satisfies particular quality goals, but it also provides insight into how those quality goals interact with each other—how they *trade off* against each other [6].

References

1. Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics. "Report of the Defense Science Board Task Force on Development Test and Evaluation." May 2008 <www.acq.osd.mil/dsb/reports/2008-05-DTE.pdf>.
2. Marshall Space Flight Center. "NASA Report: Overview of the DART Mishap Investigation Results – For Public Release." 15 May 2006 <www.spaceref.com/news/viewsr.html?pid=20605>.
3. Songini, Marc L. "Ford Abandons Oracle Procurement System: Switches Back to Mainframe Apps." Computerworld Software. 23 Aug. 2004 <www.computerworld.com/software-topics/erp/story/0,10801,95404,00. html>.
4. Hammonds, Keith H., and Susan Jackson. "Behind Oxford's Billing Nightmare: How a Misconceived System Cost the Health-Care Giant Millions." Business Week. 17 Nov. 1997 <www.businessweek.com/1997/46/b3553148.htm>.
5. National Defense Industrial Association Systems Engineering Division Task Group Report. "Top Five Systems Engineering Issues Within Department of Defense and Defense Industry." July 2006.
6. Clements, Paul, Rick Kazman, and Mark Klein. Evaluating Software Architectures: Methods and Case Studies. Boston: Addison-Wesley Professional, 2001.
7. Naval Studies Board. C4ISR for Future Naval Strike Groups. Washington, D.C.: The National Academies Press, 2006.

About the Authors

Michael Gagliardi has more than 25 years experience in real-time, mission-critical software architecture and engineering activities on a variety of DoD systems. He currently works in the SEI Research, Technology, and System Solutions Program on the Architecture-Centric Engineering initiative, and is involved in the development of architecture evaluation methods for SoS architectures and system architectures.

SEI
4500 Fifth AVE
Pittsburgh, PA 15213
Phone: (412) 268-7738
Fax: (412) 268-5758
E-mail: mjg@sei.cmu.edu

William G. Wood has been a member of technical staff at the SEI for more than 22 years. During this time, he has managed a technical program and technical projects, and has provided technical support to the program development organization. Wood is currently working in software architecture with a number of clients. He has a master's degree in electrical engineering from Carnegie Mellon University, and a bachelor's degree in physics from Glasgow University, Scotland.

Phone: (412) 268-7723
Fax: (412) 268-5758
E-mail: wgw@sei.cmu.edu

John Klein is a senior technical staff member in the SEI's Research, Technology, and System Solutions Program. He does research and consulting on architecture-centric methods and tools for developing, documenting, and evaluating SoS and enterprise architectures. Klein also assesses and improves the architecture competence of individuals, teams, and organizations. Klein has more than 25 years experience in systems development including sensors and weapons, as well as collaboration systems.

Phone: (412) 268-4553
E-mail: jklein@sei.cmu.edu

John Morley is a member of the operating staff at the SEI. He has reported on model-based engineering, architecture-centric engineering, and service-oriented architecture for SEI publications and has more than 20 years experience in writing and editing scientific and technical materials. Morley recently co-wrote "Building Secure Systems Using Model-Based Engineering and Architectural Models," which appeared in the September 2008 edition of CROSSTALK.

Phone: (412) 268-6599
Fax: (412) 268-5758
E-mail: jmorley@sei.cmu.edu