Assessing the Longevity of Simulation & Training Architectures

Wesley A. Milks, PhD
Applied Research Associates, Inc.
3452 Lake Lynda Drive, Suite 250
Orlando, FL 32817
wmilks@ara.com

Sandra Veautour
US Army, PEO STRI, Project Support Group
12350 Research Parkway
Orlando, FL 32826-3276
sandy.veautour@us.army.mil

ABSTRACT

Architectural longevity is a frequently desired and rarely achieved goal of many simulation and training systems. Requiring attention to detail with respect to openness, modularity, reliability, performance, scalability, interoperability, and maintainability, addressing architectural longevity from the onset of a project offers potential for significant tangible and intangible benefits. The inevitability of requirements changes, technology evolution, emerging standards, and funding changes further complicates architecture longevity. Investing a relatively small effort to establish an architectural assessment framework from project start can yield a significant savings while also providing an environment for informed decision-making by the collective development team.

The Software Engineering Institute offers several general techniques for assessing software and system architectures. Tailoring these techniques to the unique aspects of the military simulation and training community opens the potential for improving training, enhancing analysis, and reducing support costs for delivered systems. This paper offers a summary of methods for architectural assessment, lessons learned applying the methods, and a recommended best practice for a new standard methodology for integrating architectural assessment into the military simulation and training community.

ABOUT THE AUTHORS

Dr. Milks has over 18 years of experience performing systems engineering and architecture development for large, software-intensive military systems. His architecture experience includes chief architect for the Common Training Instrumentation Architecture (CTIA) program and the lead architect for the Warfighter Simulation (WARSIM) and Joint Simulation System – Land Component (JSIMS-Land). While working as an Army civilian, he was selected as the Standards Category Coordinator for System Design & Architecture. In this role, Dr. Milks coordinated and documented architectural standards for Army simulation systems. While leading the CTIA project, Dr. Milks was selected by Software Engineering Institute (SEI) as in invited speaker for their annual DoD Product Line Architecture workshop. Dr. Milks also performs operations research and experimental analysis. Recent work in this area includes his lead role as the experimental design, data collection, and analysis lead for DARPA's Real-time Adversarial Intelligence and Decision making (RAID) project. His support to RAID established the program as one of the most objectively evaluated DARPA projects in recent history. Dr. Milks holds a bachelor's degree in Mechanical Engineering from Ohio Northern University, an MS in Industrial Engineering from Texas A&M, and a PhD focused on Operations Research from the University of Central Florida.

Sandra Veautour is currently the Program Executive Office for Simulation, Training, and Instrumentation (PEO STRI) Chief Systems Engineer. As the PEO's technical force provider, she is responsible to ensure that the engineering and computer science personnel requirements of the Program Managers are adequately supported. Ms. Veautour is responsible for the education, training and career development of the PEO STRI technical workforce. In addition, she is the senior engineer in PEO STRI and is one of twelve Chief Systems Engineers in the Army. She defines and implements all technical policies and procedures for PEO STRI and its programs. Prior to her Chief Systems Engineer position, Ms. Veautour was the Assistant Project Manager and Chief Engineer for the Warfighter's Simulation (WARSIM) program under the Project Manager Constructive Simulation. Prior to her assignment in WARSIM, Sandy was the first Project Director on the OneSAF program. Prior to joining STRICOM, Ms. Veautour worked for two years at the Naval Training Systems Center as a facilities engineer supporting Army and Marine Corps training systems. Ms. Veautour has a Bachelors degree in Aeronautical Engineering from Embry-Riddle Aeronautical University and is a member of the Army Acquisition Corps.

Lessons Learned Assessing the Longevity of Simulation & Training Architectures

Wesley A. Milks, PhD
Applied Research Associates, Inc.
3452 Lake Lynda Drive, Suite 250
Orlando, FL 32817
wmilks@ara.com

Sandra Veautour
US Army, PEO STRI, Project Support Group
12350 Research Parkway
Orlando, FL 32826-3276
sandy.veautour@us.army.mil

INTRODUCTION

Architectural longevity requires attention to detail with respect to openness, modularity, reliability, performance, scalability, interoperability, and maintainability. The force driving these desired qualities is a combination of the functional and business requirements for the system under development. We begin this paper with a discussion of the challenges associated with architectural longevity. Second, we provide a summary of leading architectural assessment methods that address the general challenge. Next, we discuss tailoring of the general methods to the unique aspects of the military modeling and simulation community and a few example efforts with lessons learned. We close with a recommended best practice for integrating continuous architectural evaluation into the DoD acquisition life cycle.

THE CHALLENGE OF ARCHITECTURAL LONGEVITY

Traditionally, system architects focus their attention on the system's functional requirements. As a result, the stakeholders' business goals (e.g., modularity, extensibility, etc.) are frequently not met. Exaggerating the problem, the defense acquisition cycle involves multiple organizations who typically only participate in one or two of the acquisition stages (e.g., organizations performing concept refinement typically only perform concept refinement, the production and deployment contractor typically hands the system over to a support organization and moves on to the next opportunity). The challenge before the collective team is to create a means of consistently addressing the key functional and business requirements of the system through all stages of the acquisition lifecycle.

Why is it important?

One of the most cost-effective risk mitigation activities a system development team can perform is building confidence that the system's architecture is a proper fit for the intended purpose of the system. The Army continues to grow its dependence on complex software-intensive systems to support the ever changing and evolving battlefield. While this is challenging for all systems under development, developing software that is agile and flexible is especially challenging to the modeling and simulation community. This community must develop systems that support mission critical requirements quickly, efficiently, and with limited resources. Providing a solution that addresses only one need is no longer a viable way to fulfill the modeling and simulation (M&S) requirements of the Army. Designers must exploit innovative approaches to acquire, verify, and sustain software-intensive systems that provide stable, agile, and robust capabilities across the spectrum of M&S uses.

The Software Engineering Institute (SEI) at Carnegie Mellon University is well recognized and respected for their work related to capability maturity models. Their work has led to significant increases in the maturity of organizational software development processes and improvement in the quality of software intensive systems. They have also performed pioneering research that led to several well-proven methods for architecture design and evaluation.

Techniques from the SEI such as Architecture Tradeoff Analysis Methodology (ATAM™) and Quality Attribute Workshops (QAW) provide a common baseline from which all systems development activities flow and a framework for all stakeholders to use as a frame of reference as the system development progresses. Solid architectural foundations established at the inception of a project are key to providing solutions that ultimately deliver significant return on the initial architectural investment through reduction in total ownership costs (especially in the sustainment) of the system.

Why is it hard?

Most development teams find the task of architecture evaluation difficult due to a lack of education and training, lack of experience, and lack of knowledge of formal methodologies / processes that can help.

Lack of Education / Training

Architecture evaluation is not part of formal software engineering / computer science curriculum. Although there are courses that deal with creating software architectures, they do not address methods for formally *evaluating the goodness* of the architecture. Without the initial education basis, developers continue to be unaware of what they do not know.

Lack of Experience

The progression from computer scientist, to software engineer, to software architect typically relies on mentoring, hands-on experience, and on-the-job-training. Architecture evaluation is a relatively new concept. As a result, the "old salts" of the software engineering community have limited exposure to the concepts and are therefore unable to pass on knowledge and experience to the next generation. Additionally, the most experienced members of the various organizations involved in the acquisition process typically write the majority of acquisition strategies, proposals, and project plans. Since these individuals typically lack exposure to formal architecture evaluation throughout their careers and are therefore unfamiliar with techniques and associated benefits, they tend to limit, often to zero, the project resources (i.e., funding) allocated to formal architecture evaluation.

Lack of Knowledge of Formal Methodologies / Processes

Underlying the previous two points is the lack of knowledge of the formal methodologies, processes, and training opportunities that exist within the software development community. The SEI has not only led the research to develop the methodologies, they have also taken the initiative to develop training courses and formal certification programs associated with architecture development and evaluation (SEI (Training), 2008).

ARCHITECTURE ASSESSMENT METHODS

In this section, we provide a summary of leading architectural assessment methods that address the general challenge of achieving architecture longevity.

Architecture Tradeoff Analysis Methods (ATAM)

The SEI's Architecture Tradeoff Analysis Method (ATAM) is the leading methodology for evaluating software architectures (SEI (ATAM), 2008). One of the unique aspects of the ATAM is that it evaluates from a *stakeholders' perspective*, which maintains an external view of the architecture and focuses on achieving the desired architectural qualities without dictating design decisions to the architecture development team.

Conducting the ATAM process begins with generation of an ATAM Utility Tree similar to the example in Figure 1. The Utility Tree is a hierarchy of the *quality attributes*, *specific quality attribute issues*, and *evaluation scenarios*. The architectural quality attributes and associated specific issues are derived using the driving functional and business requirements of the system. The evaluation scenarios are analogous to use cases and range from *typical uses* (e.g., how the user will employ the system most frequently) to *anticipated changes* (e.g., new standards, emerging technology) to *exploratory changes* (e.g., disruptive technology, major shift in US foreign policy threats). Two ratings characterize each scenario. First, a *relative importance* rating supports comparison of alternate architecture designs. Second, a *relative difficulty* rating supports prioritization across the quality attributes. Each ranking is assigned a value of high (H), medium, (M), or low (L).

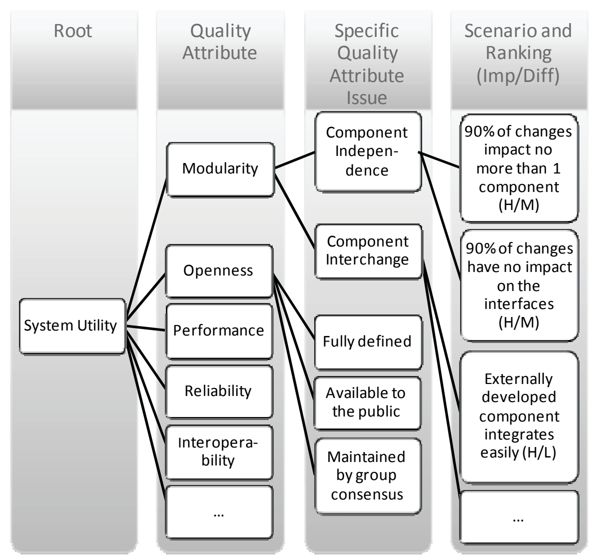

Figure 1. Example of an ATAM Quality Attribute Hierarchy. Quality attributes, issues, and evaluation scenarios provide an objective basis for understanding the impact of architectural decisions on functional and business requirements.

Once generated, the utility tree provides a basis for assessment of each candidate architecture / design alternative in light of the evaluation scenarios. The evaluation results in identification of *risks* (i.e., decisions that have not been made yet or ones that are not fully understood), *sensitivity points* (i.e., aspects of the architecture that are tightly correlated with measurable quality attribute response), and *trade-offs* (i.e., aspects of the architecture that host more than one sensitivity point that are affected differently) associated with each design alternative. The team can then consider the expected impact and corresponding risks when selecting the most appropriate design alternative.

The collective system development team (e.g., users, acquisition organizations, development contractors, etc.) can expect to see a significant range of benefits from the ATAM process for a relatively small investment. Benefits include; clarified requirements, improved architecture documentation, documented basis for architectural decisions, improved team communication, and enhanced risk identification and management.

Quality Attribute Workshops (QAW)

Quality attribute workshops (QAW) are a companion to the ATAM and provide a formal method for identifying critical quality attributes (e.g., availability, performance, security, interoperability, and modifiability) of a system and its underlying architecture (SEI (QAW), 2008). QAWs are facilitated exchanges between the various stakeholders associated with a project. These exchanges yield a set of prioritized scenarios that define the quality attribute requirements for the system driven by the system's business goals and typically form the basis for the ATAM Utility Tree. These scenarios also form the basis to conduct discussions about the system by providing a common frame of reference for all stakeholders to elaborate further system details.

Cost-Benefit Analysis Method (CBAM)

The Cost-Benefit Analysis Method (CBAM) is another companion to the ATAM that provides a formal method for making investment decisions associated with addressing the findings from the ATAM evaluation. At the conclusion of an ATAM evaluation, the team reports the risks, tradeoffs, and sensitivity points associated with the current of planned design alternatives. The project's management team / decision makers use the ATAM results to focus on project and architectural alternatives that can help reduce the identified risks. The CBAM takes the architectural decision analysis done during the ATAM and helps make it part of a strategic roadmap for software design and evolution by associating priorities, costs, and benefits with architectural decisions (Nord, et al., December 2003).

The ATAM identifies the set of key architectural decisions that are relevant to the key quality attribute scenarios elicited from the stakeholders. The decisions generate quality attribute responses (e.g., levels of availability, performance, security, usability, and modifiability, etc.). Each architectural decision has a cost. For example, designing in redundant hardware to increase availability has one cost (e.g., additional hardware, additional integration, etc) and periodically storing system state to an external file (i.e., checkpointing) has a second cost. Both alternatives will result in some expected level of availability with each level having some value to the stakeholders. Given the characterization of the impact of each design alternative on the system's quality attributes (i.e., the *benefit*) and the investment required for each alternative (i.e., the *cost*) the management team can make an informed decision based on their view of the expected return on investment.

In the same way that the ATAM does not make decisions for the system architects, the CBAM does not make decisions for the stakeholders. The CBAM provides a framework for identifying and documenting the costs, benefits, and uncertainty of a range of potential architectural investments.

Evaluating Service Oriented Architectures

Many new information systems, including military training and simulation systems, are moving to service-oriented architectures (SOA) as a design basis. The transition to SOA introduces new risks and challenges for the development team. SOA designs integrate components and applications through exposure of services rather than more traditional data-exchange based interfaces. The unknowns associated with embracing a new concept such as SOA, increases the risk of fully satisfying the stakeholders' quality attributes. For example, there is an additional challenge associated with assessing how an SOA implementation will affect performance, availability, security, and modifiability attributes since the evaluation team has a limited basis upon which to base their assessment (Bianco, Kotermanski, & Merson, September 2007).

Design considerations and tradeoffs for SOA implementations require a different approach and thought process. Basic SOA design includes *service provider* components (i.e., elements offering services to be used by others) and *service user* components (i.e., elements that invoke services provided by others). A component is not required to be of only one type (e.g., a service provider will likely also be a service user of services provided by other components). The service itself is generally self-contained, available over a network, self-defining / discoverable, and dynamically bound.

Evaluation of SOAs is unique in that it needs to focus on the integration of services via the architecture as opposed to the traditional focus on data exchange. However, service-level evaluation does not replace the requirement to evaluate the internal design of the service user and service provider components. The three essential steps of the evaluation process are still the same; (1) select the appropriate stakeholders, (2) specify the quality attributes, (3) develop an expressive and complete description of the architecture. Each of these steps requires tailoring for SOA applications (Bianco, Kotermanski, & Merson, September 2007).

Defining SOA Stakeholders

The nature of an SOA allows new components to join the collective system long after the SOA is specified and initially developed. As a result, it is not possible to generate an exhaustive list of architectural stakeholders early in the architecture development process. Therefore, two new roles are included in the stakeholder set for evaluation of SOA designs (Bianco, Kotermanski, & Merson, September 2007). The *service usage regulator* is responsible for creating and managing the policies, standards, and service level agreements to govern later component development. *External service providers* are responsible for representing the interests of existing external services that the system under development will access.

Specifying SOA Quality Attributes

With respect to the system's quality attributes, the selection of an SOA should not influence the selection of quality attributes, but it does affect how well some of the qualities will be satisfied. For example, the flexibility inherent in an SOA implementation will likely have a positive impact on interoperability and modifiability type qualities. However, the ability for new components to discover available services provides significant challenges to quality attributes related to system security and information assurance.

Expressive and Complete SOA Documentation

Finally, the nature of the underlying SOA concept (i.e., the interactions between a service user and service provider are manifested at runtime) introduces challenges associated with providing an expressive and complete description of the architecture. As a result, the module, deployment, and data model views typically associated with documenting an architecture are of limited value to understanding and assessing an SOA. Runtime views best capture a service-centric architecture. However, the runtime views are generally not effective conveying the structural aspects of the architecture that are critical to understanding performance and modularity qualities of the design.

TAILORING TO ARMY MODELING & SIMULATION

In this section, we discuss tailoring of the general methods to the unique aspects of the military modeling and simulation community from the perspective of the aspects that are unique to acquisition agencies and aspects that make evaluating training systems architectures unique from other types of information systems.

Aspects Unique to Acquisition Agencies

Several aspects of architectural evaluation are unique to Army acquisition that leads us to consider tailoring the architecture assessment methodology to best support our systems. First, the government program team needs to develop and execute a Software Architecture Evaluation Plan (SAEP) to comply with the program acquisition documentation that governs the execution of the program. The SAEP clearly lays out the specifics of the architectural assessment methods to employ during system development. Depending on the size and complexity of the program, it can be developed as a stand-alone document or the software architecture evaluation considerations can be included in other program acquisition documents (i.e., Acquisition Strategy, Systems Engineering Plan, Test and Evaluation Master Plan, etc.).

The second aspect is a deliberate inclusion of architecture assessment methodologies at program initiation in contractual requirements. Specific language and guidance must be included in the request for proposal, statement of work, and the final contract vehicle for the system. This language and guidance specifies the baseline requirements needed to ensure the system is designed with a solid software architecture as its foundation. This language includes specific requirements for ATAMs, QAWs, CBAMs, and delivery of software documentation artifacts to the government under the contract, frequency of reviews and workshops, and inclusion of the technical evaluation factors for contract award. Including these requirements in the program acquisition documentation allows the system development approach to be outlined in a manner that clearly describes the design and, more importantly, the risks related to the stakeholders' architectural qualities.

The third key aspect, and potentially the most crucial to success, is educating the workforce in architecture assessment methodologies and their value to system acquisition, design, and fielding. Government teams must understand the data provided to them via ATAMs, QAWs, CBAMS, etc. to be able to leverage this information in a consistent review and feedback framework for the development that affords a proactive versus reactive approach to managing the system development. This education process must include all stakeholders from the materiel developers to the combat developers to the testers.

Aspects Unique to Training Systems

Training system development should begin with a formal instructional system design activity to ensure proper understanding of the tasks to be trained, learning objectives, prerequisites, appropriate training media, and other training unique aspects of the information system. This needs analysis forms the basis for the development requirements of the training system and is the key to what distinguishes training systems from other information systems. Currently, evaluating the validity of the transition from training needs to training system requirements is a highly subjective and error prone process. Improper translation frequently leads to training systems that provide ineffective training. Even though the training system may fully satisfy the system requirements, if the requirements are a poor interpretation of the training needs the resulting system may be ineffective. Incorporating aspects of the training needs into the quality attributes as part of a more formal architecture evaluation effort offers opportunity to improve the effectiveness of the training system. The difficulty lies in how to reflect the training needs as quality attributes.

Cognitive performance indicators are a formal means of capturing and understanding the cognitive basis of an information system (Brown, Kosnick, & Cox, 2007). The cognitive performance indicators characterize the cognitive aspects of system using nine key indicators as listed in Table 1. Although these indicators were developed with an emphasis on evaluating the extent to which a user interface allows the user to perform their required tasks, they have applicability to characterizing the extent to which a training system enables opportunities to train and exercise the required training tasks. Not all indicators apply to all systems. However, the indicators do provide a basis for creating quality attributes associated with the training needs.

Table 1. Cognitive Performance Indicators for Information Systems (Brown, Kosnick, & Cox, 2007). The CPI provide a basis for characterizing the extent to which a training system enables opportunities to train and exercise the required training tasks.

Indicator	Description
Option Workability	Systems should enable users to quickly determine if an option is workable.
Cue Prominence	Systems should allow users to rapidly locate key cues from the information presented.
Direct Comprehension	Systems should allow users to directly view key cues rather than requiring users to manually calculate information to comprehend these cues.
Fine Distinctions	Systems should allow users to investigate or at least access unfiltered data.
Transparency	A system should provide access to the data that it uses and show how it arrives at the processed data.
Historic Information	Systems should capture and display historic information so that users can quickly interpret situations and diagnose problems.
Situation Assessment	Systems should help users form their own assessment of a situation rather than provide decisions and recommendations.
Directabiltiy	Systems should support the directing and redirecting of system priorities and resources so that users can effectively adapt to changing situations.
Flexibility in Procedures	Systems should allow users to modify the order of procedures as doctrine changes or situations call for flexibility.

EXPERIENCE AND LESSONS LEARNED

In this section, we provide a summary of lessons learned applying formal architecture evaluation to recent training systems projects and general experience integrating formal architecture evaluation into organizational processes.

Experience from One Semi-Automated Forces

The One Semi-Automated Forces (OneSAF) architecture was required by PEO STRI Program Manager for Constructive Simulation (PM ConSim) to reduce duplication of investments by eliminating the Army's need for multiple simulations across the Advanced Concepts and Requirements (ACR), Research, Development, and Acquisition (RDA), and Training, Exercise, and Military Operations (TEMO) M&S domains, foster interoperability and reuse, encourage collaborative development, and meet M&S requirements of the future force. The OneSAF architecture needed to meet a number of stringent quality goals, principal among these are flexibility, composability, platform independence, and ability to use 3rd party vendors.

In FY06 the SEI conducted an ATAM evaluation of the OneSAF program to explore how well the architecture met the program's business and quality goals (Batman, Blair, Gagliardi, Little, & Nord, August 2006). The intent of an ATAM effort is not to provide precise analyses; rather, the intent is to discover risks created by architectural decisions.

The results of this evaluation identified 25 risks which were distilled into four risk themes; System Resource Management, Additional Architecture Documentation for Developers, Analytical Underpinnings, Correctness and Verifiability. These risks have been addressed, monitored, and managed throughout the development of OneSAF. For example, in the area of architecture documentation for developers, the OneSAF program increased the robustness of the OneSAF.net web portal to provide detailed architectural artifacts for software developers.

PEO STRI Integration and Interoperability Initiatives

In late 2007, PEO STRI formally chartered an Integration and Interoperability Advisory Board (I2AB). The I2AB is an evolution of the Enterprise Architecture Steering Committee (EASC) and Program Management Board (PMB) from two separate reporting bodies into a joint leadership body providing guidance and direction in the fulfillment of PEO STRI's mission. As the primary technical and programmatic body within PEO STRI, the I2AB facilitates integration and interoperability (I2) programs and systems in support of the current and future customer base.

The I2AB has three focus areas: *program synchronization*, *interoperability*, and *common components*. Program synchronization provides a clear understand and purposeful management of the dependencies that result from implementing product lines and capability reuse across the PEO's portfolio of programs. Additionally, this information helps the PEO STRI leadership make informed decisions with an understanding of 2nd and 3rd order effects.

The focus on interoperability enhances the benefits of the Army's current training and testing products without having to develop new systems. This is key to leveraging the substantial investments that have been made in legacy M&S tools. Interoperable capabilities enhance expansion of training & test capabilities, enhance system realism, and enable functional expansion through the deployment of complex environments and more detailed physical and behavioral modeling.

The emphasis on common components is central to systematic reuse and product line management in the development of training and testing solutions for the Army. The more reuse PEO STRI employs in the development of its products and services, the less the development, testing, and life cycle cost to the Army, resulting in reduced total ownership costs and significantly increased return on the services for M&S investments, reduced testing resources and time, and significantly reduced support cost.

The end state of this key strategic initiative for the PEO is better training and testing for the Army and the nation, which is the chartered mission for the organization. I2AB efforts include incorporation of formal architecture evaluation processes to facilitate and manage achievement of the I2 vision. Organizational insight into the risks and tradeoffs of alternative strategies provides similar benefits as those seen within a specific program.

General Initiatives within PEO STRI

On August 12, 2002 the Assistant Secretary of the Army (Acquisition, Logistics, and Technology) ASA(ALT) approved the Army Strategic Software Improvement Program (ASSIP), which established broad strategic objectives for software acquisition. This program is a long-term effort focused on institutionalizing continuous improvement in the acquisition of software intensive systems. An ASSIP Action Group (AAG) was chartered to plan, coordinate, manage, and execute the ASSIP strategy. AAG members include Army PEO representatives, Army Materiel Command Software Engineering Centers' representative, ASA(ALT) and SEI.

The AAG documents specific initiatives, commitments, resource requirements, and milestones annually in the Strategic Software Improvement Master Plan (SSIMP). All Army PEOs, including PEO STRI, participate in the ASSIP. The PEO has developed and is executing a detailed Software Acquisition Improvement Plan (SAIP) specific to PEO STRI systems and activities. The PEO STRI FY 08 SAIP identifies four tasks to address during the fiscal year. Each task has a specific purpose, one or more actions, and one or more specific products. PEO STRI is helping to shape the strategic future of software development within the Army to include methodologies for evaluation of architectures for software-intensive systems.

Common Training Instrumentation Architecture

The Common Training Instrumentation Architecture (CTIA) is a PEO STRI Program Manager for Training Devices (PM TRADE) initiative to specify and implement a software product line architecture for a instrumented systems for live training and testing. The CTIA objectives are to conceptualize, design, develop, produce, modify, test, deliver, and sustain a product line architecture for instrumentation, services, and equipment. From an architectural development perspective, CTIA had numerous challenges including the need to maximize the commonality across the product family while also facilitating the variability that made each product unique.

While the overarching architectural construct was relatively stable from the beginning of the development effort, the design of the CTIA Services component, which provided the central data management and messaging services, matured through several iterations with each iteration offering significant variation from previous designs. Under normal circumstances, the significant changes between design iterations would generate numerous risks and concerns, if the evolution was even allowed to occur. However, early in the development process, the CTIA team invested in the generation of an ATAM Utility Tree that captured the quality attributes and key evaluation scenarios. As each design alternative for the CTIA Services matured, the team evaluated the new design in light of the quality attributes and associated scenarios. The new design alternative was then compared with current design allowing a decision to be made relatively quickly (i.e., within hours) due to the insight provided by the ATAM Utility Tree. If the ATAM process had not been used, the team estimated that characterization and selection of the design alternatives would have required days or weeks rather than hours.

RECOMMENDED BEST PRACTICE BY ACQUISITION PHASE

Our experience performing formal architectural evaluation has led to the basic conclusion that no matter when the effort is actually started, it should have been started sooner. Nothing defines the boundaries of a system development effort better than capturing the stakeholders' quality attributes and associated evaluation scenarios from the beginning of the project. The quality attributes tend to be easier to read and understand compared to trying to discern the system scope from a requirements document. Figure 2 shows our recommended practice for rigorously integrating

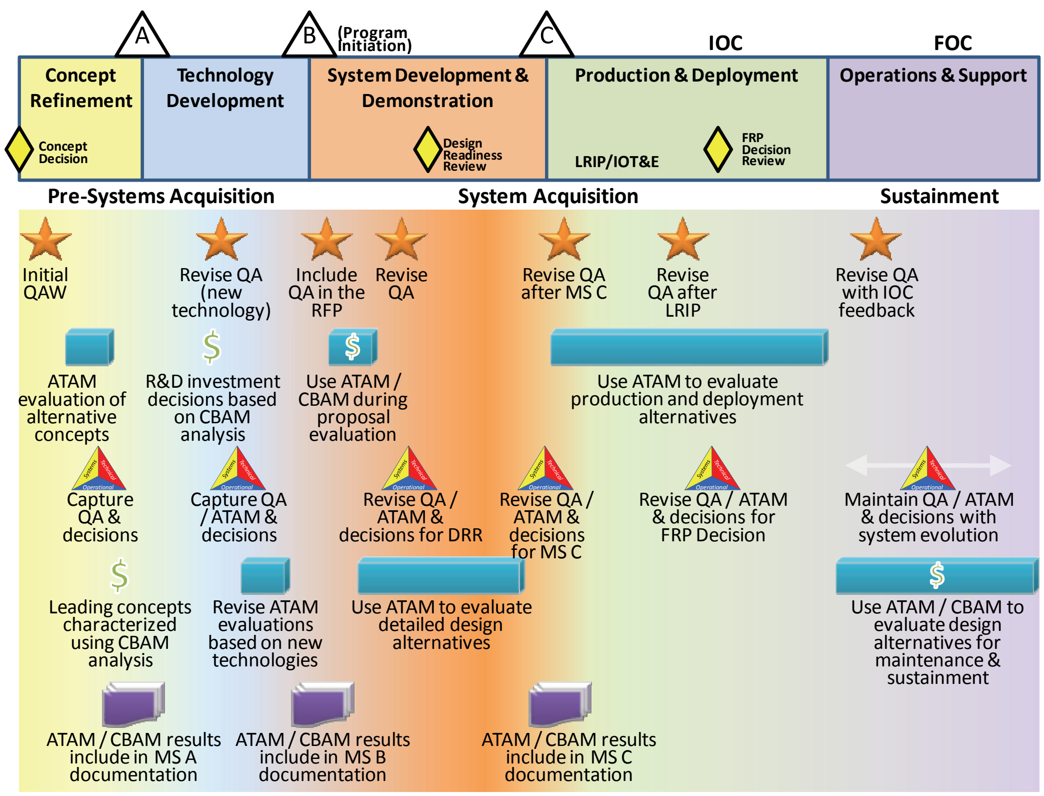

Figure 2. Recommended Best Practice for Overlaying QAW, ATAM, and CBAM Architectural Evaluation Activities on the DoD Acquisition Life Cycle. Our recommended best practice includes conducting a quality attribute workshop at the beginning of the concept refinement phase, numerous ATAM evaluations to identify risks with various design alternatives, documentation using DODAF views that are maintained throughout the project's lifecycle, CBAM evaluations to support investment decisions, and inclusion of the architecture evaluation results in the decision package for milestones A, B, and C.

architecture evaluation into the DoD Acquisition Life Cycle. We describe the recommend best practice activities associated with each life cycle phase in the following paragraphs.

Concept Refinement

Based on our experience, we recommend initiating the architecture evaluation effort early in the concept refinement phase. The development of quality attributes should begin near the onset of the concept refinement phase with a formal quality attribute workshop (QAW) held with the stakeholders. Although the quality attributes will be immature at this very early stage, even vague quality attributes will be critical to ensuring the system concepts focus on the highest priority aspects of the stakeholders' needs.

As alternative concepts evolve and mature, we recommend employing the ATAM process as a guide to characterize and evaluate each concept against the quality attributes. When used during the design portion of system development and demonstration phase, the ATAM evaluation will settle on a single design that best meets the quality attributes. However, during the concept refinement stage, it may be too early to settle on a single system concept. Applying the ATAM process here should focus on elimination of concepts that do not address the quality attributes well, leaving multiple viable concepts as potential alternatives.

A CBAM analysis should follow to understand the cost / benefit tradeoffs for the concepts that emerge from the ATAM evaluation. Although one concept may emerge from the CBAM as the most attractive (e.g., best value), significant assumptions will be required due to the extent of unknowns regarding each viable concept. Therefore, we recommend ranking the viable concepts based on the CBAM results. However, none of the concepts should be eliminated at this stage.

The generated quality attributes, ATAM evaluation results, and CBAM conclusions, to include all assumptions, must be well documented to be useful. We recommend incorporating the architecture evaluation documentation into DoDAF by adding new views. The quality attribute hierarchy should be added as a new operational view. When viewed as a hierarchy with a root node, quality attributes, specific quality attribute issues, and evaluation scenarios (see Figure 1), the quality attribute hierarchy is a decomposition much like an organizational chart. Therefore, we recommend capturing the quality attribute hierarchy using an organization chart diagram within the DoDAF development tools. The descriptions of the elements of the hierarchy and the results of the ATAM and CBAM evaluations, to include explicitly documenting all assumptions associated with both evaluations, should be captured as text documents that are included in the DoDAF set as new systems "views". The collective set of new views should also be included as part of the Milestone A decision package.

Technology Development

Following the Milestone A decision, we recommend revisiting the CBAM analysis and using it as the basis of research and development investment decisions. Each viable concept will likely require research and development activity to close gaps between current and required capabilities. Using the CBAM as the basis for the investment decisions will allow the investment to focus on the technologies that offer the highest probability to reduce the risks associated with each concept thereby maximizing the return on investment.

As new technologies emerge both from direct investment and others in the broader community, additional quality attribute workshops should be held to validate and / or revise the quality attributes. Similarly, we recommend revising the ATAM and CBAM analysis in light of the quality attribute evolution. Near the end of the technology development phase, we recommend conducting a formal ATAM evaluation with emphasis on how the technology development efforts influence the evaluation of each viable concept that emerged from Milestone A. At this stage, a leading concept should emerge and form the basis for entering Milestone B.

System Development & Demonstration

One of our most significant recommendations is to integrate the quality attribute hierarchy into the request for proposal (RFP) materials. We believe this will benefit all organizations involved in the RFP process. Specifically, the contractors responding to the RFP (e.g., bid decisions can be made quickly based on the capability to address the needs as expressed in the quality attributes, proposals can be properly scoped to the highest priority system needs, etc.) the acquisition agency (e.g., all proposals will have a common focus, deriving the technical factors for the proposal evaluation from the quality attributes provides a consistent / objective basis for evaluating different technical approaches, etc.), and the users / stakeholders (e.g., highest probability of having their real needs met) should all benefit from exposure to the quality attributes. Figure 3 provides details on SEI's view of incorporating the architecture evaluation methods into the RFP and proposal evaluation process.

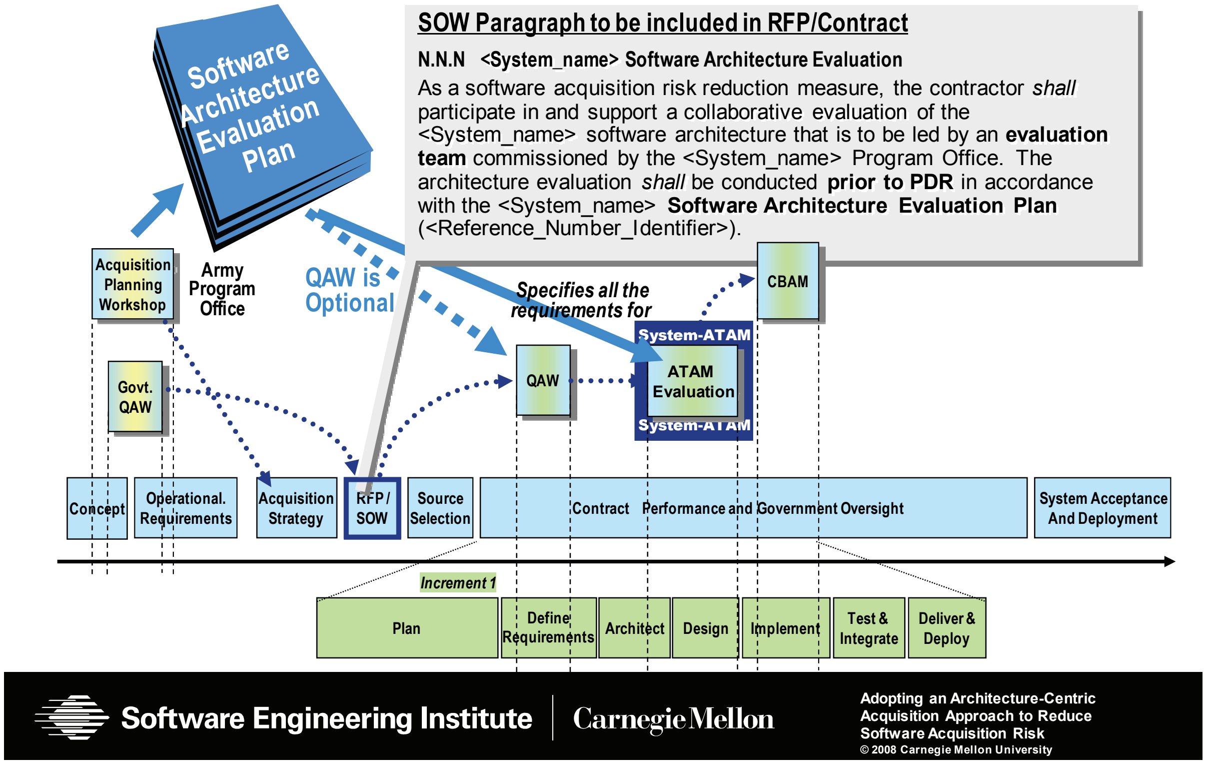

Figure 3. SEI Perspective on Incorporating Architecture Evaluation in an RFP / Contract.

Additionally, we recommend including tasks in the statement of work that requires the development organization to use a formal architecture evaluation process as part of their design and development efforts. Specifically, the architecture evaluation process should form the basis of design decisions, set the conditions / standards for all design reviews, and be incorporated into the system and acceptance testing plans.

The DoDAF views created during the previous stages should be handed over to the development contractor for maintenance and revision with the final set of views being included in the Milestone C decision package.

Production and Deployment

As the system moves into Production and Deployment, we recommend continuing to use the quality attributes in conjunction with the ATAM and CBAM processes as the basis for design changes and system evolution. Despite the best efforts during the System Development and Demonstration phase, design changes during production are inevitable. Basing the design evolution on the same architecture evaluation process and existing work products allows for consistency and continued focus on the stakeholders' needs.

While the architecture evaluation products should be maintained throughout the effort, conclusion of the low-rate initial production (LRIP) / initial operatonal test and evaluation (IOT&E) effort and the full rate production review offer specific opportunities to revise the documentation and capture new decisions.

Operations and Support

The system design will continue to evolve as feedback is received from the fielded systems during the Operations and Support phase. We recommend that the architecture evaluation work products be transitioned to the logistics support and maintenance team for continued evolution. As P3I efforts and other modifications are made to the system, the logistics contractor should continue to use the ATAM and CBAM as the basis for decision-making.

CONCLUSION

This paper has offered a summary of methods for architectural assessment, lessons learned applying the methods, and a recommended best practice for new standard methodology for integrating architectural assessment into the military simulation and training community through the DoD Acquisition Life Cycle. Following the recommended practices will require investment of time and key program resources to ensure the validity of the quality attributes and associated analysis efforts. However, the payoff is terms of providing a materiel solution that satisfies the user's need is substantial and the return on investment is significant. Tailoring the SEI techniques to the unique aspects of the military simulation and training community opens the potential for improving training, enhancing analysis, and reducing support costs for delivered systems while also providing a path to true architectural longevity.

REFERENCES

Batman, J., Blair, C., Gagliardi, M., Little, R., & Nord, R. (August 2006). *OneSAF Architecture Evaluation Report.* Carnegie Mellon University. Pittsburgh, PA: Software Engineering Institute.

Bianco, P., Kotermanski, R., & Merson, P. (September 2007). *Evaluating a Service-Oriented Architecture.* Software Architecture Technology Initiative. Pittsburgh, PA: Software Engineering Institute.

Billingsley, P. (1999). *Convergence of Probability Measures.* New York: John Wiley & Sons, Inc.

Brown, J., Kosnick, L., & Cox, D. A. (2007, October 1-5). Enhancing an Application of Dynamic Management of System Capacity using Cognitive Assessment Indicators. *Paper presented at the Human Factors and Ergonomics Society 51st Annual Meeting* .

DARPA. (2008). *Real-Time Adversarial Intelligence & Decision Making (RAID)*. Retrieved June 11, 2008, from DARPA - Information Processing Techniques Office Web Page: http://www.darpa.mil/ipto/programs/raid/raid_vision.asp

Driels, M. R. (2004). *Weaponeering: Conventional Weapon Systems.* American Institute of Aeronautics and Astronautics (AIAA).

Kechris, A. (1995). *Classical Descriptive Set Theory.* Graduate Texts in Mathematics 156.

Nord, R. L., Barbacci, M. R., Clements, P., Kazman, R., Klein, M., O'Brien, L., et al. (December 2003). *Integrating the Architecture Tradeoff Analysis Method (ATAM) with the Cost Benefit Analysis Method (CBAM).* Pittsburgh, PA: Carnegie Mellon University / Software Engineering Institute.

SEI (ATAM). (2008). *Architecture Tradeoff Analysis Method*. Retrieved June 17, 2008, from Software Egnineering Institute: http://www.sei.cmu.edu/architecture/ata_method.html

SEI (QAW). (2008). *Quality Attribute Workshops*. Retrieved June 17, 2008, from Software Engineering Institute: http://www.sei.cmu.edu/architecture/qaw.html

SEI (Training). (2008). *Software Architecture Curriculum*. Retrieved June 17, 2008, from Software Engineering Institute: http://www.sei.cmu.edu/architecture/arch_curriculum.html