System-of-Systems Influences on Acquisition Strategy Development

Rita Creel

Robert J. Ellison

June 23008

ABSTRACT: An *acquisition strategy* is a top-level roadmap that focuses on highlighting and managing risks to a successful outcome. Business requirements for supporting work processes require integration across multiple systems, spanning multiple business or organizational units. This operational context incorporates significant new sources of risk, such as a large and diverse user community and increasing uncertainty and complexity, that become key drivers for acquisition strategy development and execution. Some recommendations for addressing these challenges in an acquisition strategy are discussed.

INTRODUCTION

Systems acquisition has historically focused on purchasing or outsourcing the development of a system that operated in a stand-alone fashion or had few interactions with other systems. The acquisition process typically concentrated first on the functionality required and then on monitoring development and reviewing products to ensure the required functionality was provided. The underlying assumptions were that requirements were static and that acquisition should concentrate on monitoring costs and schedule. Acquisition strategies and plans were developed with these assumptions in mind.

Today, an increasing number of acquisitions are focused on systems that are intended to function as components within a larger system-of-systems (SoS) context. At the same time, business and mission needs for SoS constituents continue to evolve, and users expect an ability to adapt their systems accordingly. Along with providing new or modified capabilities, a system may need to communicate with other systems that were not identified up front. In such an environment, it becomes critical to specify requirements related to assurance goals and to build in the qualities needed to enable correct operation in the midst of a high degree of complexity and change.

Acquisition of a system that is intended to function within an SoS has thus become a dynamic integration problem, a scenario in which the job of eliciting and communicating requirements, understanding system interfaces and usage patterns, and refining assurance strategies is never quite finished. For software, this

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-2612

Phone: 412-268-5800
Toll-free: 1-888-201-4479

www.sei.cmu.edu

has led to a growing gap between expectations and practice—in particular, the scope and complexity of socio-technical domains that are the context for many SoS creates challenges in the identification of requirements [Easterbrook 2007]. The Easterbrook reference includes a link to a later presentation of his keynote address, which is an excellent introduction to how the scope of an SoS affects a software development life cycle.

The security world is constantly evolving, and assumptions about certain kinds of vulnerabilities can change overnight [Howard 2007]. As discussed in this paper, an acquisition done in an SoS context has to consider a wider spectrum of failures and mitigate the effects of changes in usage, technology, and in the individual components. Unanticipated interactions among systems can induce degraded service. Howard emphasizes a critical lesson that most vendors have learned the hard way: Today's denial of service is tomorrow's exploit [Howard 2007]. As a consequence, security now needs to also consider general system failures that usually have been associated with reliability.

The realities of system acquisition in an SoS environment, therefore, place significant new demands on the acquisition strategy, which must be robust against considerable change. This article discusses SoS influences on developing an acquisition strategy. The article introduces acquisition strategy development, identifies key challenges for acquisition in an SoS environment, and provides recommendations for building an acquisition strategy to help cope with these challenges.

ACQUISITION STRATEGY DEVELOPMENT

An acquisition strategy is a top-level roadmap used to guide acquisition toward a successful outcome in terms of cost, schedule, delivered capability, and quality. It spans the life cycle from initiation of acquisition through sustainment and is focused on highlighting and managing risks to a successful outcome.

Developing an acquisition strategy is a key component of acquisition planning. The DoD acquisition process explicitly requires an approved acquisition strategy at program initiation [DAU 2012], but the concept is equally applicable to large-scale commercial acquisitions. To ensure its continuing relevance and usefulness, the acquisition strategy is updated based on major decisions, program milestones, emerging risks, and other events that occur during acquisition.

Figure 1, from Ward et al., illustrates the relationship between the acquisition strategy and the acquisition plan and other artifacts it guides [Ward 2006]. As shown, a robust acquisition strategy works to mitigate and reduce risk during acquisition, in concert with other risk management and mitigation processes.

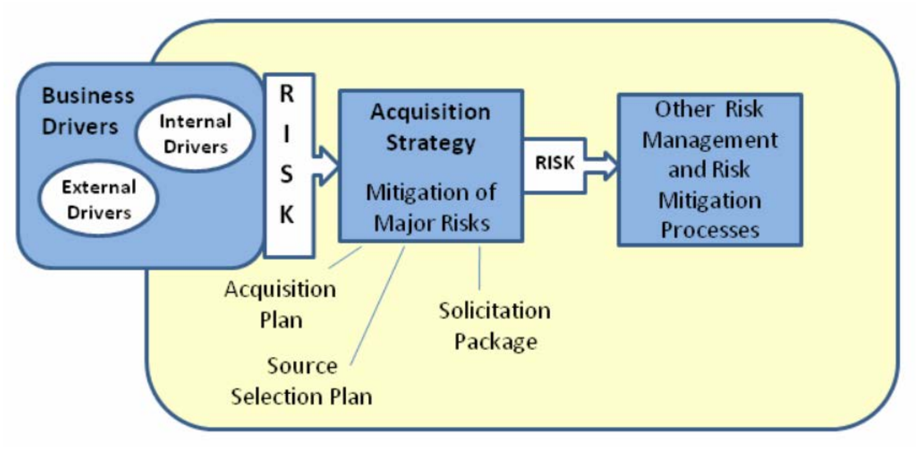


Figure 1. Relationships among drivers, risks, and acquisition strategy [Ward 2006]

Considerable analysis and collaboration are required to develop a sound acquisition strategy. For DoD programs, the acquisition strategy is constructed by analyzing a number of strategy considerations (also called strategy elements). As an example, Table 1, adapted from the Defense Acquisition Guidebook, lists key considerations for a DoD acquisition strategy [DAU 2012]. Many of these considerations are also applicable to commercial acquisitions.

Table 1. Acquisition strategy considerations (or elements) [adapted from DAU guidebook, DAU 2012]

Strategy Element	Definition and Example Choices
Acquisition Life Cycle Model	Identifies the life cycle model the acquirer will use to achieve full capability, and tailors the model to the acquisition program's specific needs. Example options include "single step" (e.g., waterfall), incremental (phased development and deployment of defined capability), and spiral (evolutionary definition, development, and deployment).
Business Considerations	Explains the business strategy and constraints, providing information on competition, collaboration, regulatory and statutory concerns, solicitation and supplier selection approach, etc. Example considerations that might be part of a business strategy include decisions on the desirability of commercial products versus custom-developed hardware and software, or in-house or externally contracted development. Supplier selection could be competitive or single source.
Applicable Best Practices	Defines the approach to select and apply appropriate development and acquisition best practices and to review best practices at each decision point; identifies initial set of best practices to apply. As examples, best practices for acquisition might include practices for developing and managing requirements, program planning, supplier monitoring, and risk management.

Requirements Management	Defines the approach to be used to elicit, refine, control, and resolve conflicts with respect to requirements. Identifies requirements tools and may describe expected structure, content, and specification rules for requirements.
Test and Evaluation	Defines a test and evaluation strategy that is integrated with activities throughout the life cycle. Test and evaluation activities should be designed to provide information about risk, provide empirical data to validate models and simulations, evaluate technical performance and system maturity, and determine whether systems are operationally effective, suitable, and survivable against threats.
Risk Management	Establishes a risk management approach to identify and manage uncertainties in achieving program goals and objectives within performance, cost, and schedule constraints. Identifies initial risk areas and describes associated risk mitigation plans. Encompasses risk identification, analysis, mitigation planning, mitigation plan implementation, and tracking; is integrated with systems engineering and program management; and includes risks to acquisition and operations.

Acquisition Strategy Drivers

The development of a successful acquisition strategy depends on an understanding of the internal and external factors that will drive the acquisition. It is important to have a thorough understanding of capabilities and needs to be met by the acquired system (as known today and as it will likely evolve), system stakeholders, resource and schedule constraints, supplier capabilities, performance and quality expectations, and operations and sustainment concepts—including interaction with other systems and participation as an SoS constituent.

These factors need to be assessed for risks they may carry. We call the most significant factors acquisition strategy drivers. The corresponding acquisition strategy elements should be configured to mitigate the risk represented by these drivers.

For example, a large and diverse set of stakeholders, conflicting objectives among stakeholders, and unstable requirements can all create acquisition risks which may drive choices for one or more strategy elements. In this example, an evolutionary acquisition life cycle model would likely be chosen to accommodate change and to emphasize the need for risk reduction activities related to unstable requirements. An acquisition strategy business consideration for a system with significant technology risk may be inclusion of a technology development contract with multiple suppliers, one of which will be chosen to move forward to systems development. The acquisition life cycle model for development of this system may include activities and decision points related to prototyping or demonstrating the new technology.

Typical Acquisition Strategy Drivers

Acquisition planning must acknowledge a variety of risks and their impact on acquisition strategy elements. Ward et al. developed a taxonomy of driver categories for software risks along with an approach and tool to assist in developing an acquisition strategy that is robust against risks represented by these drivers [Ward 2006]. Other categories and drivers can be identified depending on the particulars of a given acquisition.

As examples, typical acquisition strategy drivers might fall into one or more of the following categories:

- Environmental: Characteristics of the acquisition environment, which may include, for example, regulations, supplier availability, and acquirer and supplier capability
- Stakeholder: Characteristics of the user community and other stakeholders for a system or SoS
- Business: Business parameters, such as product and contractual requirements, market factors, supplier availability and expertise, funding, and schedule
- Organizational: Characteristics of the acquisition organization, such as how staff are assigned and incentivized, and staff turnover rate
- Engineering: Characteristics of acquirer and supplier technical activities for defining, developing, verifying, and deploying the system
- Operations and Sustainment: Concepts for operations and sustainment, and expectations of change in usage, including eventual disposal

The next section will discuss how membership in an SoS increases the significance of specific acquisition strategy drivers and introduces new ones. Note that the critical role of software in an SoS raises the importance of software behavior, and thus of software assurance, for the delivered product.

EMERGING STRATEGY DRIVERS FOR ACQUISITION IN AN SOS ENVIRONMENT

Acquiring large, software-intensive systems has long been a challenge, one that continues to grow as new technologies are employed in a bid to meet ever greater expectations for system capability and performance. Figure 2 represents the context for acquisition drivers in the SoS environment.

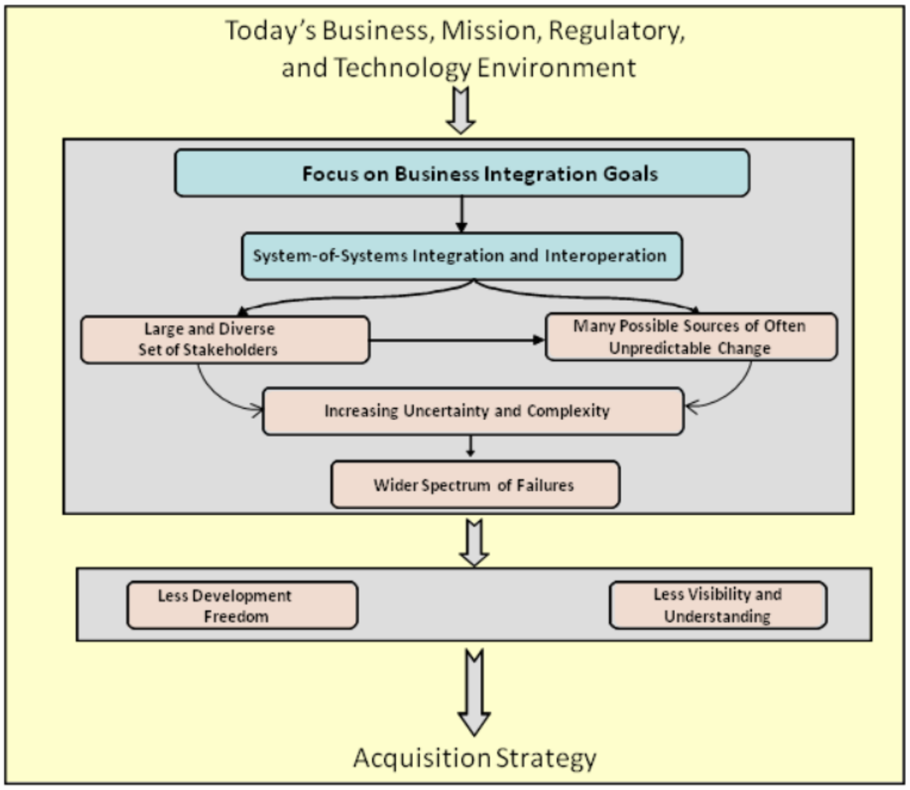

Today's Business, Mission, Regulatory, and Technology Environment
Focus on Business Integration Goals
System-of-Systems Integration and Interoperation
Large and Diverse Set of Stakeholders
Many Possible Sources of Often Unpredictable Change
Increasing Uncertainty and Complexity
Wider Spectrum of Failures
Less Development Freedom
Less Visibility and Understanding
Acquisition Strategy

Figure 2. Acquisition strategy drivers for systems of systems

Business requirements for supporting work processes require integration across multiple systems, essentially an enterprise SoS, spanning multiple business or organizational units. This context introduces significant new sources of acquisition risk. Among these sources of risk, which have become key drivers for acquisition strategy development and execution, are the following:

- Large and diverse stakeholder community: A large number of often-diverse stakeholders, some of whom may not be known at the start of acquisition
- Many possible sources of often-unpredictable change: Potential for change from any direction (i.e., from any SoS stakeholder or constituent system, as well as from evolving business requirements) at any time
- Increasing uncertainty and complexity: Less predictability regarding stakeholder needs, technology advances, and component behavior in an environment with no central control; more complexity in terms of the number and variety of constituents able to influence the SoS; and the number, variety, and volatility of SoS components
- Wide spectrum of failures: Failures with causes or impact beyond the individual system boundary, reflecting uncertainty and complexity with respect to external systems and stakeholders

- Limited development freedom: An existing system of systems represents a collection of design choices that can constrain new development and evolution.
- Limited visibility and understanding: Limited knowledge of individual system status and behavior, particularly as work processes cross organizational boundaries and are implemented by multiple interoperating systems with different owners

Given this list of drivers, it is clear that the acquisition strategy must motivate development of a system that will be responsive to necessary change while assuring dependability and supportability of critical services. This heightened need to focus on adaptability and assurance, to emphasize a robust architectural foundation, and to expand the scope of risk management beyond the system boundary will heavily influence the acquisition strategy. We will discuss some of these influences later in this article, but first we will explore the significance of the strategy drivers identified above.

Large and Diverse Stakeholder Community

The evolution from stand-alone systems to systems with a few known interfaces to systems of systems increases the number and diversity of stakeholders. This increase, along with the potential volatility of the stakeholder community, results in conflicts of interest and understanding that may be very difficult to resolve, creating technical and management problems.

The most obvious stakeholder conflicts surround priorities for required capabilities and assured system qualities. For example, conflicts may arise as systems are integrated across business units within or across organizations. One unit's work process may require a capability for remote usage, which might represent a security risk for one of the other units' work processes. The technical challenges to accommodate both requirements are often difficult to negotiate. As another example, when components are reused or services shared as with service-oriented architectures, different component or service users may have conflicting requirements.

Stakeholders are beginning to demand high levels of assurance for system qualities such as safety, security, reliability/availability/maintainability, performance, adaptability, interoperability, usability, and scalability. The stakeholders for an SoS may disagree on which quality attributes take precedence, as in the remote usage and security risk example above. In such an environment, a successful acquisition likely means that no one is completely satisfied, but everyone gets something they need and can use.

So in considering a large and diverse stakeholder community, one can see that an acquisition in the SoS context has to deal with a complex set of tradeoff relationships. The acquisition strategy, therefore, must incorporate a robust approach to stakeholder management that accommodates conflict resolution, risk management, and incrementally revisiting agreements among stakeholders.

Many Possible Sources of Often-Unpredictable Change

An SoS has been described as a collection of independently managed and operated systems which are geographically distributed, exhibit emergent behavior, and are developed using an evolutionary approach [Maier 1998]. In today's world, business and mission systems are expected to adapt to market changes and changes in the world environment. In an environment with components independently managed and operated, adaptations one constituent makes to respond to change may result in unintended side effects, not only to the constituent system but to other systems as well.

The acquisition strategy is heavily influenced by the call for adaptability, the need to respond to change. First, requirements development and management processes must be constructed to negotiate and establish key infrastructure and quality attribute requirements early on and enable capability requirements to evolve over the life cycle. Second, assurance practices that work through development and into operations and sustainment must be built and tested during acquisition. Finally, risk management processes must actively look for the next change both within and beyond system boundaries, and incorporate agile analyses of impact.

Uncertainty and Complexity

A potentially unbounded stakeholder community, the number and diversity of components, systems, and services to be integrated, and evolution in both stakeholder needs and system configurations create unprecedented levels of uncertainty and complexity. It is virtually impossible to understand everything about an SoS, let alone to influence all decisions made on behalf of its constituent components.

With respect to uncertainty, technology choices, unadvertised changes, and unanticipated system interactions may increase the chances of failure. For example, technologies such as Web services make it easier to assemble systems, but ease of assembly may only increase the risk of deploying systems whose behavior is not predictable. Business requirements increase the likelihood of failure by bringing together incompatible systems or by simply growing beyond the ability to manage change. An increasing number of failures are caused by unanticipated interactions between SoS components. Failures may be the result of discrepancies between the expected activity and the actual behavior that occurs normally in business processes. Often, such discrepancies are introduced by changes in business processes and systems. Consequently, the overall success of a business process depends on how these discrepancies are dealt with by staff and supporting computing systems.

Concerning complexity, fairly simple computing architectures that could be understood and their behavior characterized have been replaced by distributed, interconnected, and interdependent networks. As we depend more on complex, interdependent systems, failures are not only more likely but also harder to identity and fix as the number of participants—people and systems—increases. Each participant has to deal with multiple sources of discrepancies, and a single discrepancy can affect multiple participants. A poorly managed discrepancy, and the changes made to resolve it, may result in new discrepancies affecting additional participants. Failures are frequently the result of multiple, often individually manageable errors that collectively become overwhelming.

While we may have incomplete understanding and little control over the total SoS environment, the acquisition strategy can incorporate activities to help increase understanding and mitigate the impact of complexity by focusing on the acquired system's quality attributes.

Wider Spectrum of Failures

It is clear that the move toward systems of systems is increasing business and government use of software at unprecedented levels of scale and complexity. Software is, indeed, the mechanism that enables systems of systems to function. Add to this the move toward decentralization and the pace at which business and mission requirements change, and a great deal of uncertainty results regarding both the configuration of the SoS at any given time and the behavior that can be expected by its constituent components. A quote by Peter Neumann in a 2007 New York Times article captures the problem. "We don't need hackers to break the systems because they're falling apart by themselves" [Schwartz 2007].

While the individuals interviewed for the New York Times article included a number of well-known computing security experts, the general observations focused more on the underlying complexity than on security.

- Most of the problems we have today have nothing to do with malice. Things break. Complex systems break in complex ways.
- Simpler systems could be understood and their behavior characterized, but greater complexity brings unintended consequences. Problems are increasingly difficult to identify and correct with the shift from stand-alone systems to interdependent systems.

- Business usage requires change, but such change increases complexity by attempting to integrate incompatible computer networks or increasing the scale and scope of systems beyond the ability of the current capabilities to manage and sustain.

Hence, requirements for software assurance and other quality attributes related to software dependability and supportability need a strong emphasis. Historically, except for safety-critical systems and systems controlling financial transactions, efforts to build in these quality attributes have had much lower priority than efforts to develop functionality. This must change, but it will change only when the acquirer incentivizes performance with respect to assurance and quality requirements, not just cost, schedule, and functionality.

The acquisition strategy must incorporate activities to scan for and evaluate risks from a variety of potential sources of failure throughout the life cycle. In addition, the strategy must emphasize system quality attributes essential for robust operations. From an assurance perspective, changes must be analyzed and carefully managed to ensure they do not introduce vulnerabilities, degrade the quality of critical services, or lead to infrastructure fragility over time. This is no small task, and carrying it out effectively will require an acquisition, operations, and sustainment focus on tradeoff analyses that consider needs and risks across the SoS.

Limited Visibility and Understanding

The complexity and evolving nature of an SoS limit the ability to fully identify risks, understand the consequences, and analyze mitigations in advance of the start of development. Requirements will be incomplete. In addition, the use of COTS components, legacy systems, and independently developed and managed systems limits full system understanding.

Large projects typically encounter limited understanding of the problem domain in the initial development phases, but with systems of systems there may be no means to effectively reduce the uncertainties. A system owner will typically have knowledge for only those systems that they manage and not for any external systems. An objective for a technology such as Web services is to reduce the coupling among systems. A caller of a service is only aware of the system attributes that are available in the published interface. The provider of a service for an SoS may encounter changes in usage patterns that invalidate design assumptions, leading to unanticipated service behavior. Other users of that service won’t know the cause of service failure, which limits the ability to mitigate the impact or prevent such failure in the future. Likewise, the service provider may not be aware of the impact of any failure on current users, which may also limit effective mitigation on the provider's part.

Limited Development Freedom

Software is touted for its flexibility in terms of meeting requirements, but that flexibility is fully available only at the start of development and only to the extent that the environment allows. An acquisition in an SoS context rarely means clean slate development, as the SoS itself is usually an existing operational system that the newly acquired system has to join. Thus, SoS development has a sustainment flavor. While many current monolithic systems have similar constraints due to COTS use or legacy software, the constraints associated with an SoS arise from the collection of decisions that may have been made independently for each constituent system.

To deal effectively with limited development freedom, the acquisition strategy must identify known constraints on architecture and design that result from SoS membership, such as the existence of legacy or commercial components or services that are expected to be incorporated into the system. Where existing components or services are to be used, rigorous analyses must be conducted to ensure suitability for the intended purpose, both in terms of capability and quality attributes.

Now that we have reviewed some key acquisition strategy drivers for systems of systems, we will discuss how these drivers influence the acquisition strategy in general. Then we will examine their impact on two foundational acquisition strategy elements, the acquisition life cycle model and the risk management approach.

FRAMING AN ACQUISITION STRATEGY TO MEET THE CHALLENGES

The effectiveness of an acquisition strategy depends on the ability to make informed choices with respect to each relevant strategy element, integrate the elements into a cohesive whole, and document and communicate the strategy in a way that facilitates acquisition planning, monitoring, control, and decision making. Choices for acquisition strategy elements should be made through a careful analysis of the drivers specific to the acquisition. For an SoS constituent, a set of relevant drivers was discussed above. This is a generic, partial set: while there are commonalities across acquisitions, each acquisition may have unique characteristics that affect its set of drivers.

Table 1 identified an example set of strategy elements relevant for an acquisition program. In this section, we discuss two strategy elements that are central to successful acquisition of an SoS with assured behavior, the acquisition life cycle model and the risk management approach. We have chosen these two elements because of their prominent and continuous role in planning for and carrying out the technical and management activities of a complex acquisition.

First, in Table 2 we summarize the implications SoS acquisition drivers have for the acquisition strategy. These implications will translate to requirements to be satisfied by the acquisition life cycle model and risk management approach strategy elements.

Table 2. Implications of SoS strategy drivers

Strategy Driver for SoS	Implications and Requirements for Acquisition Strategy
Focus on Business and Mission Integration Goals	Ability to • understand integration goals and changes in these goals over the course of the acquisition life cycle • articulate assurance and other quality (dependability/supportability) goals for user communities and tasks • measure and demonstrate progress in meeting integration goals • identify a "successful" integration
SoS Integration and Interoperation	Ability to • understand interface particulars of systems to be integrated • identify degree of uncertainty and to tolerate uncertainty • accommodate change in constituent systems and interfaces • understand impact of constituent systems' behaviors on integration goals and assurance and other quality attribute goals (e.g., for dependability/supportability) • specify assurance and dependability/supportability requirements • estimate (a) cost and calculate cost risk, (b) schedule, (c) expertise and other resource requirements • conduct trade studies that consider cost, schedule, capabilities, and quality attributes
Large and Diverse Set of Stakeholders	Robust approach to stakeholder management with ability to • identify current and potential stakeholder communities • facilitate tradeoff analyses that consider functionality, cost, schedule, and assurance and other quality attributes • manage expectations, handle interactions, and resolve conflicts • accommodate new stakeholders/stakeholder sets • incrementally revisit agreements among stakeholders

Many Possible Sources of Unpredictable Change	Robust foundation that can accommodate change: • requirements processes constructed to establish key infrastructure and assurance and other quality attribute requirements early and enable capability requirements to evolve over the life cycle • assurance practices built and applied in acquisition and development that can work through operations and sustainment • risk management processes that actively look for the next change both within and beyond system boundaries, and incorporate agile analyses of impact • mature change identification and management process, including proactive and predictive activities • ability to distinguish between types of change, e.g., necessary change, desirable change, and destructive change
Uncertainty and Complexity	Strong and continued emphasis on identifying, understanding, and managing system impact related to areas of uncertainty and complexity Focus on • specifying quality attributes that are central to dealing with uncertainty and complexity • how quality attributes themselves are affected by uncertainty and complexity
Wider Spectrum of Failures	Robust activity to define failure modes with respect to assurance and other quality attribute requirements as well as functional requirements Early (i.e., during concept design) and continuous focus on failure modes and effects, linked with tradeoff analysis to balance assurance and other quality attributes against needs and risks Approach to scan environment for changes leading to new failure modes, vulnerabilities, and other potentially adverse effects of change Approach to identify, define, and manage assurance activities related to circles of control, influence, and concern for the acquisition • Control – self-protection: assurance external inputs will not cause internal system failure • Influence – co-constituent protection: assurance no output harms another constituent if used per agreement • Concern – SoS protection: basic SoS governance tenets applied to preclude avoidable failures Evaluate the costs and risks of failure against the costs of quality and assurance activities
Less Development Freedom	Understanding, documentation, and communication to stakeholders of constraints and potential impact of constraints, for example, reliance on legacy or commercial components and impact on achievable capabilities and levels of assurance

Less Visibility	Understanding, documentation, and communication of what cannot be seen/known; preventive action and risk management for these “known unknowns” Possible investments in (but not reliance on) modeling and simulation advances in an attempt to elicit information about the unknowns

The above implications should be used to guide choices for individual acquisition strategy elements. We illustrate this process for the acquisition life cycle model and risk management approach strategy elements below.

ACQUISITION LIFE CYCLE MODEL

The acquisition life cycle model element defines the general acquisition life cycle model to be used and describes how it will be applied to the acquisition at hand. It describes major life cycle phases, activities, reviews, milestone and interim decision points, deliverables, and other such information. To make the life cycle model relevant to stakeholders, major participants, roles, and responsibilities should be identified to the extent possible.

Considerations for Systems of Systems

For systems of systems, the choice of acquisition life cycle model is influenced by the drivers and implications identified in Table 2. Thus, the acquisition life cycle model and associated activities, processes, and decision structures must be developed to address the following characteristics:

- Multiple owners and stakeholders to manage
    - Deal effectively and agilely with a large number of stakeholder interactions and issues of scale, complexity, and uncertainty.
    - Incorporate effective information management and communications strategies that work internally as well as with other SoS components.
    - Establish and apply effective stakeholder management mechanisms, including explicit points for agreeing to move forward.
- Evolving user needs and requirements understanding – high volatility
    - Continually elicit information on needs and the business and mission environment.
    - Accommodate changes in needs, requirements, technologies, and interfacing systems.
- Large-scale or unprecedented development and integration

    - Emphasize the need to conduct tradeoff analyses throughout the life cycle that asses the relative importance of a variety of system capabilities and features, and assurance requirements and other quality attributes.
    - Develop an acquisition environment and encourage a development environment that support efficient, adaptation-friendly artifacts and processes.
    - Develop, analyze, and use indicators of progress and quality and movement toward achievement of acquisition project goals.
- Interim operational deliveries
    - Ensure that the life cycle model incorporates sufficient integration and test resources and plans and, where needed, transition equipment, to facilitate interim deliveries and operator training.
    - Ensure that iterations are designed to provide meaningful functionality while at the same time focusing early on “hard problems.”
- External technology impact and emergent behavior
    - Focus early on developing (and sustaining throughout the life cycle) a robust, assured, and adaptable architectural foundation and system infrastructure.
    - Identify and manage constraints and changes due to legacy and commercial components.
- Explicit high-assurance focus
    - Maintain a focus on assurance and other quality attribute requirements, assessing the ability to meet these requirements and analyzing associated failure modes and effects.
    - Conduct systematic event-driven and periodic analyses of artifacts for vulnerabilities and the existence of required quality attributes.

While the acquisition life cycle model must be responsive to evolving needs, it must guard against the intrusion of uncontrolled change that could undermine the integrity of the SoS or one or more of its constituent components.

Types of Acquisition Life Cycle Models

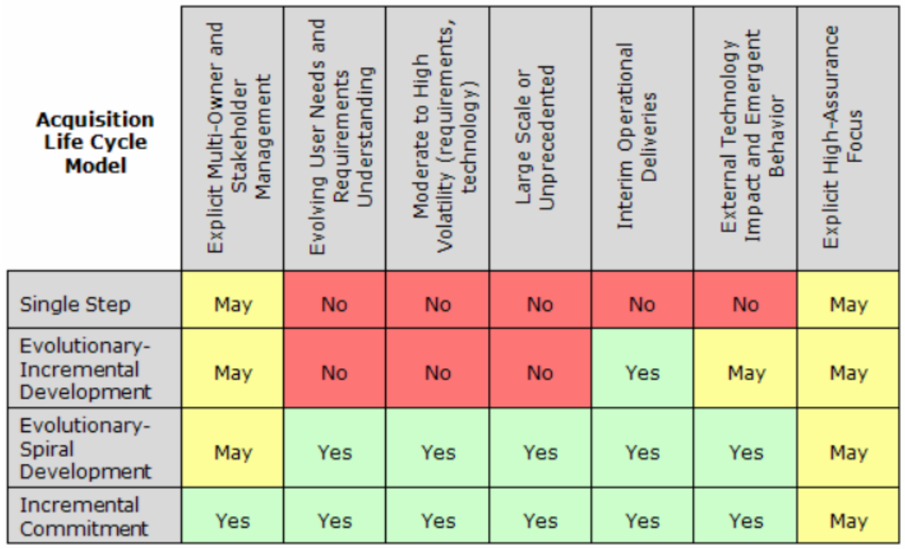
Typical choices for the acquisition life cycle model for major systems have included those in which required capability is delivered through a “single step” (single-step acquisition) and those in which capability is “evolved” through iteration (evolutionary acquisition).

Single step acquisition assumes that requirements are known at the outset and technology is mature. There are two types of evolutionary acquisition, one that incorporates incremental development and one that incorporates evolutionary spiral development. The first type, referred to as evolutionary-incremental, assumes the end-state capability is defined up front but is developed and delivered over time, in increments. This approach is desirable when requirements are relatively well understood and there is a demand for early deployment of capabilities, or when technology planned for use in later increments requires further development. The second type of evolutionary acquisition, referred to as evolution-evolutionary-spiral, assumes the end-state capability cannot be fully defined up front. Instead, requirements are defined (and may be developed and deployed) over time as user needs evolve and technology matures. The end-state thus evolves as well.

Recently, Boehm et al. introduced the Incremental Commitment Model [Boehm 2007]. The ICM extends the evolutionary-spiral model to emphasize a high degree of stakeholder involvement, interim milestones to commit to the next activity, and increased iteration and concurrency.

Table 3 maps acquisition life cycle models to the SoS characteristics discussed above.

Table 3. SoS characteristics accommodated by various life cycle models



Acquisition Life Cycle Model	Explicit Multi-Owner and Stakeholder Management	Evolving User Needs and Requirements Understanding	Moderate to High Volatility (requirements, technology)	Large Scale or Unprecedented	Interim Operational Deliveries	External Technology Impact and Emergent Behavior	Explicit High-Assurance Focus
Single Step	May	No	No	No	No	No	May
Evolutionary-Incremental Development	May	No	No	No	Yes	May	May
Evolutionary-Spiral Development	May	Yes	Yes	Yes	Yes	Yes	May
Incremental Commitment	Yes	Yes	Yes	Yes	Yes	Yes	May

The table indicates that both the evolutionary-spiral model and the incremental commitment model are reasonable choices. However, the incremental commitment model (ICM) has an advantage because it is specifically designed to deal with complex stakeholder dynamics and to integrate human, hardware, and software aspects of development in an SoS context.

The Incremental Commitment Model

The ICM incorporates the following five process principles as critical success factors [Boehm 2007]:

- Stakeholder Satisficing: Identify and engage key stakeholders (i.e., those critical to success) early and often to validate and re-validate requirements, solutions, and plans and to discuss potential and proposed changes.
- Incremental and Evolutionary Growth of System Definition and Stakeholder Commitment: Establish and apply a robust framework for evolution, including stakeholder interaction and agreement processes that gradually build understanding and trust, enabling objective tradeoff analyses and resulting in incremental commitment to plans to move forward.
- Iterative System Development and Definition: Iteratively refine tradeoff analyses, requirements, solutions, and plans based on new information, new needs, and new technologies.
- Concurrent System Definition and Development: Define, analyze, and refine requirements and solutions concurrently, especially in environments in which legacy and commercial components factor into the solution. Refine requirements as more is known about constraints on the solution and to adapt to changes in mission and business needs.
- Risk Management Through Risk-Driven Anchor-Point Milestones: Apply risk-driven milestones to synchronize and stabilize concurrent, iterative, evolutionary activities and products. Evaluate business, technical, and operational feasibility by independent experts, discuss risks and risk management plans, and decide whether or not to proceed.

Anchor-point milestones are focused on the high degree of concurrency surrounding complex systems development and the stakeholder commitments needed to move forward. These milestones look at concurrent activities, which may span independently developed systems, with a view toward synchronizing, stabilizing, and assessing risk before obtaining commitment to proceed to the next development phase. The ICM includes the following anchor-point milestones: Exploration Commitment Review, Valuation Commitment Review, Architecture Commitment Review, Development Commitment Review, and one or more Operations Commitment Reviews. These reviews are discussed by Boehm and Lane in their paper, Using the Incremental Commitment Model to Achieve Successful System Development [Boehm 2007] along with general pass-fail criteria for the milestones.

ICM principles and anchor point milestones are highly applicable to dealing with the acquisition strategy drivers we have identified for systems of systems. While the ICM does not explicitly focus on all relevant dependability and supportability quality attributes, ICM life cycle activities and anchor points can easily accommodate them. Table 4 summarizes quality and assurance focus strategies related to each ICM principle. These strategies would be incorporated into each ICM anchor point milestone.

Table 4. Key principles/critical success factors of the Incremental Commitment Model [Boehm 2007]

ICM Principle	Related Assurance and Quality Strategy
Stakeholder Satisficing	With key stakeholders, identify and evaluate tradeoffs regarding assurance and quality attribute requirements, including developing business and mission usage scenarios and assurance cases. Focus heavily on maintaining regular communications around these attributes, scenarios, and assurance cases throughout the life cycle.
Incremental and Evolutionary Growth of System Definition and Stakeholder Commitment	Focus on an assured hardware/software infrastructure on which functionality can be built and modified per incremental discovery of mission and business needs; identify assurance case requirements for infrastructure. Develop prototypes around complex problems, quality attribute scenarios, assurance cases, and ability to deal with change. Ensure these prototypes go beyond user interface demonstrations toward highly critical infrastructure, assurance, and quality-attribute related requirements.
Iterative System Development and Definition	Develop “living” artifacts and data that reflect history, snapshots of the current working baseline (state), and projections toward candidate future states. Comprehensively and proactively identify and manage needed changes. Iteratively evaluate assurance and other quality attributes, revisiting quality attribute scenarios and assurance cases. Incrementally deliver capabilities, as appropriate, to enable early use, problem identification, and additional requirements discovery.
Concurrent System Definition and Development	At anchor point milestones, conduct activities to synchronize quality attribute analyses across components, stabilize requirements and performance with respect to quality attributes, and assess associated risks before moving to the next phase.
Risk-Driven Anchor Point Milestones	Exploration Commitment Review: Ensure the exploration phase plan includes tasks, deliverables, and resources related to developing scenarios and requirements for quality attributes and developing an assurance case framework. Valuation Commitment Review: Obtain commitment to the set of quality attribute scenarios and requirements and the assurance case framework. Architecture Commitment Review: Analyze the high-level operational concept, requirements, architecture, and plans and document and prepare mitigation plans for assurance and quality risks. Also, update quality attribute scenarios and requirements and develop assurance criteria to apply to architecture evaluation. Development Commitment Review: In addition to the artifacts examined in the Architecture Commitment Review, analyze detailed architecture artifacts. Evaluate artifacts with respect to quality attribute requirements and assurance cases. Ensure sufficient robustness and regression testing and analysis activities are incorporated into the development phase, including those that exercise quality-related scenarios. Require a risk-based verification plan that focuses on robustness of critical system services in the face of uncertainty, complexity, and change.

	Operations Commitment Review: Analyze high-risk artifacts and threads and associated test results against quality attribute requirements and assurance cases. Evaluate results of test and analysis activities carried out in the previous phase and repeat as risk assessments indicate. Conduct risk based, end-to-end evaluations of the system in a simulated SoS environment, if possible.

While the ICM may be a suitable acquisition life cycle model for an SoS, the model only provides a framework for what is needed. Successful application of any life cycle model to a particular acquisition requires details such as the length of major life cycle phases; activities, reviews, and deliverables for each phase; milestones and interim decision points; key capability and quality focus areas; major participants, roles, and responsibilities; and other information.

We now turn to another important acquisition strategy element, the risk management approach.

RISK MANAGEMENT APPROACH

The risk management approach strategy element establishes a process to identify, analyze, and manage risks to achieving acquisition program goals and objectives. It also identifies known risk areas and describes associated risk mitigation plans. A sound risk management approach produces information and strategies that can be pivotal to the success or failure of an acquisition. Typically, risk management has focused on only a single system, and often only on development risks. For systems of systems, the approach has to reach beyond these boundaries to other constituents and across the life cycle from before acquisition begins to operations and sustainment.

Considerations for Systems of Systems

Risk management represents a good starting place for reviewing how an existing acquisition process addresses SoS. Risk analysis is typically an early step in an acquisition process, and for an SoS that risk analysis is an especially critical step in selecting the appropriate acquisition strategies. For an SoS or for the more likely case of a system or component that participates in an existing SoS, an effective risk management approach should

- scale to size and complexity of systems of systems
- incorporate dynamics
- integrate across full life cycle: requirements to sustainment
- focus on success as well as failure

Scale to Size and Complexity of Systems of Systems

Most traditional risk management methods do not have the capabilities needed to address the complexity that arises when the resources of multiple systems are routinely combined in pursuit of a single mission. State-of-the-practice risk management approaches provide stakeholders with a long list of potential problems (i.e., risks). The list for a single project, program, or technology often includes more than one hundred statements of risk. For systems of systems, that list of potential problems can be overwhelming.

Incorporate Dynamics

The dynamic nature of the operational environment also means that the risks are dynamic. As work processes cross business units and multiple organizations, change becomes increasingly difficult to control, and any changes might invalidate the existing risk analysis. An SoS involves multiple typically independent risk assessments for the participating systems. Changes in the risk mitigations for individual systems can affect the collective behavior of the SoS.

Software failure analysis in this context may require a different model of accidents than that used for hardware. Hardware failure analysis typically relies on event-based models of accidents. Such models, with their relatively simple cause-and-effect links, were created in an era of mechanical systems and then adapted for electromechanical systems. The use of software in engineered systems has removed many of the physical constraints that limit complexity and has allowed engineers to incorporate greatly increased complexity and coupling in systems containing large numbers of dynamically interacting components. In the simpler systems of the past, where all the interactions between components could be predicted and handled, component failure was the primary cause of accidents. In today’s highly complex systems, this is no longer the case.

The nature of failures is also dynamic. Discrepancies between the expected and the actual arise frequently in the normal course of business processes. Discrepancies can be thought of as stresses that may drive a business process into an unacceptable state. Stress types include interactions, resources, and people. Missing, inconsistent, or unexpected data are examples of interaction stresses, whereas resource stresses may include excessive network latency, insufficient capacity, and unavailable services. People stresses can consist of information overload that slows analysis, distraction (too much browsing) and a “Not my job” attitude, which can inhibit effective responses to problems.

Integrate Across Full Life Cycle: Requirements to Sustainment

Often the issue is not the risk management method but its execution. In practice, risk management might be applied separately for security threats with other assessments required for interoperability, performance, business, and project risks. It is not unusual for an acquisition to focus on risks that affect costs or schedule. Risk management solutions should look across a broad spectrum of success and failure drivers to provide an overall picture of the potential for success. The uncertainty that exists for an SoS also applies to risks. Business risks identified during development may lead to changes in requirements or reengineering of aspects of the software architecture.

Focus on Success as Well as Failure

Risk management must also consider opportunities that motivated the organization and provide a capability for managers to sort through the inherent complexity of multi-organizational missions to determine their chances of succeeding.

The combination of a seemingly unending list of risks and the focus of most traditional risk management methods on such hazards encourages a focus on prevention, but that emphasis on prevention also fosters a mindset of “playing not to lose.” Managers following this philosophy can easily lose sight of what needs to be accomplished to achieve a successful outcome, with overall failure often the end result [Alberts 2007].

Several Approaches to Risk Management

There are no existing best practices to address these risk management problems. Complexity is unavoidable and analysis cannot consider all factors. However, experience associated with the Mission-Oriented Success Analysis and Improvement Criteria (MOSAIC) and Survivability Analysis Framework (SAF) projects at the Software Engineering Institute (SEI) does suggest some approaches for SoS risk management that may assist today’s practitioner in dealing with problems that have no demonstrated solutions [Alberts 2007, Ellison 2008].

Both MOSAIC and SAF stress the importance of maintaining traceability with business usage and requirements. This is a typical requirement for risk analysis, which now has to be satisfied for a complex organization and operational environment such as that for an SoS. Carefully defining the usage context is a critical step that helps to focus analysis on the most essential system and business factors and hence places some constraints on the complexity. An objective for SAF is to construct a shared view of a system and its role in a business process to enable better communication among business stakeholders, management, developers, support staff, and users to more precisely define the context for the analysis. The construction of that shared view includes

- defining the success criteria of the operational business
- selecting the qualities to be evaluated and monitored to accommodate operational change

- building one or more detailed operational business flows incorporating information about the selected qualities
- identifying stresses (by people, systems, or external events) that individually or collectively push the operational processes beyond the limits of acceptable degradation and recovery

For example, the business process might be a medical situation in which an emergency room physician needs to prescribe tests. Defining the context of that event is critical, as different contexts could have different success criteria and different stresses. Is the prescription written on paper, entered by the physician on a nearby hospital computing terminal, or entered by the physician on a portable device assigned to him? Is the laboratory that processes the test a hospital unit that uses the hospital's computing system or an external laboratory that has been contracted to provide this service? How might that context change over time? An effective risk management approach for this situation and others involving multiple interacting systems and users must define and answer such context questions. Especially in the complex environments discussed in this article, a thorough understanding of the operational context is essential for risks to be adequately identified, analyzed, and mitigated.

SUMMARY

This article has described the effects of drivers associated with systems of systems on key elements of the acquisition strategy. The challenge is that while these drivers can affect current acquisitions, recommended practices for dealing with them do not yet exist. Under these conditions, acquirers need to be aware of the limitations of their current practices when applied in the context of a system of systems and be adaptable in managing the problems that do arise.

A first step we have suggested is to consider the effects a system of systems context has on the acquisition strategy. These effects may then be used to drive development of relevant acquisition strategy elements, such as the acquisition life cycle model and risk management approach. A complementary step would be to generate a list of problems that might arise for such an acquisition and consider how these should drive strategy. For example, changes in mission or business needs, emerging technologies, or other external factors may lead to implementation changes that cannot be accommodated given schedule and resource constraints. Similarly, complex dependencies among systems may expose life cycle processes that are inadequate for managing interaction, coordination, insight, and technical expert reviews of the emerging system, leading to serious deficiencies in the technical solution and inability to deliver. The analysis of such potential problems may then be applied in tailoring and incorporating new choices for acquisition strategy elements, such as MOSAIC for risk management or the ICM as the life cycle model.

REFERENCES

URLs are valid as of the publication date of this document.

[Alberts 2007]	Alberts, Chris, Dorofee, Audrey, and Marino, Lisa. Executive Overview of SEI MOSAIC: Managing for Success Using a Risk-Based Approach (CMU/SEI-2007-TN-008). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2007.
[Boehm 2007]	Boehm, Barry and Lane, Jo Ann. Using the Incremental Commitment Model to Achieve Successful System Development (USC-CSSE-2007-710). Los Angeles, CA: Center for Systems and Software Engineering, University of Southern California, 2007.
[DAU 2012]	Defense Acquisition University (DAU). Defense Acquisition Guidebook, (2012).
[Ellison 2008]	Ellison, Robert J., Goodenough, John, Weinstock, Charles, and Woody, Carol. Survivability Assurance for System of Systems (CMU/SEI-2008-TR-008). Pittsburgh, PA, Software Engineering Institute, Carnegie Mellon University, 2008.
[Esterbrook 2007]	Esterbrook, Steve. “Scale Changes Everything: Understanding the Requirements for Systems of Systems.” Keynote address, 6th IEEE International Conference on COTS-based Software Systems, Baniff, CA, 2007.
[Howard 2007]	Howard, Michael. “Lessons Learned from Five Years of Building More Secure Software.” MSDN Magazine, November 2007.
[Maier 1998]	Maier, Mark. “Architecting Principles for Systems of Systems.” Systems Engineering 1, 4 (1998): 267-84.
[Schwartz 2007]	Schwartz, John. “Who Needs Hackers?” New York Times, September 12, 2007.
[Ward 2006]	Ward, Mary Catherine, Elm, Joseph P., and Kushner, Susan. Techniques for Developing an Acquisition Strategy by Profiling Software Risks (CMU/SEI-2006-TR-002). Pittsburgh, PA, Software Engineering Institute, Carnegie Mellon University, 2006.