SYM-AM-26-123

Excerpt from the Proceedings

of the

Twenty-Third Annual Acquisition Research Symposium and Innovation Summit

Thursday, May 7, 2026 Sessions
Volume II
"Accelerating Warfighting Capabilities"

Acquisition Software Management Planning (SMP) Is Critical to Mission Success

Published: April 30, 2026

Approved for public release; distribution is unlimited.

Prepared for the Naval Postgraduate School, Monterey, CA 93943.

The research presented in this report was supported by the Acquisition Research Program, Graduate School of Defense Management at the Naval Postgraduate School.

To request defense acquisition research, please contact:

Acquisition Research Program
Department of Defense Management
Naval Postgraduate School
E: arp@nps.edu
www.acquisitionresearch.net

Copies of Symposium Proceedings and Presentations; and Acquisition Sponsored Faculty and Student Research Reports and Posters may be printed from the **NPS Defense Acquisition & Innovation Repository** at **https://dair.nps.edu/**.

ACQUISITION RESEARCH PROGRAM
DEPARTMENT OF ACQUISITION, FINANCE, AND MANPOWER
NAVAL POSTGRADUATE SCHOOL

Acquisition Software Management Planning (SMP) Is Critical to Mission Success

Carol Woody—is a principal researcher in the CERT Division of the Software Engineering Institute (SEI) at Carnegie Mellon University. She leads a team that builds capabilities and competencies for measuring, managing, and sustaining software assurance and cybersecurity for highly complex software-intensive systems and supply chains throughout the Acquisition life cycle. She coauthored the book Cyber Security Engineering: A Practical Approach for Systems and Software Assurance, which was published by Pearson Education as part of the SEI Series in Software Engineering. The CERT Cybersecurity Engineering and Software Assurance Professional Certificate is based on the research she led. [cwoody@cert.org]

Mike Bandor—is a senior software engineer in the CERT Division of the Software Engineering Institute (SEI) at Carnegie Mellon University. He leads teams that enable organizations to enhance the predictable performance and mission assurance in the acquisition, evolution, and operations of software-reliant systems. He has more than 36 years of experience with military systems, including business systems, command and control systems, satellite systems, ground systems, aircraft, and ground-based radar systems. Prior to joining the SEI in 2005, he was an enlisted computer programmer in the U.S. Air Force, and he retired with almost 23 years on active duty. [mbandor@sei.cmu.edu]

Abstract

Today's systems are increasingly software intensive, complex, and reliant on third-party technology. We live in a world of systems of systems linked by software that connects services and hardware and essentially removes many previous human and geographic restrictions. Unfortunately, acquisition practices have not kept pace with these changes. Leadership is still primarily monitoring cost and schedule. Today's systems can be assembled faster and cheaper because software is rarely built for its intended use. Instead, much of it is reused, sourced from third parties (and increasingly from open source sites), but with increased risk. All software contains potential vulnerabilities that increase the risk of experiencing successful cyber attacks. It is critical to ensure that system requirements are met without extraneous behaviors that would jeopardize the mission. This paper explains why effective software management is critical to the acquisition of today's systems, which are primarily software intensive. It also shares lessons learned in current efforts underway to build and implement a Software Management Plan (SMP) in major Department of War (DoW) acquisitions and describes the research underway to improve how software is monitored and managed.

Overview of the Software Challenges

The use of software in acquisition is growing exponentially both as a component within every technology acquisition and as the enabler for creating software components. Software is also used as an enabler to connect legacy systems with new capabilities to enhance existing operational processes. Software is fast becoming the primary enabler of the functionality that systems provide. However, for most acquisition programs, software is "behind the scenes" facilitating a wide range of activities that were previously handled by hardware but with greater flexibility and lower cost.

Some of this software is built as part of the acquisition, but much of it is reused from similar platforms, brought into each software component from existing language libraries, or pulled from open source sites. For the prime contractor, only a finite portion of the software in the acquisition is built for purpose. Much of it is repurposed from other similar projects, acquired from subcontractors, linked into a system from specialized service providers (e.g., Cloud), or downloaded from open source sites. The acquirer must then confirm that these pieces that come from a range of sources and built by a range of development processes meet the intent of the contract governing the acquisition. An example of software component composition is shown in Figure 1.

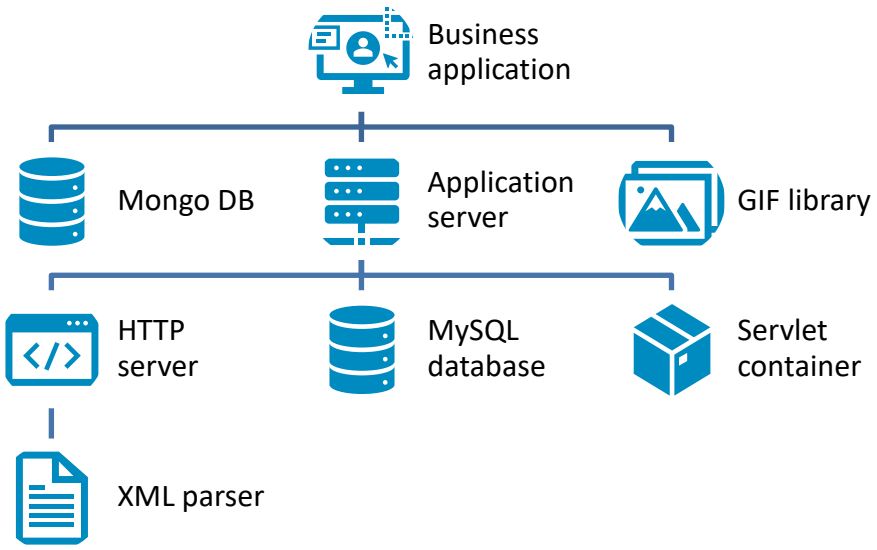

Figure 1. Assembly of Software from Components

To deliver systems that operate successfully in today's contested environments, which is becoming the norm for the Department of War (DoW), software must be assembled and maintained with high assurance. This result does not come automatically. Instead, it requires extensive planning and monitoring. All software contains potential vulnerabilities that increase the risk of successful cyber attacks (Woody et al., 2014). Therefore, it is critical to ensure that system requirements are met without extraneous behaviors that would allow attackers to jeopardize the mission.

There is a wide range of recent guidance that focuses on improving the speed of software delivery, such as DevSecOps guidance (DoD, 2024) and Software Acquisition Pathway guidance (DoD, 2020). In addition, there is recent guidance for program protection that identifies and addresses risks that software and the software supply chain represent to the mission (DoW, 2025). However, connecting this guidance to each acquisition is a challenge. While there is a great deal of software within every acquisition, it is spread somewhat invisibly throughout the system.

The requirements that are prepared for each acquisition are allocated to various system components, some of which may be outsourced. Each component is decomposed into subcomponents, which are composed of hardware, software, and firmware elements, that will be built by different processes and integrated to deliver the planned functionality. This decomposition is reflected in the Work Breakdown Structure (WBS) for the acquisition. If the component is outsourced, integration occurs first at the subcomponent level and then at the component level. For major systems, the link to software does not appear until levels 4 and 5 in the decomposition and WBS, and this link is not always clearly defined until Milestone B (DoD, 2022).

The primary focus of acquisition program management is typically on cost and schedule for the delivery of a well-engineered system that meets the mission. However, with software, it is possible to have a system that functions as intended but is riddled with vulnerabilities, which make it too easy to attack (Woody et al., 2026). The challenge in managing this software is twofold:

- Potential vulnerabilities can be introduced at each step in the development life cycle, and delays in identifying and removing these vulnerabilities will result in higher costs and potential schedule delays as the software is integrated into larger pieces of the system.
- Managing these many pieces of software is impossible without consistent standards that apply across the system. The acquisition process needs to include good software engineering processes and practices that will reduce the introduction of vulnerabilities and steps for managing software to ensure it conforms to the acceptable level of quality.

There are many ways to address these two challenges within an acquisition. The keys are to have a plan for how they will be managed and the resources with the appropriate knowledge to execute the plan across the life cycle. Moving into software development without a plan for how it will be managed from an acquisition perspective and confirmation that the vendors are capable of building and delivering software at the acceptable level of rigor puts the program at high risk. Too many acquisitions focus only on obtaining Software Bill of Material (SBOMs) to meet a compliance mandate (Wallen, 2023), but these will not provide a way of determining where potential vulnerabilities are located and their potential risk to the program. A Software Management Plan (SMP) is needed that is built on solid software management and assurance practices that are well integrated into the system acquisition to ensure that the software is acquired, integrated, tested, and implemented successfully to meet mission requirements.

An SMP is not the same as a Software Development Plan (SDP). An SDP is typically written by the developing organization or contractor and is focused on the development and implementation of the system. An SMP is written by the Program Management Office (PMO) and describes how the software aspects of the acquisition will be managed, what activities are to be performed (including oversight), and how they are performed. Unlike the SDP, the SMP addresses the entire lifecycle of the system. The intent of the SMP is to get program offices to think about the software side of the acquisition much earlier in the acquisition life cycle and not leave those decisions until much later in the overall system acquisition.

Five Key Pillars of Software Assurance Critical to Software Acquisition Management

While the software development processes and practices may vary greatly among programs, our team of researchers in cybersecurity in acquisition and system and software engineering have identified five foundational capabilities that must be well established for effective program results (Woody, 2026). These capabilities (i.e., pillars) are Software Requirements, Software Supply Chain Risk Management, Software Quality, System Integration, and Software Metrics. How each of these capabilities is addressed for a specific program varies greatly, but each one is critical to effective software management. An SMP should include the processes, practices, and responsible resources that will implement each of these capabilities. Inconsistencies, mistakes, and disconnects in these five areas can lead to poor software management decisions and gaps in effective software risk management for both organic software development and software from the supply chain.

These gaps can include improper assumptions about how protections will perform when communications and data sharing occur among the various software and hardware components of the system, including interactions with the hardware platforms that execute the software. There are also gaps in the control of the contents of data packets and other information exchanges that can facilitate attackers. Further misunderstandings occur when designs are created for hardware, but the resulting function is handled by software that may have inherent vulnerabilities. Complex software integrations are assembled from existing components and

language libraries that are reused to perform similar actions but may not be built to specification. Also, reused components can include unexpected capabilities and attack surfaces that permit unacceptable behaviors that attackers can leverage. We next consider each pillar and its contribution to software management.

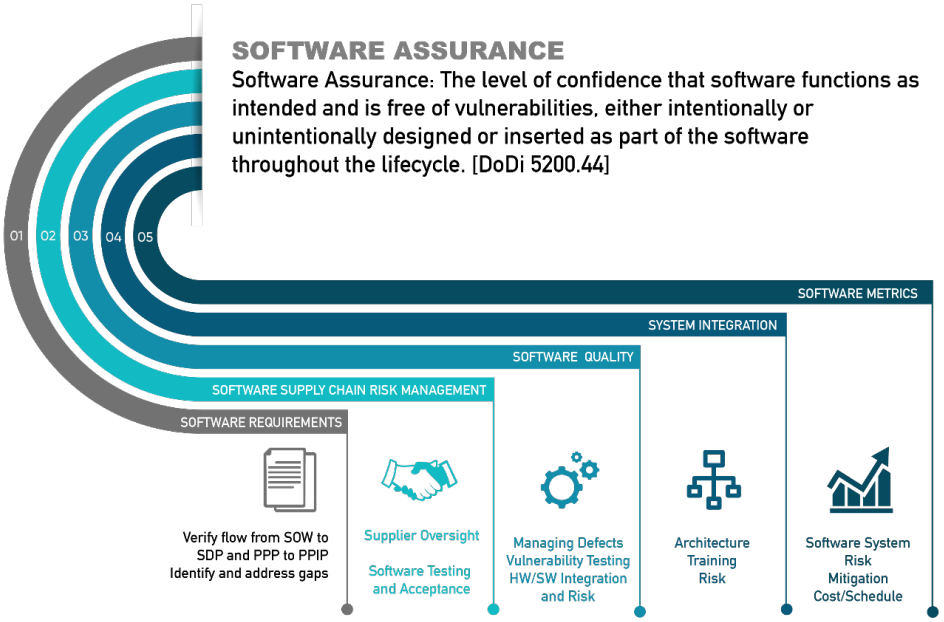

Figure 2. The Five Pillars of Software Assurance Critical to Software Acquisition Management

The first pillar is Software Requirements. This pillar defines the required functionality assigned by the system design to the software/firmware components of the system to perform their allocated capabilities within acceptable constraints and quality to support the system mission. Software developers translate functional requirements into software requirements that feed a software pipeline. The constraints and controls that are placed at the system level should be translated into potential misuse and abuse cases that bound the software and establish what the software will not do and what is not wanted.

Typically, the allocated functional requirements are complete, since they are part of the executable mission that the software must address. However, specific qualities (e.g., performance, reliability, safety, and security) are too frequently incomplete. Decisions made by engineers trained in hardware without software expertise make incorrect assumptions that can severely impact the resulting software. In one program, an assumption was made that the software had no reliability issues because it did not wear out like hardware. This assumption caused the program to ignore software in its safety criticality analysis, which resulted in an insufficient failure mode analysis. Steps for verifying software requirements to ensure they are complete, accurate for the mission, and verifiable need to be part of the SMP.

In addition to the functional requirements levied on a system, there should be other requirements for the development of the system that focus on avoiding system and software vulnerabilities (e.g., the use of secure coding standards, fuzz testing, and penetration testing). Insight into monitoring those additional requirements throughout the acquisition and how they will be verified needs to be a component of the SMP.

Requirements relevant to software can be extracted from the Statement of Work (SOW) and mapped to the SDP to confirm contractor planning, but the SMP should describe the

monitoring to ensure these requirements are delivered as planned. Additional software-related cybersecurity requirements are identified in the Program Protection Plan (PPP) and mapped to the contractor-delivered Program Protection Implementation Plan (PPIP), and the SMP should describe how the acquisition will be monitored to ensure effective implementation.

The second pillar is Software Supply Chain Risk Management. This pillar governs and monitors the people, processes, and technology needed to acquire, manage, and sustain third-party software/firmware to ensure it meets system requirements and mission needs throughout its implementation. Using third-party software can offer significant advantages in cost and schedule, but it can also create cybersecurity and life cycle dependency risks that must be managed. When software is acquired, there is an ongoing relationship with the owner of this intellectual property. Managing this vendor relationship needs to be integrated with the system life cycle so that bugs, new features, and vulnerabilities will be addressed by the software owner and transferred to the acquirer securely. It is critical to ensure that the third party can address problems, deliver a software product that shows attention to issues, and use secure transport mechanisms so that the software is not compromised in transit (Ellison et al., 2010). Open source software (OSS) is estimated to be used by 90% of organizations (Gehring, 2022). For OSS, access to the source code is provided, but fixes must usually be posted back to the source provider as required by the license agreement, which provides visibility of these fixes to others who have also installed the code.

Our research has found that decisions about software and its sources are influenced throughout the life cycle in program management, engineering, supplier dependency management, technology support functions, independent assessment and compliance, and process management (Alberts et al., 2023). These processes, practices, and responsible resources for all aspects of software supply chain risk management need to be clearly defined in a SMP to ensure completeness, effective integration, and consistency.

The third pillar is Software Quality. This pillar builds confidence that the people, processes, and practices used to create and acquire software are delivering needed mission results within acceptable levels of rigor and risk. Too often, quality reviews focus only on how well the software development process is performed and do not look at the actual quality of the product that the process is delivering. Higher quality code has been shown to have fewer defects and fewer vulnerabilities, which increases its ability to perform effectively in high-assurance environments (Woody et al., 2014). How quality will be evaluated and who will be responsible for its monitoring should be clearly described in the SMP.

The fourth pillar is System Integration. IEEE defines integration as the process of combining software components, hardware components, or both into an overall system (Institute of Electrical and Electronics Engineers, 2017). Software does not function in isolation, so analyzing and testing it in isolation is only a piece of the puzzle that must be solved to make it functional with sufficient assurance within a system. The process steps and responsibilities for how the software is integrated into the system and verified within that context are too often ad hoc. This provides opportunities for tampering and introducing vulnerabilities into the system because of its integration with other software, firmware, and hardware. An effective integration process should be part of the SMP for both new software and third-party software.

The fifth, and final, pillar is Software Metrics. This pillar identifies and analyzes metrics to demonstrate effective life cycle software assurance to meet mission needs. At the product level, there can be a diversity of metrics that correspond to a particular quality attribute of the system under development and process attributes that relate to the conduct of the engineering activity. These metrics provide objective data that can be used to evaluate the progress, process, and quality of the delivered software. A well-structured SMP will leverage

available metrics to monitor acquisition progress and provide early indications of potential problems. As part of our research, we assembled a range of metrics that are typically available from software development activities that can be leveraged in building the specific approach used for software management (Woody et al., 2019).

The selected metrics, which are appropriate to the mission and development approach, are used by the program, and will be analyzed and used for monitoring software, should be clearly described in the SMP. A key aspect to consider is that monitoring must address the full life cycle. Trending will be important in determining whether the qualities the metrics are supporting are improving. The data collected in the early steps of the life cycle (i.e., requirements, design, and architecture phases) can be predictive of how the software will perform, and information collected in later life cycle steps should verify the results (see Figure 3).

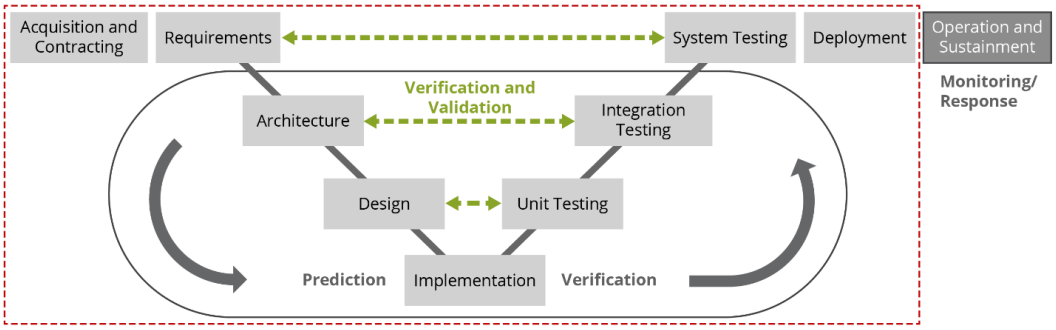

Figure 3. Life Cycle Considerations for Metrics

Defining Threats Critical to Mission Protection

In addition to effective processes and practices for software creation, integration, and monitoring, determining the threats that represent critical risks to mission success need to be analyzed to determine the level of risk that software within the system must address. There are many approaches to threat identification and management. It is critical that the steps for identifying and addressing software threats that will impact the mission be included in the SMP. There are many threat-modeling approaches available. However, the program must ensure that software threats, including those that arise from the software supply chain, are considered. The volume of threats can be overwhelming, and prioritization that balances budget and scheduling constraints with potential impacts of the threats must be considered. The Security Engineering Risk Analysis (SERA) approach (Alberts et al., 2014) can deliver this capability. The SMP will need to include which steps to take to identify and prioritize software threats, handle mitigations, and perform ongoing monitoring to identify when the risk profile for software changes. Software requirements must reflect the mitigations selected for high-priority threats, and enhanced testing should verify that planned mitigations are in place and functioning as expected.

Software Management Plan

In today's technology-driven environments, the importance of software to the proper functioning of a system cannot be overstated. Every acquirer must ensure that the software, which is a key part of the acquired system, is effectively developed and supported to meet mission requirements. This is a complex process. Many acquisitions further increase this process's complexity by spreading the responsibility for software across many parts of the organization. All groups addressing software must collaborate to ensure that they all apply consistent and effective software management. Without effective planning, a well-engineered system that is delivered with software that meets needs consistently and continually cannot be

assumed. This planning must describe the processes and practices to be applied across the software within the system's acquisition, which includes the software used in the pipelines to build the software products as well as the third-party selection and the integration process used.

This material is based upon work funded and supported by the Department of War under Air Force Contract Nos. FA8702-15-D-0002, and FA870225DB003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM26-0325

References

Alberts, C., Bandor, M., Wallen, C., & Woody, D. (2023, October 2). *Acquisition security framework (ASF): Managing systems cybersecurity risk (expanded set of practices)* (Technical Note CMU/SEI-2023-TN-004). https://doi.org/10.1184/R1/24128475

Alberts, C., Woody, D., & Dorofee, A. (2014, December 4). *Introduction to the security engineering risk analysis (SERA) framework* (Technical Note CMU/SEI-2014-TN-025). https://doi.org/10.1184/R1/6574856.v1

DoD. (2020, October 2). *Operation of the software acquisition pathway* (DoD Instruction 5000.87). https://www.esd.whs.mil/Portals/54/Documents/DD/issuances/dodi/500087p.PDF

DoD. (2022, May 13). *Department of Defense standard practice, work breakdown structures for defense materiel items* (MIL-STD-881F). https://quicksearch.dla.mil/qsDocDetails.aspx?ident_number=36026

DoD. (2024, October). *DoD enterprise DevSecOps fundamentals*, Version 2.5. DoD Software Modernization Senior Steering Group. https://dodcio.defense.gov/Portals/0/Documents/Library/DoD%20Enterprise%20DevSecOps%20Fundamentals%20v2.5.pdf

Department of War Under Secretary of War Research and Engineering. (2025, October 28). *Program protection plan outline & guidance*, Version 2.0. Department of War. https://aaf.dau.edu/storage/2025/11/PPP-OG-v2.0.pdf

Ellison, R., Goodenough, J., Weinstock, C., & Woody, D. (2010, May 1). *Evaluating and mitigating software supply chain security risks* (Technical Note CMU/SEI-2010-TN-016). https://doi.org/10.1184/R1/6573497.v1

Gehring, W. (2022). *The state of open source software*. Octoverse. https://octoverse.github.com/2022/

Institute of Electrical and Electronics Engineers (2017, August 28). *ISO/IEC/IEEE International Standard - Systems and software engineering—Vocabulary (*ISO/IEC/IEEE 24765:2017(E)). https://doi.org/10.1109/IEEESTD.2017.8016712

Wallen, C., Alberts, C., Bandor, M., & Woody, C. (2023, June 14). *Software bill of materials framework: Leveraging SBOMs for risk* reduction. https://www.sei.cmu.edu/documents/5364/Leveraging_SBOM_for_Risk_Reduction.pdf

Woody, D., Alberts, C., Bandor, M., & Chick, T. (2026, March 4). *The five pillars of software assurance in system acquisition*. https://doi.org/10.58012/r8q1-zp76

Woody, C. & Bandor, M. (2026, March 11). *Acquisition oversight for software assurance* [Webinar]. Software Engineering Institute (SEI). https://www.sei.cmu.edu/library/acquisition-oversight-for-software-assurance/

Woody, C., Ellison, R., & Nichols, B. (2014, December 22). *Predicting software assurance using quality and reliability measures* (Technical Note CMU/SEI-2014-TN-026). https://doi.org/10.1184/R1/6582113.v1

Woody, D., Ellison, R., & Ryan, C. (2019). *Exploring the use of metrics for software assurance* (Technical Note CMU/SEI-2018-TN-004). https://doi.org/10.1184/R1/12366842.v1


Acquisition Research Program
Department of Acquisition, Finance, and Manpower
Naval Postgraduate School
555 Dyer Road, Ingersoll Hall
Monterey, CA 93943

www.acquisitionresearch.net\