Software Engineering and Systems Engineering in the Department of Defense

Mark Schaeffer
Office of the Undersecretary of Defense for Acquisition and Technology

As the Department of Defense (DoD) moves toward acquisition without rigid DoD standards, acquisition is becoming less an issue of hardware vs. software or software engineering vs. systems engineering—it is becoming an integrated whole. New DoD directives, an integrated Capability Maturity Model, highly trained acquisition personnel, and a partnership with industry to manage risk, will help us meet the challenges of acquisition in the next century.

Twenty years ago, complex machines were controlled by hydraulic actuators or mechanical linkages using position switches and analog control circuitry. Today, much of the mechanical or hydraulic equipment and almost all analog circuitry has been replaced by servo motors, digital position encoders, and microprocessors. Complex machines still have a large mechanical component, but their control is often digital rather than analog. Presently, every DoD development project involves digital technology at some level in the work breakdown structure—either digital control systems, automated test equipment, or training devices. Software engineering has become a critical and key component of the engineering development process for both military and commercial product development.

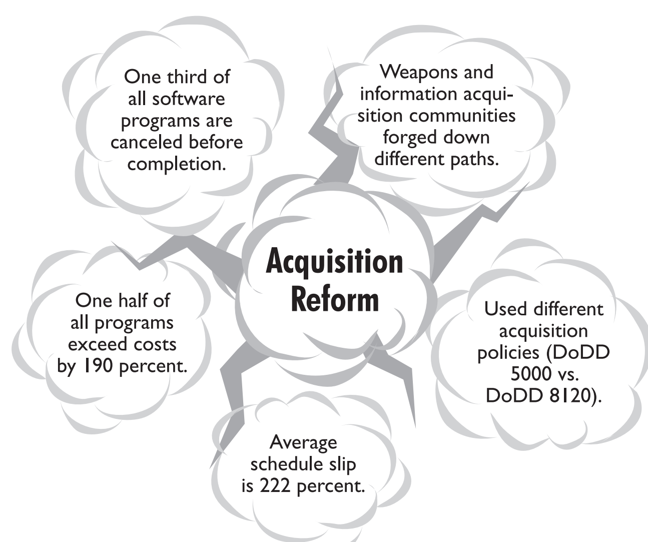
Over the past 20 years, our systems engineering policy and practice has evolved alongside the revolution in digital technology. However, as digital technology evolved in its importance in control systems, the software needed to make them work properly became more problematic. Our track record using DoD Directive (DoDD) 5000.1 for weapons systems acquisition and DoDD 8120.1 for information systems acquisition was not good and was not getting better. In 1991, the DoD published its first acquisition reform study, and in June 1993, Deputy Undersecretary of Defense for Acquisition Reform Colleen Preston opened the DoD's acquisition reform office. Prior to the official start of acquisition reform, studies of software engineering noted frequent and serious schedule, cost, and performance slippages. Figure 1 illustrates the issues.

As acquisition reform gained momentum, the DoD embraced a key conceptual change in policy. Instead of developing and procuring hardware and software using different processes, the DoD sought to acquire weapons systems that met stated performance objectives. We moved away from telling contractors how to build a product and toward defining the end performance of a product we would buy. We sometimes refer to this as the Performance-Based Business Environment. We no longer require that electronic components meet and be tested in accordance with rigid military standards. Instead, we asked developers to show us that their products could operate over a military operational mission profile, could be effectively and efficiently maintained, and would have the required operational life span. Commercial design and development practices are encouraged, and blind adherence to military standards is not allowed. This shift of philosophy was influenced by another reality of the 1990s: The number of new-start major programs and new-start non-major programs was decreasing dramatically when compared to the 1970s and 1980s.

Acquisition reform provided the impetus and the structure for the DoD to take the initial step toward incorporating software engineering into the larger context of systems engineering. But was not software engineering unique or at least fundamentally different from the rest of the development process?

Our analysis confirmed our belief that the objectives of software engineering, the problems encountered managing it, and the techniques used to resolve those problems are essentially the same as those found in systems engineering. For example, the management of risk in software-intensive programs is fundamentally the same as for any other type of program. This similarity led us to combine the treatment that software and hardware risk management receives in

Figure 1. *The pre-acquisition reform environment.*



Managing the Job
- Integrated Product and Process Development (IPPD)
- Planning and Estimating
- Work Breakdown Structure
- Contracting
- Personnel and Resources
- Integrated Program Management
- Engineering Management
- Requirements Management
- Interface Management
- Configuration Management
- Risk Management
- Policy
- Licenses
- Logistics
- Training

Developing Product
- IPPD
- Product Lines
- Modeling and Simulation
- Design
- Integration
- Trade-off Studies
- Open Systems
- Continuous Process Improvement
- Reliability
- Interoperability
- Producibility
- Maintainability
- Non-Developmental Item
- Government-off-the-Shelf Software and Commercial-off-the-Shelf Software

Assuring Product
- IPPD
- Quality
- Verification and Validation
- Inspections
- Measurement
- Tracking
- Sustainment
- Supportability
- Traceability
- Test and Evaluation
- Safety

Special Considerations
- Architecture
- Programming Languages
- Modification and Upgrade
- Post-Deployment Support

Figure 2. *Elements of systems acquisition.*

the Defense Acquisition Deskbook. Figure 2 shows the activities that are common to systems engineering and software engineering.

We did find a few software engineering activities that deserve special consideration, but dealing with them in the systems engineering context is not difficult. We found that

- Architecture selection requires a global approach early in the design process because these architecture selections affect many other design choices downstream and are costly to reverse.
- With relaxation of the Ada mandate, there is a greater choice of development languages, which increases program risk. Mitigating this risk is accomplished using rigorous business case analyses for language selection.
- As hardware systems age, they generate a larger pool of trained maintainers; however, rapid technological change reverses this process for software. Searching for programmers to meet Year 2000 demands is an example of a work force that although technologically advanced is not able to meet post-deployment support needs.
- Post-deployment support is common to both hardware and software but different in its application.

There is no software equivalent of a form-fit-function replacement using a different product. Similar to post-deployment support, modification issues are not unique to software. Although software modification holds the promise of greater and easier system improvement, it has not, in practice, generated the promised efficiencies and benefits. Software is not as malleable as originally believed, nor are "tweaks" necessarily as low risk and inexpensive as hoped.

Dealing with each of these areas is not beyond our ability; the key lies in early planning and having a risk reduction effort. Once we saw the significant degree of commonality between the elements of systems engineering and software engineering, we set out to deal with both of them in DoD acquisition policy.

In March 1996, the DoD created a single acquisition process that "states the policies and principles for all DoD acquisition programs ..." when it issued DoDD 5000.1, Defense Acquisition, and DoD Regulation 5000.2-R, Mandatory Procedures for Major Defense Acquisition Programs and Major Automated Information System Acquisition Programs. It also canceled DoDD 8120.1, Life-Cycle Management of Automated Information Systems, and provided program managers the means to manage their programs using a total systems approach, to optimize total system performance, and to minimize total ownership costs. Our belief that systems engineering and

Figure 3. *DoD acquisition management key policies.*

DoDD 5000.1 Defense Acquisition

"Software is a key element in DoD systems. It is critical that software developers have a successful past performance record, experience in the software domain or product line, a mature software development process, and evidence of use and adequate training in software methodologies, tools, and environments" (DoDD 5000.1, para. 2.k).

DoDD 5000.2 Mandatory Procedures for Major Defense Acquisition Programs and Major Automated Information Systems Acquisition Programs

"Software shall be managed and engineered using best processes and practices that are known to reduce cost, schedule, and performance risks. It is DoD policy to design and develop software systems based on systems engineering principles ..." (DoD 5000.2-R, para. 4.3.5).

To include
- Developing software system architectures that support open system concepts.
- Exploiting COTS products.
- Identifying and exploiting software reuse.
- Selecting a programming language in the context of systems and software engineering factors (ASD (C31) memo, April 29, 1997).
- Use of DoD standard data (DoDD 8320.1).
- Selecting contractors with
  - Domain experience in comparable systems.
  - Successful past performance record.
  - Demonstrable mature software process.
- Use of software metrics.
- Assessing information warfare risks IAW DoDD TS-3600.1.

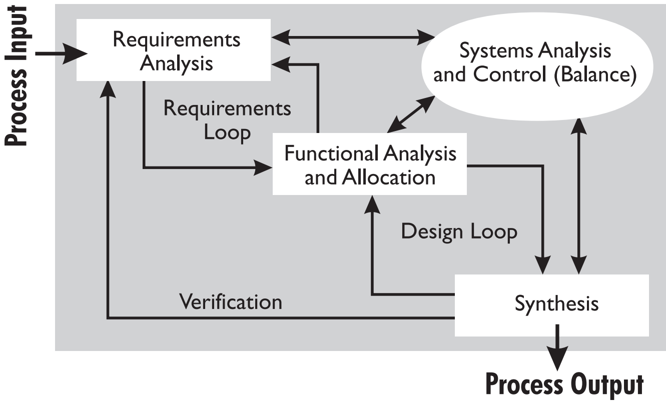


Figure 4. *The systems engineering process.*

software engineering are integral was recently validated when we updated DoD Regulation 5000.2-R to comply with the Clinger-Cohen Act. Most of the revisions were not technical in nature but consisted of adding to existing sections of the regulation references to the Act. Figure 3 summarizes how we implemented provisions of the Clinger-Cohen Act and integrated software engineering policy into the framework of systems engineering and major program acquisition.

So, how does the software development process integrate into the systems engineering process? There are certain aspects of the systems engineering process that are highly correlated, while others are more difficult to match up. The system engineering process, as addressed in the Defense Acquisition Deskbook, is comprised of four elements:

- Translating stated operational requirements into an integrated product design using a systematic, concurrent approach.
- Transitioning multidisciplinary technical inputs (including concurrent engineering of manufacturing, logistics, and testing) into a coordinated effort to meet program cost, schedule, and performance objectives.
- Ensuring functional and physical interface compatibility so system definition and design meet all hardware, software, facilities, people, and data requirements.
- Establishing a risk management program to reduce risk early through system element tests and demonstrations.

Software engineering management uses both sequential and nonsequential processes that correspond to activities in the systems engineering process. The waterfall model, shown in Figure 5, is a sequential process that has the following characteristics:

- Do the steps in a specified order.
- Define all the requirements upfront.
- Use comprehensive reviews as gates.
- Complete program design before coding.
- Emphasizes functional and allocated baselines.
- "Do the job twice if possible and involve the customer."

Figure 6 illustrates how steps in the waterfall model correspond to iterations of the systems engineering model.

There are several other models for software lifecycle management, and they all can be mapped to the iterative systems engineering process—even those models designed to handle ill-defined user requirements.

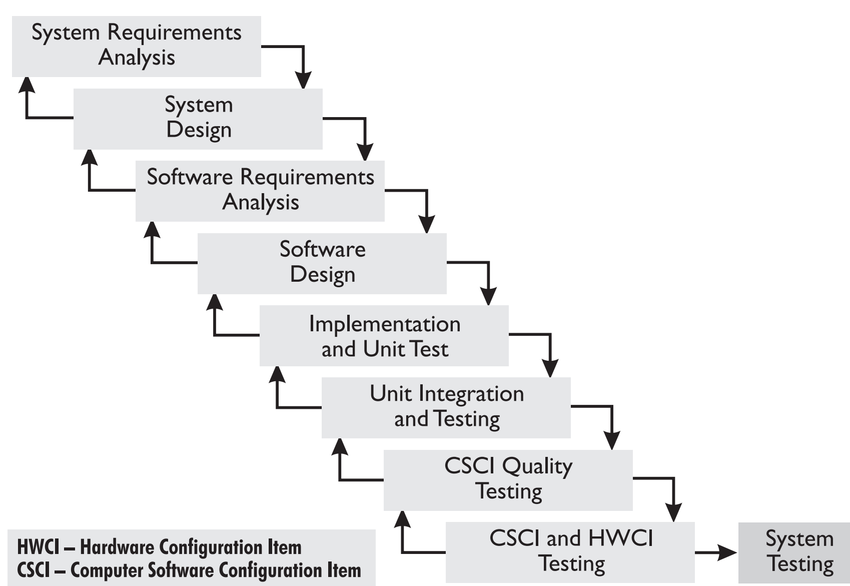
Our conclusion, reached several years ago, and confirmed repeatedly since then, is that software engineering is an integral part of systems engineering. The systems engineering management techniques we have honed and the process improvement efforts we have undertaken consolidate well with similar processes and improvement efforts in software engineering. We are about to embark on the next step in recognizing software engineering as an integral part of the systems engineering process.

Where Will We Focus Our Future Effort?

First, we are engaged in a dialog with the command, control, communications, and intelligence community about consolidating regulations that affect information technology acquisition. There are a number of duplicative and overlapping regulations that have their genesis in the days when there were different "stovepipes" for weapons systems and automated information system acquisition. As the information revolution progresses and gains momentum, virtually all acquisition will contain information technology. Our intent is to derive the most synergism we can from the DoD's implementation of the Clinger-Cohen Act while making good on our commitment to provide clear, unambiguous, and realistic guidance to program managers.

Second, to take advantage of the information revolution, we must have an acquisition work force that has the

Figure 5. *The waterfall model.*



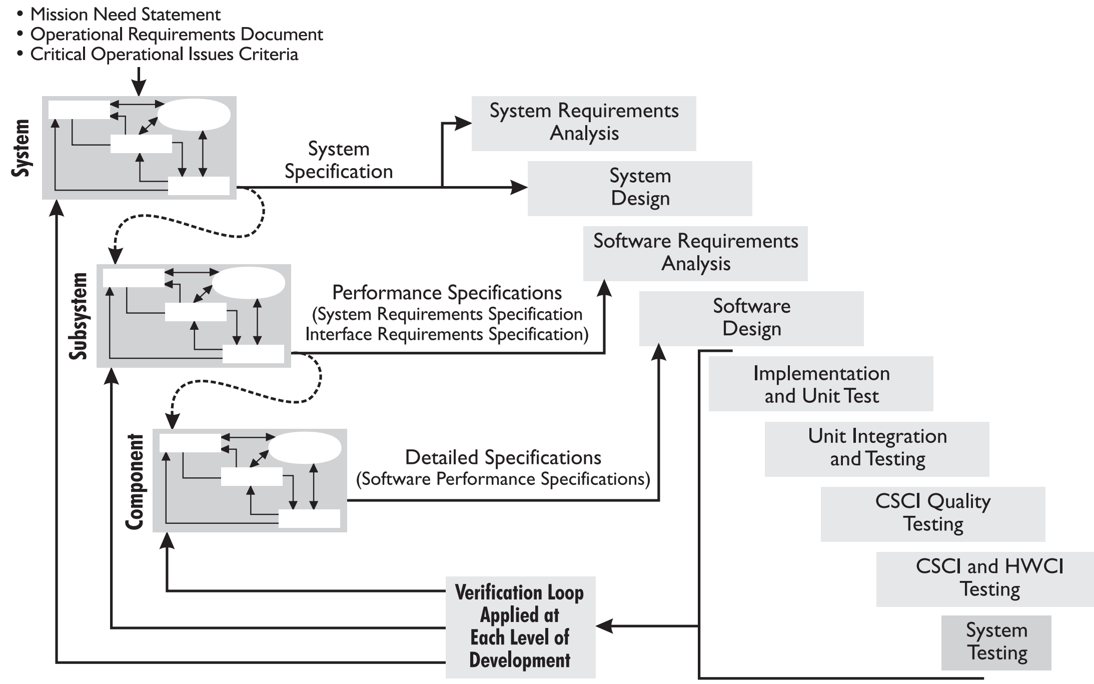


Figure 6. *The relationship between software engineering and systems engineering.*

education and training to manage complex integrated hardware and software engineering activities. We have initiated a comprehensive upgrade of the training materials used by the Defense Acquisition University for acquisition work-force training. This upgrade will incorporate software engineering management into the resident, nonresident, and distance learning curriculum. Our goal is to have an acquisition work force that is capable of maximizing the benefit of the process improvements we are putting in place.

Third, we are in a partnership with industry to develop new practices, procedures, and techniques for the management and reduction of risk of complex weapons systems acquisitions. The National Defense Industrial Association is supporting extensive work in this area, and it holds the promise of big dividends helping the DoD complete the transition from the "how to" of military specifications and standards to the performance-oriented environment of acquisition reform.

Last but not least, we are engaged in a comprehensive integration of Capability Maturity Models (CMM) originally developed by the Software Engineering Institute, and now being championed by industry at large. The initial common framework effort will be based on the software CMM, the systems engineering capability model, and the Integrated Process Development (IPD) CMM. Other functional disciplines may be added later. The work accomplished to date on the software CMM, Version 2.0, and the IPD CMM have been included in the initial CMM Integration baseline. The goal is to improve efficiency, return on investment, and effectiveness by using models that integrate disciplines such as systems engineering and software engineering—disciplines that are inseparable in a systems development endeavor.

As we cross the millennium, we are committed to developing and implementing a complete framework for the management of acquisition within the DoD. The days of separate hardware and software acquisition is gone. The challenge of creating a new, integrated systems engineering and software engineering framework is a daunting one, but we are up to the task. ◆

About the Author

Mark Schaeffer is the deputy director for systems engineering in OUSD(A&T). He is responsible for policy and implementation of systems engineering, technical risk management, manufacturing, quality, reliability and maintainability, acquisition logistics, modeling and simulation, and software engineering.

Schaeffer has over 20 years experience in weapons systems acquisition and program management in the Office of the Secretary of Defense, Naval Sea Systems Command, and as congressional staff. He has served in several challenging management positions within the Naval Sea Systems Command, to include deputy program manager and technical director of the MK-48/ADCAP Torpedo Project; program manager of the CV/Amphib Firefighting Improvement Program; and special assistant to the Shipyard Planning officer and chief design engineer at Mare Island Naval Shipyard.

Schaeffer has a bachelor's degree in mechanical engineering from California State University at Sacramento and has completed graduate studies at Massachusetts Institute of Technology, Duke University, and Georgetown University.

E-mail: mschaeff@acq.osd.mil